Alexander Gelbukh (Ed.)

Computational Linguistics and Intelligent Text Processing

9th International Conference, CICLing 2008
Haifa, Israel, February 17-23, 2008
Proceedings

Springer

Volume Editor

Alexander Gelbukh
National Polytechnic Institute (IPN)
Center for Computing Research (CIC)
Col. Nueva Industrial Vallejo, 07738, Mexico City, Mexico
E-mail: gelbukh@gelbukh.com

Library of Congress Control Number: 2008920439

CR Subject Classification (1998): H.3, I.2.7, I.7, I.2, F.4.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN	0302-9743
ISBN-10	3-540-78134-X Springer Berlin Heidelberg New York
ISBN-13	978-3-540-78134-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12227754 06/3180 5 4 3 2 1 0

Preface

CICLing 2008 (www.CICLing.org) was the 9th Annual Conference on Intelligent Text Processing and Computational Linguistics. The CICLing conferences are intended to provide a wide-scope forum for the discussion of both the art and craft of natural language processing research and the best practices in its applications.

This volume contains the papers accepted for oral presentation at the conference, as well as several of the best papers accepted for poster presentation. Other papers accepted for poster presentation were published in special issues of other journals (see the information on the website). Since 2001 the CICLing proceedings have been published in Springer's Lecture Notes in Computer Science series, as volumes 2004, 2276, 2588, 2945, 3406, 3878, and 4394.

The book consists of 12 sections, representative of the main tasks and applications of Natural Language Processing:

– Language resources
– Morphology and syntax
– Semantics and discourse
– Word sense disambiguation and named entity recognition
– Anaphora and co-reference
– Machine translation and parallel corpora
– Natural language generation
– Speech recognition
– Information retrieval and question answering
– Text classification
– Text summarization
– Spell checking and authoring aid

A total of 204 papers by 438 authors from 39 countries were submitted for evaluation (see Tables 1 and 2). Each submission was reviewed by at least two independent Program Committee members. This volume contains revised versions of 52 papers by 129 authors from 24 countries selected for inclusion in the conference program (the acceptance rate was 25.5%). In addition, the volume features invited papers by

– Ido Dagan, Bar Ilan University, Israel,
– Eva Hajičová, Charles University, Czech Republic,
– Alon Lavie, Carnegie-Mellon University, USA, and
– Kemal Oflazer, Sabancı University, Turkey,

who presented excellent keynote lectures at the conference. Publication of extended full-text invited papers in the proceedings is a distinctive feature of the

Table 1. Statistics of submissions and accepted papers by country or region

Country or region	Authors	Papers[1]	
	Subm	Subm	Accp
Argentina	2	0.5	0.5
Australia	3	1	–
Belgium	3	1	1
Brazil	15	5.67	2.67
Canada	6	2.83	1
China	26	12.92	–
Czech Republic	12	5.67	0.67
Estonia	1	1	1
Finland	2	1.5	1
France	28	12.14	5.5
Germany	21	11.75	5.75
Greece	1	0.33	–
Hong Kong	5	1.75	–
Hungary	4	2	–
India	42	19	2.33
Iran	2	0.33	–
Israel	17	6.83	1.33
Italy	12	7.25	4.25
Japan	18	9.33	1.33
Jordan	1	0.17	–
Korea, South	20	6	–
Lithuania	2	1	–
Macau	4	1	–
Mexico	23	11.67	3.58
Netherlands	5	3.67	2.67
Poland	5	2	–
Portugal	2	1	–
Romania	29	13.86	1
Russia	10	5.92	0.67
Saudi Arabia	1	0.33	–
Serbia and Montenegro	2	1	–
Spain	56	23.75	5.17
Sweden	2	1.5	0.5
Switzerland	6	2.33	1.67
Taiwan	4	1.5	1.5
Tunisia	5	3	–
Turkey	6	3.5	2.5
United Kingdom	6	3.5	0.67
United States	29	14.5	3.75
Total:	*438*	*204*	*52*

[1] Counted by authors. E.g., for a paper by 3 authors, 2 from Mexico and 1 from USA, we added $\frac{2}{3}$ to Mexico and $\frac{1}{3}$ to USA.

CICLing conferences. What is more, in addition to the presentation of their invited papers, the keynote speakers organized separate vivid informal events; this is also a distinctive feature of this conference series.

The following papers received the Best Paper Awards and the Best Student Paper Award, correspondingly (the best student paper was selected from papers where the first author was a full-time student, excluding the papers that received a Best Paper Award):

1st Place:	*Discovering word senses from text using random indexing*, by Niladri Chatterjee and Shiwali Mohan;
2nd Place:	*Non-interactive OCR post-correction for giga-scale digitization projects*, by Martin Reynaert;
3rd Place:	*Lexical cohesion-based topic modeling for summarization*, by Gonenc Ercan and Ilyas Cicekli;
Student:	*SIGNUM – a graph algorithm for terminology extraction*, by Axel-Cyrille Ngonga Ngomo.

The authors of the awarded papers were given extended time for their presentations. In addition, the Best Presentation Award and the Best Poster Award winners were selected by a ballot among the attendees of the conference.

Table 2. Statistics of submissions and accepted papers by topic [2]

Accepted	Submitted	Topic
16	49	Lexical resources
13	41	Statistical methods (mathematics)
12	52	Information extraction
10	37	Symbolic and linguistic methods
9	47	Information retrieval
9	33	Clustering and categorization
8	35	Semantics and discourse
7	26	Machine translation
6	34	Formalisms and knowledge representation
6	18	Morphology
5	32	Text mining
5	24	Other
5	16	Word Sense Disambiguation
4	9	Summarization
3	30	Syntax and chunking (linguistics)
3	5	Spell checking
2	10	Anaphora resolution
2	6	Text generation
1	24	Natural language interfaces
1	6	Speech processing
1	3	Emotions and humor
0	15	POS tagging
0	10	Parsing algorithms (mathematics)

[2] According to the topics indicated by the authors. A paper may be assigned to more than one topic.

Besides their high scientific level, one of the success factors of CICLing conferences is their excellent cultural program. CICLing 2008 was held in Israel, at the ancient cultural and religious crossroads of our civilization. The participants enjoyed tours to Jerusalem with its Wailing Wall, Via Dolorosa, Church of the Holy Sepulture, and the remnants of King David's City, Nazareth, Sea of Galilee, an old Jewish burial site Beit She'arim with underground catacombs, a beautifully preserved Roman city of Beit Shean, the Beatitude monastery ("Sermon on the Mount"), Arab Quarter of Haifa, and Bahai Gardens—the world center of the Baháı faith; see photos at www.CICLing.org.

I would like to thank all those involved in the organization of this conference. In the first place these are the authors of the papers constituting this book: it is the excellence of their research work that gives value to the book and sense to the work of all of the other people involved. I thank the Program Committee members for their hard and very professional work. Very special thanks go to Manuel Vilares and his group, Ted Pedersen, Nicolas Nicolov, and Mohamed Abdel Fattah for their invaluable support in the reviewing process.

I am extremely grateful to the University of Haifa for the organizational and financial support. The generous contributions of the Faculty of Social Sciences,

the Rector's Office and the President's office helped us keep the registration fees reasonably low. I am particularly happy to thank the Caesarea Edmond Benjamin de Rothschild Foundation Institute for Interdisciplinary Applications of Computer Science (CRI) at the University of Haifa for the tremendous administrative help. This conference would have been impossible to run without the help of the CRI: Prof. Martin Charles Golumbic, Rona Perkis, Avital Berkovich, George Karpatyan, and Orly Ross. I greatly enjoyed my collaboration with Shuly Wintner on the preparation of the conference—he is a wonderful organizer and host, and it is his enthusiasm and hard work that made this conference a reality.

I thankfully acknowledge the generous financial support of the Center for Complexity Science in Israel. The entire submission, reviewing, and selection process, as well as putting together the proceedings, was supported for free by the EasyChair system (www.EasyChair.org); I express my gratitude to its author Andrei Voronkov for his constant support and help. Last but not least, I deeply appreciate the Springer staff's patience and help in editing this volume—it is always a great pleasure to work with them.

January 2008 Alexander Gelbukh

Organization

CICLing 2008 was co-organized by the University of Haifa, Israel, and the Natural Language and Text Processing Laboratory of the Center for Computing Research (CIC) of the National Polytechnic Institute (IPN), Mexico. It was held at the University of Haifa and hosted by the Computational Linguistics Group and Caesarea Edmond Benjamin de Rothschild Foundation Institute for Interdisciplinary Applications of Computer Science (CRI) at the University of Haifa.

Program Chair

Alexander Gelbukh

Program Committee

Eneko Agirre
Christian Boitet
Nicoletta Calzolari
John Carroll
Kenneth Church
Dan Cristea
Walter Daelemans
Barbara Di Eugenio
Claire Gardent
Alexander Gelbukh
Gregory Grefenstette
Eva Hajicova
Yasunari Harada
Eduard Hovy
Nancy Ide
Diana Inkpen
Aravind Joshi
Dimitar Kazakov
Alma Kharrat
Adam Kilgarriff
Alexander Koller
Sandra Kuebler
Hugo Liu
Aurelio Lopez Lopez
Diana McCarthy
Igor Mel'cuk
Rada Mihalcea
Masaki Murata
Nicolas Nicolov
Kemal Oflazer
Constantin Orasan
Manuel Palomar
Ted Pedersen
Viktor Pekar
Stelios Piperidis
James Pustejovsky
Fuji Ren
Horacio Rodriguez
Vasile Rus
Ivan Sag
Franco Salvetti
Serge Sharoff
Grigori Sidorov
Thamar Solorio
Oliviero Stock
John Tait
Linda C. Van Guilder
Karin Verspoor
Manuel Vilares Ferro
Yorick Wilks
Dekai Wu
Annie Zaenen

Award Committee

Alexander Gelbukh
Eduard Hovy
Rada Mihalcea
Ted Pedersen
Yorick Wiks

Additional Referees

Jisha Abubaker
Iãki Alegria
Miguel A. Alonso
Muath Alzghool
Xabier Arregi
Fco. Mario Barcala
Sara Carrera
Hakan Ceylan
Victor Darriba
Alexander Dikovsky
Denys Duchier
Mohamed Abdel Fattah
Marcello Federico
Milagros Fernandez-Gavilanes
Sergio Ferrández
Óscar Ferrández
Davide Fossati
Mary Ellen Foster
Oana Frunza
Alfio Massimiliano Gliozzo
Jos M. Gómez
Jorge Graña
Marco Guerini
Carlos Gómez-Rodríguez
Eva Hajicova
David Hope
Aminul Islam
Steliana Ivanova
Ruben Izquierdo
Heng Ji
Sylvain Kahane
Cynthia Kersey
Rob Koeling
Anna Korhonen
Zornitsa Kozareva
Svitlana Kurella
Kow Kuroda
Fotis Lazaridis
Mihai Lintean
Erica Lloves-Calviño
Miguel Angel Molinero Alvarez
Andrea Mulloni
Fernando Magán-Muñoz
Sergio Navarro
Roberto Navigli
María Pardiño
Jurgen Paulus
Emanuele Pianta
Juan Otero Pombo
Judita Preiss
Marcel Puchol-Blasco
Jonathon Read
Francisco Ribadas-Pena
Rachel Roxas
Kepa Sarasola
Martin Scaiano
Luciano Serafini
Chung-chieh Shan
Ravi Sinha
Georg Sommer
Aitor Soroa
Mark Stevenson
Rajen Subba
Swati Tata
Stefan Thater
Masatsugu Tonoike
Alberto Tretti
Jesus Vilares
David Weir
Taras Zagibalov
Beñat Zapirain

Organizing Committee

Shuly Wintner (Chair)	George Karpatyan
Rona Perkis	Orly Ross
Avital Berkovich	

Website and Contact

The website of the CICLing conferences is www.CICLing.org. It contains information on the past CICLing events and satellite workshops, abstracts of all published papers, photos from all CICLing events, and video recordings of some keynote talks, as well as the information on the forthcoming CICLing event. Contact: gelbukh@cicling.org, gelbukh@gelbukh.com; more contact options can be found on the website.

Table of Contents

A Distributed Database System for Developing Ontological and Lexical Resources in Harmony

Aleš Horák[1], Piek Vossen[2], and Adam Rambousek[1]

[1] Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno
Czech Republic
hales@fi.muni.cz, xrambous@fi.muni.cz
[2] Faculteit der Letteren
Vrije Universiteit van Amsterdam
e Boelelaan 1105, 1081 HV Amsterdam
The Netherlands
Piek.Vossen@irion.nl

Abstract. In this article, we present the basic ideas of creating a new information-rich lexical database of Dutch, called Cornetto, that is interconnected with corresponding English synsets and a formal ontology. The Cornetto database is based on two existing electronic dictionaries - the Referentie Bestand Nederlands (RBN) and the Dutch wordnet (DWN). The former holds FrameNet-like information for Dutch and the latter is structured as the English wordnet. In Cornetto, three different collections are maintained for lexical units, synsets and ontology terms.

The database interlinks the three collections and aims at clarifying the relations between them. The organization and work processes of the project are briefly introduced.

We also describe the design and implementation of new tools prepared for the lexicographic work on the Cornetto project. The tools are based on the DEB development platform and behave as special dictionary clients for the well-known DEBVisDic wordnet editor and browser.

1 Introduction

Lexical data and knowledge resources has rapidly developed in recent years both in complexity and size. The maintenance and development of such resources require powerful database systems with specific demands. In this paper, we present an extension of the DEBVisDic environment [1] for the development of a lexical semantic database system for Dutch that is built in the Cornetto project. The system holds 3 different types of databases that are traditionally studied from different paradigms: lexical units from a lexicological tradition, synsets within the wordnet framework and an ontology from a formal point of view. Each of these databases represents a different view on meaning. The database system is specifically designed to create relations between these databases and to allow to edit the information in each. It represents a complex editing environment but also a research tool to study the relations between language, as defined in a lexicon and wordnet, and knowledge, as defined in an ontology.

The paper is further structured as follows. In the Section 2, we will describe the Cornetto project in terms of the design of the database structure and the major editing actions. The Section 3 introduces the DEB platform and the new features that have been introduced for the Cornetto project. Finally, we describe the specific client interface for viewing and editing the data in the Section 4.

2 The Cornetto Project

Cornetto is a two-year Stevin project (STE05039) in which a lexical semantic database is built, that combines Wordnet with FrameNet-like information [2] for Dutch. The combination of the two lexical resources will result in a much richer relational database that may improve natural language processing (NLP) technologies, such as word sense-disambiguation, and language-generation systems. In addition to merging the Wordnet and FrameNet-like information, the database is also mapped to a formal ontology to provide a more solid semantic backbone.

The database will be filled with data from the Dutch Wordnet [3] and the Referentie Bestand Nederlands [4]. The Dutch Wordnet (DWN) is similar to the Princeton Wordnet for English, and the Referentie Bestand (RBN) includes frame-like information as in FrameNet plus additional information on the combinatoric behaviour of words in a particular meaning.

An important aspect of combining the resources is the alignment of the semantic structures. In the case of RBN these are lexical units (LUs) and in the case of DWN these are synsets. Various heuristics have been developed to do an automatic alignment. Following automatic alignment of RBN and DWN, this initial version of the Cornetto database will be extended both automatically and manually.

The resulting data structure is stored in a database that keeps separate collections for lexical units (mainly derived from RBN), synsets (derived from DWN) and a formal ontology (SUMO/MILO plus extensions [5]). These 3 semantic resources represent different view points and layers of linguistic, conceptual information. The alignment of the view points is stored in a separate mapping table. The database is itself set up so that the formal semantic definition of meaning can be tightened for lexical units and synsets by exploiting the semantic framework of the ontology. At the same time, we want to maintain the flexibility to have a wide coverage for a complete lexicon and encode additional linguistic information. The resulting resource will be made available in the form of an XML database.

The Cornetto database provides a unique combination of semantic, formal semantic and combinatoric information.

2.1 Architecture of the Database

Both DWN and RBN are semantically based lexical resources. RBN uses a traditional structure of form-meaning pairs, so-called Lexical Units [6].

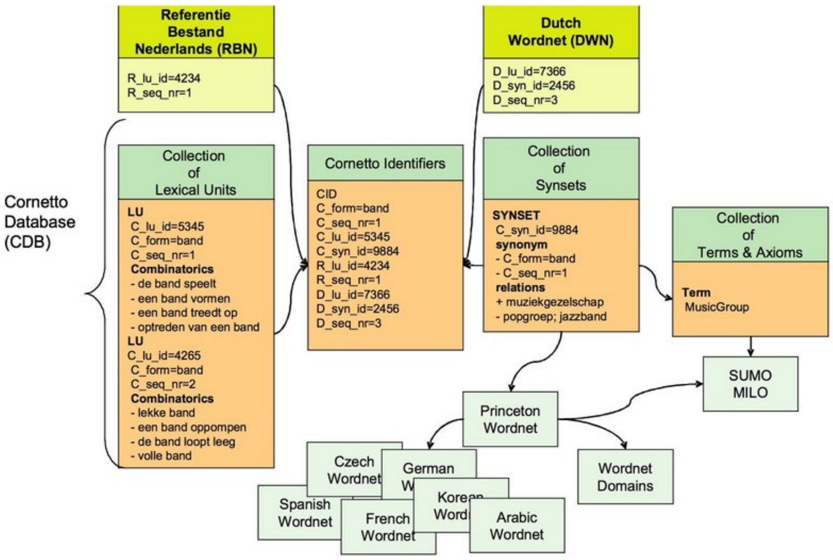


Fig. 1. Data collections in the Cornetto database

The Cornetto database (CDB) consists of 3 main data collections:

1. Collection of Lexical Units, mainly derived from the RBN
2. Collection of Synsets, mainly derived from DWN
3. Collection of Terms and axioms, mainly derived from SUMO and MILO

The Lexical Units are word senses in the lexical semantic tradition. They contain all the necessary linguistic knowledge that is needed to properly use the word in a language. The Synsets are concepts as defined by [7] in a relational model of meaning. Synsets are mainly conceptual units strictly related to the lexicalization pattern of a language. Concepts are defined by lexical semantic relations. For Cornetto, the semantic relations from EuroWordNet are taken as a starting point [3].

Outside the lexicon, an ontology will provide a third layer of meaning. The Terms in an ontology represent the distinct types in a formal representation of knowledge. Terms can be combined in a knowledge representation language to form expressions of axioms. In principle, meaning is defined in the ontology independently of language but according to the principles of logic. In Cornetto, the ontology represents an independent anchoring of the relational meaning in Wordnet. The ontology is a formal framework that can be used to constrain and validate the implicit semantic statements of the lexical semantic structures, both the lexical units and the synsets. In addition, the ontology provides a mapping of a vocabulary to a formal representation that can be used to develop semantic web applications.

In addition to the 3 data collections, a separate table of so-called Cornetto Identifiers (CIDs) is provided. These identifiers contain the relations between the lexical units and the synsets in the CDB but also to the original word senses and synsets in the RBN and DWN.

The Figure 1 shows an overview of the different data structures and their relations. The different data can be divided into 3 layers of resources, from top to bottom:

– The RBN and DWN (at the top): the original databases from which the data are derived;
– The Cornetto database (CDB): the ultimate database that will be built;
– External resources: any other resource to which the CDB will be linked, such as the Princeton Wordnet, wordnets through the Global Wordnet Association, Wordnet domains, ontologies, corpora, etc.

The center of the CDB is formed by the table of CIDs. The CIDs tie together the separate collections of LUs and Synsets but also represent the pointers to the word meaning and synsets in the original databases: RBN and DWN and their mapping relation.

Furthermore, the LUs will contain semantic frame representation. The frame elements may have co-indexes with Synsets from the wordnet and/or with Terms from the ontology. This means that any semantic constraints in the frame representation can directly be related to the semantics in the other collections. Any explicit semantic relation that is expressed through a frame structure in a LU can also be represented as a conceptual semantic relation between Synsets in the Wordnet database.

The Synsets in the wordnet are represented as a collection of synonyms, where each synonym is directly related to a specific LU. The conceptual relations between Synsets are backed-up by a mapping to the ontology. This can be in the form of an equivalence relation or a subsumption relation to a Term or an expression in a knowledge representation language.

Finally, a separate equivalence relation is provided to one ore more synsets in the Princeton Wordnet.

The work is divided in 4 steps:

1. Automatic alignment of the word meanings of the two resources
2. Import of the result of the alignment into the database
3. Import of the SUMO ontology and WordNet domains to the synsets of the Dutch wordnet
4. Manual revision of the lexical units, the synsets and the ontological mapping

In the next paragraphs, we will discuss these steps briefly.

2.2 Aligning Word Meanings

To create the initial database, the word meanings in the Referentie Bestand Nederlands (RBN) and the Dutch part of EuroWordNet (DWN) have been automatically aligned. The word *koffie* (coffee) for example has 2 word meanings in RBN (*drink* and *beans*) and 4 word meanings in DWN (*drink, bush, powder* and *beans*). When we try to automatically align these meanings, we can get a complete match, no match or a partial match between these meanings. This then results in 4, 5, or 6 distinct meanings in the Cornetto database depending on the degree of matching across these meanings. Note that this alignment is different from aligning WordNet synsets because RBN is not structured in synsets. We can for example not use the overlap of synonyms because RBN has no synonyms. For measuring the match, we used all the semantic information that was available in both resources: e.g. definitions and domain labels.

To match word meanings with the same domain label, we first had to normalize the labels. We first cleaned the labels manually (e.g., *pol* and *politiek* can be merged). Next, we measured the overlap in vocabulary associated with each domain. So if the label *oorlog* (war) in RBN is associated with the same words as the label *geweld* (violence) in DWN, we can make these labels equivalent. The overlap was expressed using a correlation figure for each domain in the matrix with each other domain. Domain labels across DWN and RBN do not require an exact match. Instead, the scores of the correlation matrix can be used for associating them.

Overlap of definitions was based on the overlapping normalized content words relative to the total number of content words. For other features, such as part-of-speech, we manually defined the relations across the resources. We only consider a possible match between words with the same orthographic form and the same part-of-speech. The strategies used to determine which word meanings can be aligned are:

1. The word has one meaning and no synonyms in both RBN and DWN
2. The word has one meaning in both RBN and DWN
3. The word has one meaning in RBN and more than one meaning in DWN
4. The word has one meaning in DWN and more in RBN
5. If the broader term (BT or hypernym) of a set of words is linked, all words which are under that BT in the semantic hierarchy and which have the same form are linked
6. If some narrow term (NT or hyponym) in the semantic hierarchy is related, siblings of that NT that have the same form are also linked.
7. Word meanings that have a linked domain, are linked
8. Word meanings with definitions in which one in every three content words is the same (there must be more than one match) are linked.

Each of these heuristics will result in a separate score for all possible mappings between word meanings. In the case of *koffie* (coffee), we thus will have 8 possible matches: RBN1-DWN1, RBN1-DWN-2, RBN1-DWN-3, RBN1-DWN4, . . . , etc, . . . RBN2-DWN-4. For the match RBN meaning 1-DWN meaning 1, we will thus get 8 scores, one for each heuristics. The number of links found per strategy is shown in the Table 1.

To weigh the heuristics, we manually evaluated each heuristics. Of the results of each strategy, a random sample was made of 100 records (800 samples in total). Each sample was checked by 8 persons (6 staff and 2 students). For each record, the word form, part-of-speech and the definition was shown for both RBN and DWN (taken from VLIS). The testers had to determine whether the definitions described the same meaning of the word or not. The results of the tests were averaged, resulting in a percentage of items which were considered good links. The averages per strategy are shown in the Table 1.

Table 1. Results for aligning strategies

	Conf.	Dev.	Factor	LINKS	
1: 1 RBN & 1 DWN meaning, no synonyms	97.1	4.9	3	9936	8.1 %
2: 1 RBN & 1 DWN meaning	88.5	8.6	3	25366	20.8 %
3: 1 RBN & >1 DWN meaning	53.9	8.1	1	22892	18.7 %
4: >1 RBN & 1 DWN meaning	68.2	17.2	1	1357	1.1 %
5: overlapping hypernym word	85.3	23.3	2	7305	6.0 %
6: overlapping hyponyms	74.6	22.1	2	21691	17.7 %
7: overlapping domain-clusters	70.2	15.5	2	11008	9.0 %
8: overlapping definition words	91.6	7.8	3	22664	18.5 %

The minimal precision is 53.9 and the highest precision is 97.1. Fortunately, the low precision heuristics also have a low recall. On the basis of these results, the strategies were ranked: some were considered very good, some were considered average, and some were considered relatively poor. The ranking factors per strategy are:

– Strategies 1, 2 and 8 get factor 3
– Strategies 5, 6 and 7 get factor 2
– Strategies 3 and 4 get factor 1

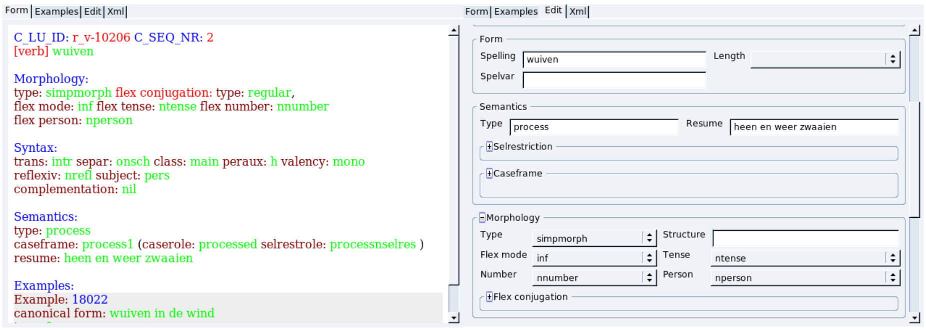
A factor 3 means that it counts 3 times as strong as factor 1. It is thus considered to be a better indication of a link than factor 2 and factor 1, where factor 1 is the weakest score. The ranking factor is used to determine the score of a link. The score of the link is determined by the number of strategies that apply and the ranking factor of the strategies. The final score is normalized to a value between 0 and 1.

In total, 136K linking records are stored in the Cornetto database. Within the database, only the highest scoring links are used to connect WordNet meanings to synsets. There are 58K top-scoring links, representing 41K word meanings. In total 47K different RBN word meanings were linked, and 48K different VLIS/DWN word meanings. 19K word meanings from RBN were not linked, as well as 59K word meanings from VLIS/DWN. Note that we considered here the complete VLIS database instead of DWN. The original DWN database represented about 60% of the total VLIS database. VLIS synsets that are not part of DWN can still be useful for RBN, as long as they ultimately get connected to the synset hierarchy of DWN.

As a result of the alignment, a new list of lexical units and synsets is generated. All the relevant data for these lexical units and synsets are copied from the RBN and DWN, respectively.

2.3 Importing External Data

DWN was linked to WordNet 1.5. WordNet domains are mapped to WordNet 1.6 and SUMO is mapped to WordNet 2.0 (and most recently to WordNet 2.1). In


Form | Examples | Edit | Xml |
Form | Examples | Edit | Xml |

C_LU_ID: r_v-10206 C_SEQ_NR: 2
[verb] wuiven

Morphology:
type: simpmorph flex conjugation: type: regular,
flex mode: inf flex tense: ntense flex number: nnumber
flex person: nperson

Syntax:
trans: intr separ: onsch class: main peraux: h valency: mono
reflexiv: nrefl subject: pers
complementation: nil

Semantics:
type: process
caseframe: process1 (caserole: processed selrestrole: processnselres)
resume: heen en weer zwaaien

Examples:
Example: 18022
canonical form: wuiven in de wind

Form
Spelling: wuiven
Length:
Spelvar:

Semantics
Type: process
Resume: heen en weer zwaaien
Selrestriction
Caseframe

Morphology
Type: simpmorph
Structure:
Flex mode: inf
Tense: ntense
Number: nnumber
Person: nperson
Flex conjugation

Fig. 2. Cornetto Lexical Units, showing the preview and editing form

order to apply the information from SUMO and WordNet domains to the synsets, we need to exploit the mapping tables between the different versions of Wordnet. We used the tables that have been developed for the MEANING project [8,9]. For each equivalence relation to WordNet 1.5, we consulted a table to find the corresponding WordNet 1.6 and WordNet 2.0 synsets, and via these we copied the mapped domains and SUMO terms to the Dutch synsets.

The structure for the Dutch synsets thus consists of:

- a list of synonyms
- a list of language internal relations
- a list of equivalence relations to WordNet 1.5 and WordNet 2.0
- a list of domains, taken from WordNet domains
- a list of SUMO mappins, taken from the WordNet 2.0 SUMO mapping

The structure of the lexical units is fully based in the information in the RBN. The specific structure differs for each part of speech. At the highest level it contains:

- orthographic form
- morphology
- syntax
- semantics
- pragmatics
- examples

The above structure is defined for single word lexical units. A separate structure will be defined later in the project for multi-word units. It will take too much space to explain the full structure here. We refer to the Cornetto website [10] for more details.

2.4 Manual Editing

The aligned data is further manually edited through various cycles of editing. For this purpose, special editing clients have been developed. We will discuss the editing clients in more detail below.

The editing process itself consists of a number of steps, where we will focus on different types of information. In the first cycle, we will manually verify the alignment of word-meanings. For this purpose, selections of words and word meanings are made. This selection involves the following criteria:

- Frequent nouns and verbs
- Words with many meanings
- Lexical units with a mapping to a synset with a low score
- Lexical units without a mapping with a synset

During this work, we typically carry out the following actions:

- Confirm or delete a mapping
- Create another mapping
- Split a single lexical unit in two lexical units
- Merge two lexical units into one
- Add lexical units or delete lexical units
- Split a synset unit in two synsets
- Merge two synsets into one
- Add synsets or delete synsets
- Add or delete synonyms to synsets

At the end of these actions, we will get a new and revised list of senses and mappings to synsets. This will be the new sense and synset structure of the Cornetto database.

The second phase of the editing involves the relation of the synsets to the ontology. The initial mapping is based on a projection of the SUMO labels to the synsets via the equivalence relations. These assignments will be revised, where we foresee two possible relations:

- a synset is a name for a SUMO term: there is direct equivalence
- a synset is defined through a KIF expression [11] that involves one or more SUMO terms.

In the last case, the synset does not name a disjunct ontological type but a lexicalization of a certain conceptualization of such a type or relation between types. For example, the next Dutch words do not require creating a new type in the ontology but will be defined as instances of the type Water in or used for specific purposes:

- zwemwater = water that is good for swimming
- drinkwater = water that is good for drinking
- zeewater = water from the sea
- rivierwater = water from a river
- theewater = water for making tea
- koffiewater = water for making coffee
- bluswater = water that is or can be used for extinguishing fire

The precise semantic implications for these concepts will be expressed by a list of triplet relations in the database, as in the following KIF expression for *rivierwater*:

```
(instance, 0, Water)
(instance, 1, River)
(origin, 0, 1)
```

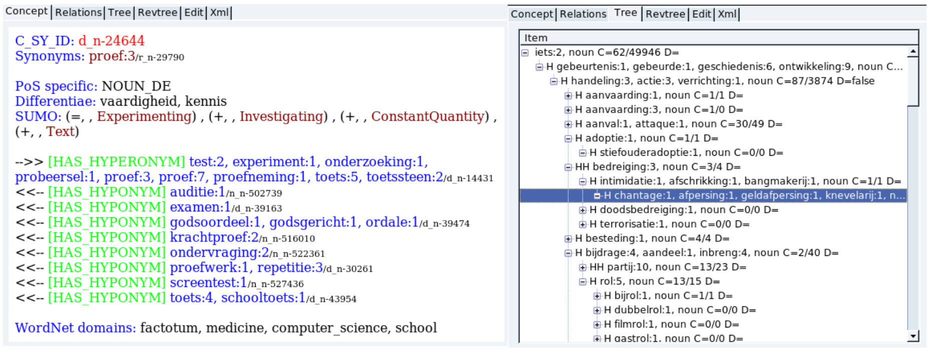
This expression can be paraphrased as “there is an instance of Water and there is an instance of River such that the former originates from the latter.” The numbers in this expression represent variables and we assume that the variable 0 corresponds to the referent of the defined synset.

During this phase, it may also be necessary to revise the hypernym relations in DWN to form a proper semantic hierarchy that is in line with the ontological decisions. During this phase, we also will formulate constraints on ontological mappings and synset relations. These constraints will be applied to all the assigned ontology relations. Any violation will be flagged and edited. Violations can follow from direct assignments or from assignments that are inherited downward through hyponymy relations.

In the final phase of the project, the editing will focus on the correlations between the frame-structures in the lexical units and the synsets. This process is called micro-level alignment. The frame structures for the lexical units, specify argument slots for verbs. These slots are now specified using semantic labels that are defined in RBN. In addition, we provided positions for pointers to synsets and pointers to SUMO labels. An example for a case frame for *genezen* (to cure) is given below:

```
<semantics_verb>
  <sem-type>action</sem-type>
  <sem-caseframe>
    <caseframe>action2</caseframe>
    <args>
      <arg>
        <caserole>agent</caserole>
        <selrestrole>agentanimate</selrestrole>
        <synset_list/>
      </arg>
      <arg>
        <caserole>theme</caserole>
        <selrestrole>themenselres</selrestrole>
        <synset_list/>
      </arg>
      </args>
  </sem-caseframe>
  <sem-resume>beter maken</sem-resume>
</semantics_verb>
```

The correlations with synsets need to be created manually. They need to be compatible with the given semantic labels and with other role relations that are listed in DWN and the matched ontological process.


Concept | Relations | Tree | Revtree | Edit | Xml
Concept | Relations | Tree | Revtree | Edit | Xml
Item

```
C_SY_ID: d_n-24644
Synonyms: proef:3/r_n-29790

PoS specific: NOUN_DE
Differentiae: vaardigheid, kennis
SUMO: (=, , Experimenting) , (+, , Investigating) , (+, , ConstantQuantity) ,
(+, , Text)

-->> [HAS_HYPERONYM] test:2, experiment:1, onderzoeking:1,
probeersel:1, proef:3, proef:7, proefneming:1, toets:5, toetssteen:2/d_n-14431
<<-- [HAS_HYPONYM] auditie:1/n_n-502739
<<-- [HAS_HYPONYM] examen:1/d_n-39163
<<-- [HAS_HYPONYM] godsoordeel:1, godsgericht:1, ordale:1/d_n-39474
<<-- [HAS_HYPONYM] krachtproef:2/n_n-516010
<<-- [HAS_HYPONYM] ondervraging:2/n_n-522361
<<-- [HAS_HYPONYM] proefwerk:1, repetitie:3/d_n-30261
<<-- [HAS_HYPONYM] screentest:1/n_n-527436
<<-- [HAS_HYPONYM] toets:4, schooltoets:1/d_n-43954

WordNet domains: factotum, medicine, computer_science, school
```

```
⊟ iets:2, noun C=62/49946 D=
  ⊟ H gebeurtenis:1, gebeurde:1, geschiedenis:6, ontwikkeling:9, noun C...
    ⊟ H handeling:3, actie:3, verrichting:1, noun C=87/3874 D=false
      ⊞ H aanvaarding:1, noun C=1/1 D=
      ⊞ H aanvaarding:3, noun C=1/0 D=
      ⊞ H aanval:1, attaque:1, noun C=30/49 D=
      ⊟ H adoptie:1, noun C=1/1 D=
        ⊟ H stiefouderadoptie:1, noun C=0/0 D=
      ⊟ HH bedreiging:3, noun C=3/4 D=
        ⊟ H intimidatie:1, afschrikking:1, bangmakerij:1, noun C=1/1 D=
          ⊟ H chantage:1, afpersing:1, geldafpersing:1, knevelarij:1, n...
        ⊞ H doodsbedreiging:1, noun C=0/0 D=
        ⊞ H terrorisatie:1, noun C=0/0 D=
      ⊞ H besteding:1, noun C=4/4 D=
      ⊟ H bijdrage:4, aandeel:1, inbreng:4, noun C=2/40 D=
        ⊞ HH partij:10, noun C=13/23 D=
        ⊟ H rol:5, noun C=13/15 D=
          ⊞ H bijrol:1, noun C=1/1 D=
          ⊞ H dubbelrol:1, noun C=0/0 D=
          ⊞ H filmrol:1, noun C=0/0 D=
          ⊞ H gastrol:1, noun C=0/0 D=
```

Fig. 3. Cornetto Synsets window, showing a preview and a hyperonymy tree

3 The DEB Platform

The Dictionary Editor and Browser platform [12,1] offers a development framework for any dictionary writing system application that needs to store the dictionary entries in the XML format structures. The most important property of the system is the *client-server* nature of all DEB applications. This provides the ability of distributed authoring teams to work fluently on one common data source. The actual development of applications within the DEB platform can be divided into the server part (the server side functionality) and the client part (graphical interfaces with only basic functionality). The server part is built from small parts, called *servlets*, which allow a modular composition of all services. The client applications communicate with servlets using the standard HTTP web protocol.

For the server data storage the current database backend is provided by the Berkeley DB XML [13], which is an open source native XML database providing XPath and XQuery access into a set of document containers.

The user interface, that forms the most important part of a client application, usually consists of a set of flexible forms that dynamically cooperate with the server parts. According to this requirement, DEB has adopted the concepts of the Mozilla Development Platform [14]. Firefox Web browser is one of the many applications created using this platform. The Mozilla Cross Platform Engine provides a clear separation between application logic and definition, presentation and language-specific texts.

3.1 New DEB Features for the Cornetto Project

During the Cornetto project the nature of the Cornetto database structure has imposed the need of several features that were not present in the (still developing) DEB platform. The main new functionalities include:

– *entry locking* for concurrent editing. Editing of entries by distant users was already possible in DEB, however, the exclusivity in writing to the same dictionary item was not controlled by the server. The new functions offer the entry locking per user (called from the client application e.g. when entering the edit form). The list of all server locks is presented in the DEB administration interface allowing to handle the locks either manually or automatically on special events (logout, timeout, loading new entry, . . .).

– *link display preview caching.* According to the database design that (correctly) handles all references with entity IDs, each operation, like structure entry preview or edit form display, runs possibly huge numbers (tens or hundreds) of extra database queries displaying text representations instead of the entity ID numbers. The drawback of this compact database model is in slowing down the query response time to seconds for one entry. To overcome this increase of the number of link queries, we have introduced the concept of *preview caching.* With this mechanism the server computes all kinds of previews in the time of saving a modified entry in special entry variables (either XML subtags or XML metadata). In the time of constructing the preview or edit form, the linked textual representations are taken from the preview caches instead of running extra queries to obtain the computed values.
– *edit form functionalities* – the lexicographic experts within the Cornetto project have suggested several new user interface functions that are profitable for other DEB-based projects like collapsing of parts of the edit form, entry merging and splitting functions or new kinds of automatic inter-dictionary queries, so called AutoLookUps.

All this added functionalities are directly applicable in any DEB application like DEBVisDic or DEBDict.

4 The New DEBVisDic Clients

Since one of the basic parts of the Cornetto database is the Dutch WordNet, we have decided to use DEBVisDic as the core for Cornetto client software. We have developed four new modules, described in more details below. All the databases are linked together and also to external resources (Princeton English WordNet and SUMO ontology), thus every possible user action had to be very carefully analyzed and described.

During the several months of active development and extensive communication between Brno and Amsterdam, a lot of new features emerged in both server and client and many of these innovations were also introduced into the DEBVisDic software. This way, each user of this WordNet editor benefits from Cornetto project.

The user interface is the same as for all the DEBVisDic modules: upper part of the window is occupied by the query input line and the query result list and the lower part contains several tabs with different views of the selected entry. Searching for entries supports several query types – a basic one is to search for a word or its part, the result list may be limited by adding an exact sense number. For more complex queries users may search for any value of any XML element or attribute, even with a value taken from other dictionaries (the latter is used mainly by the software itself for automatic lookup queries).

The tabs in the lower part of the window are defined per dictionary type, but each dictionary contains at least a preview of an entry and a display of the entry XML structure. The entry preview is generated using XSLT templates, so it is very flexible and offers plenty of possibilities for entry representation.

4.1 Cornetto Lexical Units

The Cornetto foundation is formed by Lexical Units, so let us describe their client package first. Each entry contains complex information about morphology, syntax, semantics and pragmatics, and also lots of examples with complex substructure. Thus one of the important tasks was to design a preview to display everything needed by the lexicographers without the necessity to scroll a lot. The examples were moved to separate tab and only their short resumé stayed on the main preview tab.

Lexical units also contain semantic information from RBN that cannot be published freely because of licensing issues. Thus DEBVisDic here needs to differentiate the preview content based on the actual user's access rights.

The same ergonomic problem had to be resolved in the edit form. The whole form is divided to smaller groups of related fields (e.g. morphology) and it is possible to hide or display each group separately. By default, only the most important parts are displayed and the rest is hidden.

Another new feature developed for Cornetto is the option to split the edited entry. Basically, this function copies all content of edited entry to a new one. This way, users may easily create two lexical units that differ only in some selected details.

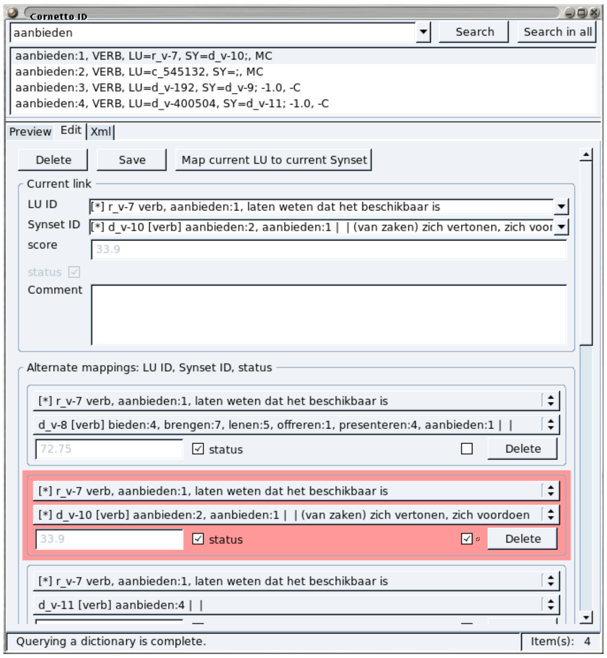
Because of the links between all the data collections, every change in lexical units has to be propagated to Cornetto Synsets and Identifiers. For example, when deleting a lexical unit, the corresponding synonym has to be deleted from the synset dictionary.

4.2 Cornetto Synsets

Synsets are even more complex than lexical units, because they contain lots of links to different sources – links to lexical units, relations to other synsets, equivalence links to Princeton English WordNet, and links to the ontology.

Again, designing the user-friendly preview containing all the information was very important. Even here, we had to split the preview to two tabs – the first with the synonyms, domains, ontology, definition and short representation of internal relations, and the second with full information on each relation (both internal and external to English Wordnet). Each link in the preview is clickable and displays the selected entry in the corresponding dictionary window (for example, clicking on a synonym opens a lexical unit preview in the lexical unit window).

The synset window offers also a tree view representing a hypernym/hyponym tree. Since the hypero/hyponymic hierarchy in Wordnet forms not a simple tree but a directed graph, another tab provides the reversed tree displaying links


Cornetto ID

aanbieden | Search | Search in all

aanbieden:1, VERB, LU=r_v-7, SY=d_v-10;, MC
aanbieden:2, VERB, LU=c_545132, SY=;, MC
aanbieden:3, VERB, LU=d_v-192, SY=d_v-9; -1.0, -C
aanbieden:4, VERB, LU=d_v-400504, SY=d_v-11; -1.0, -C

Preview | Edit | Xml

Delete | Save | Map current LU to current Synset

Current link
LU ID [*] r_v-7 verb, aanbieden:1, laten weten dat het beschikbaar is
Synset ID [*] d_v-10 [verb] aanbieden:2, aanbieden:1 | | (van zaken) zich vertonen, zich voor
score 33.9
status ☑
Comment

Alternate mappings: LU ID, Synset ID, status

[*] r_v-7 verb, aanbieden:1, laten weten dat het beschikbaar is
d_v-8 [verb] bieden:4, brengen:7, lenen:5, offreren:1, presenteren:4, aanbieden:1 | |
72.75 ☑ status ☐ Delete

[*] r_v-7 verb, aanbieden:1, laten weten dat het beschikbaar is
[*] d_v-10 [verb] aanbieden:2, aanbieden:1 | | (van zaken) zich vertonen, zich voordoen
33.9 ☑ status ☑ Delete

[*] r_v-7 verb, aanbieden:1, laten weten dat het beschikbaar is
d_v-11 [verb] aanbieden:4 | |

Querying a dictionary is complete. | Item(s): 4

Fig. 4. Cornetto Identifiers window, showing the edit form with several alternate mappings

in the opposite direction (this concept was introduced in the VisDic Wordnet editor). The tree view also contains information about each subtree's significance – like the number of direct hyponyms or the number of all the descendant synsets.

The synset edit form looks similar to the form in the lexical units window, with less important parts hidden by default. When adding or editing links, users may use the same queries as in dictionaries to find the right entry.

4.3 Cornetto Identifiers

The lexical units and synsets are linked together using the Cornetto Identifiers (CID). For each lexical unit, the automatic aligning software produced several mappings to different synsets (with different score values). At the very beginning, the most probable one was marked as the "selected" mapping.

In the course of work, users have several ways for confirming the automatic choice, choosing from other offered mapping, or creating an entirely new link. For example, a user can remove the incorrect synonym from a synset and the corresponding mapping will be marked as unselected in CID. Another option is to select one of the alternate mappings in the Cornetto Identifiers edit form. Of course, this action leads to an automatic update of synonyms.

The most convenient way to confirm or create links is to use *Map current LU to current Synset* function. This action can be run from any Cornetto client package, either by a keyboard shortcut or by clicking on the button. All the required changes are checked and carried out on the server, so the client software does not need to worry about the actual actions necessary to link the lexical unit and the synset.

4.4 Cornetto Ontology

The Cornetto Ontology is based on SUMO and so is the client package. The ontology is used in synsets, as can be seen in the Figure 3. The synset preview shows a list of ontology relations triplets – relation type, variable and variable or ontology term.

Clicking on the ontology term opens the term preview. A user can also browse the tree representing the ontology structure.

5 Conclusions

In the paper, we have described the Cornetto project workflow using the new lexicographic tools developed for this project. We have presented how a combination of automatic scored strategies with the human lexicographic work can be used for merging large databases of previous dictionaries to obtain a new qualitative language resource with complex morphological, syntactic and semantic information.

The presented project tools are, however, not a single purpose programs but they fit in the general framework of the Dictionary Editor and Browser (DEB) platform used for developing other publicly available language data tools.

Acknowledgments

The Cornetto project is funded by the Nederlandse Taalunie and STEVIN. This work has also partly been supported by the Ministry of Education of the Czech Republic within the Center of basic research LC536 and in the Czech National Research Programme II project 2C06009.

References

1. Horák, A., et al.: First version of new client-server wordnet browsing and editing tool. In: Proceedings of the Third International WordNet Conference - GWC 2006, Jeju, South Korea, Masaryk University, Brno, pp. 325–328 (2006)
2. Fillmore, C., Baker, C., Sato, H.: Framenet as a 'net'. In: Proceedings of Language Resources and Evaluation Conference (LREC 2004), Lisbon, ELRA, vol. 4, pp. 1091–1094 (2004)
3. Vossen, P. (ed.): EuroWordNet: a multilingual database with lexical semantic networks for European Languages. Kluwer Academic Publishers, Dordrecht (1998)
4. Maks, I., Martin, W., de Meerseman, H.: RBN Manual (1999)

5. Niles, I., Pease, A.: Linking lexicons and ontologies: Mapping WordNet to the suggested upper merged ontology. In: Proceedings of the IEEE International Conference on Information and Knowledge Engineering, pp. 412–416 (2003)
6. Cruse, D.: Lexical semantics. University Press, Cambridge (1986)
7. Miller, G., Fellbaum, C.: Semantic networks of english. Cognition (October 1991)
8. Daudé, J., Padró, L., Rigau, G.: Wordnet mappings, the Meaning project (2007), `http://www.upc.es/~nlp/tools/mapping.html`
9. Daudé, J., Padró, L., Rigau, G.: Validation and tuning of wordnet mapping techniques. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2003), Borovets, Bulgaria (2003)
10. Vossen, P.: The Cornetto project web site (2007), `http://www.let.vu.nl/onderzoek/projectsites/cornetto/start.htm`
11. Genesereth, M.R., Fikes, R.E. (eds.): Knowledge Interchange Format, Version 3.0, Reference Manual. Stanford University, Stanford CA, USA (1992), `http://www-ksl.stanford.edu/knowledge-sharing/kif/`
12. Horák, A., et al.: New clients for dictionary writing on the DEB platform. In: DWS 2006: Proceedings of the Fourth International Workshop on Dictionary Writings Systems, Italy, pp. 17–23. Lexical Computing Ltd, U.K (2006)
13. Chaudhri, A.B., Rashid, A., Zicari, R. (eds.): XML Data Management: Native XML and XML-Enabled Database Systems. Addison-Wesley, Reading (2003)
14. Feldt, K.: Programming Firefox: Building Rich Internet Applications with Xul. O'Reilly (2007)

Verb Class Discovery from Rich Syntactic Data

Lin Sun[1], Anna Korhonen[1], and Yuval Krymolowski[2]

[1] Computer Laboratory, University of Cambridge
15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
alk23@cam.ac.uk,ls418@cam.ac.uk
[2] Department of Computer Science, University of Haifa
31905, Haifa, Israel
yuvalkry@gmail.com

Abstract. Previous research has shown that syntactic features are the most informative features in automatic verb classification. We investigate their optimal characteristics by comparing a range of feature sets extracted from data where the proportion of verbal arguments and adjuncts is controlled. The data are obtained from different versions of VALEX [1] – a large SCF lexicon for English which was acquired automatically from several corpora and the Web. We evaluate the feature sets thoroughly using four supervised classifiers and one unsupervised method. The best performing feature set includes rich syntactic information about both arguments and adjuncts of verbs. When combined with our best performing classifier (a novel Gaussian classifier), it yields the promising accuracy of 64.2% in classifying 204 verbs to 17 Levin (1993) classes. We discuss the impact of our results on the state-or-art and propose avenues for future work.

1 Introduction

Recent research shows that it is possible, using current natural language processing (NLP) and machine learning technology, to automatically induce lexical classes from corpus data with promising accuracy [2,3,4,5]. This research is interesting, since lexical classifications, when tailored to the application and domain in question, can provide an effective means to deal with a number of important NLP tasks (e.g. parsing, word sense disambiguation, semantic role labeling), as well as enhance performance in applications, (e.g. information extraction, question-answering, machine translation) [6,7,8,9,10].

Lexical classes are useful for NLP because they capture generalizations over a range of (cross-)linguistic properties. Being defined in terms of similar meaning components and (morpho-)syntactic behaviour of words [11,12] they generally incorporate a wider range of properties than e.g. classes defined solely on semantic grounds [13]. For example, verbs which share the meaning component of ‘manner of motion’ (such as *travel*, *run*, *walk*), behave similarly also in terms of subcategorization (*I traveled/ran/walked to London*) and usually have zero-related nominals (*a run*, *a walk*).

NLP systems can benefit from lexical classes in many ways. For example, such classes can be used i) to define a mapping from surface realization of arguments to predicate-argument structure, ii) as a means to abstract away from individual words when required, or (iii) to build a lexical organization which predicts much of the syntax and semantics of a new word by associating it with an appropriate class.

While lexical classes have proved useful for various (multilingual) tasks, their large-scale exploitation in real-world or domain-sensitive tasks has not been possible because existing manually built classifications are incomprehensive. They are expensive to extend and do not incorporate important statistical information about the likelihood of different classes for words. Automatic classification can help since it is cost-effective and gathers statistical information as a side-effect of the acquisition process.

Most work on lexical classification has focussed on verbs, which are typically the main predicates in sentences. Syntactic features have proved the most informative for verb classification. The best results in automatic classification have been obtained using either (i) deep syntactic features (e.g. subcategorization frames (SCFs)) extracted using parsers and subcategorisation acquisition systems [14,3,4] or (ii) shallow ones (e.g. NPs/PPs preceding/following verbs) extracted using taggers and chunkers [2,5]. (i) capture the arguments of verbs and correspond closely with the features used for manual classification [12]. The fact that promising results have also been reported using (ii) has led to suggestions that adjuncts can also be useful for the task, e.g. [5].

To gain a better understanding of the optimal characteristics of syntactic features in verb classification, we compare a range of feature sets extracted from data where the proportion of verbal arguments and adjuncts is controlled. The data are obtained from different versions of VALEX [1] – a large SCF lexicon for English which was acquired automatically from several corpora and the Web. We evaluate the feature sets thoroughly using four supervised classifiers and one unsupervised method. The best performing feature set is the one which includes the richest syntactic information about both arguments and adjuncts of verbs. When combined with the best performing classifier (our novel Gaussian classifier), it yields the promising accuracy of 64.2% in classifying 204 verbs to 17 Levin (1993) classes. We discuss the impact of our results on the state-or-art and propose avenues for future work.

We introduce our target classification in section 2 and syntactic features in section 3. The classification and clustering techniques are presented in section 4. Details of the experimental evaluation are supplied in section 5. Section 6 provides discussion and concludes with directions for future work.

2 Test Verbs and Classes

We adopt as a target classification Levin's (1993) well-known taxonomy where verbs taking similar diathesis alternations are assumed to share meaning components and are organized into a semantically coherent class. For instance, the class of "*Break* Verbs" (class 45.1) is partially characterized by its participation in the following alternations:

1. **Causative/inchoative alternation**:
   Tony broke the window ↔ *The window broke*
2. **Middle alternation**:
   Tony broke the window ↔ *The window broke easily*
3. **Instrument subject alternation**:
   Tony broke the window with the hammer ↔ *The hammer broke the window*

Table 1. Test classes and example verbs

	LEVIN CLASS	EXAMPLE VERBS
9.1	PUT	*bury, place, install, mount, put, deposit, position, set*
10.1	REMOVE	*remove, abolish, eject, extract, deduct, eradicate, sever, evict*
11.1	SEND	*ship, post, send, mail, transmit, transfer, deliver, slip*
13.5.1	GET	*win, gain, earn, buy, get, book, reserve, fetch*
18.1	HIT	*beat, slap, bang, knock, pound, batter, hammer, lash*
22.2	AMALGAMATE	*contrast, match, overlap, unite, unify, unite, contrast, affiliate*
29.2	CHARACTERIZE	*envisage, portray, regard, treat, enlist, define, depict, diagnose*
30.3	PEER	*listen, stare, look, glance, gaze, peer, peek, squint*
31.1	AMUSE	*delight, scare, shock, confuse, upset, overwhelm, scare, disappoint*
36.1	CORRESPOND	*cooperate, collide, concur, mate, flirt, interact, dissent, mate*
37.3	MANNER OF SPEAKING	*shout, yell, moan, mutter, murmur, snarl, moan, wail*
37.7	SAY	*say, reply, mention, state, report, respond, announce, recount*
40.2	NONVERBAL EXPRESSION	*smile, laugh, grin, sigh, gas, chuckle, frown, giggle*
43.1	LIGHT EMISSION	*shine, flash, flare, glow, blaze, flicker, gleam, sparkle*
45.4	CHANGE OF STATE	*soften, weaken, melt, narrow, deepen, dampen, melt, multiply*
47.3	MODES OF BEING WITH MOTION	*quake, falter, sway, swirl, teeter, flutter, wobble, waft*
51.3.2	RUN	*swim, fly, walk, slide, run, travel, stroll, glide*

Alternations are expressed as pairs of SCFs. Additional properties related to syntax, morphology and extended meanings of member verbs are specified with some classes. The extended version of Levin's classification currently incorporated in VerbNet [15][1] provides a classification of 5,257 verb senses into 274 classes. We selected 17 of Levin's original classes and 12 member verbs per class (table 1) for experimentation. The small test set enabled us to evaluate our results thoroughly. The classes were selected in random, subject to the constraint that they (i) included both syntactically and semantically similar and different classes (to vary the difficulty of the classification task), and (ii) had enough member verbs whose predominant sense belongs to the class in question (we verified this according to the method described in [1]). As VALEX was designed for a maximum coverage most test verbs had 1000-9000 occurrences in the lexicon.

3 Syntactic Features

We employed as features distributions of SCFs specific to given verbs. We extracted them from the recent large VALEX [1] lexicon which provides SCF frequency information for 6,397 English verbs. VALEX was acquired automatically from five large corpora and the Web (up to 10,000 occurrences per verb) using the subcategorization acquisition system of Briscoe and Carroll [16]. The system incorporates RASP, a domain-independent robust statistical parser [17], and a SCF classifier which identifies 163 verbal SCFs. The basic SCFs abstract over lexically-governed particles and prepositions and predicate selectional preferences. Three versions of VALEX were employed[2]:

Valex 1: A noisy unfiltered version of VALEX which includes all the SCFs found in data.
Valex 2: A sub-lexicon created by selecting from Valex 1 only SCFs whose relative frequency is higher than a SCF-specific threshold.

[1] See http://verbs.colorado.edu/verb-index/index.php for details.
[2] See [1] for the full description of these versions of VALEX and the details of their evaluation.

Valex 3: A sub-lexicon created by selecting from Valex 1 SCFs which are also listed in the manually built ANLT [18] and the COMLEX syntax [19] dictionaries, and those whose relative frequency is higher than a SCF-specific threshold.

For our 204 test verbs, these three lexicons include 44, 5, and 5 SCFs per verb on average. According to the evaluation reported in [1], the SCF accuracy of Valex 1, 2, and 3 is 21.9, 58.6 and 83.7 according to F-measure, respectively, when evaluated on a set of 183 test verbs. Although standard text processing and parser errors result in some noise, the main source of error in SCF acquisition is the difficulty of argument-adjunct distinction. Therefore the higher the SCF accuracy of a lexicon, the higher the number of genuine arguments in the SCFs (e.g. *I sang* ***a song*** correctly analysed as SCF NP), as opposed to adjuncts (e.g. *I sang* ***in the party*** incorrectly analysed as SCF PP), and vice versa. Thus by controlling the SCF accuracy we can control the proportion of arguments and adjuncts in input data and examine the effects on classification.

A lexical entry for each verb and SCF combination provides e.g. the frequency of the entry in corpora, the POS tags of verb tokens, the argument heads in argument positions, and the prepositions in PP slots. We experimented with five feature sets:

Feature set 1: SCFs and their frequencies

Feature set 2: Feature set 1 with two high frequency PP frames parameterized for prepositions: the simple PP (e.g. *they apologized to him*) and NP-PP (e.g. *he removed the shoes from the bag*) frames refined according to the prepositions provided in the VALEX SCF entries (e.g. PP_*at*, PP_*on*, PP_*in*).

Feature sets 3-5: Feature set 2 with additional 3, 8 and 13 high frequency PP frames parameterized for prepositions, respectively.

Although prepositions are an important part of the syntactic description of Levin style classes and therefore feature set 5 should be the most informative one (and feature set 1 the least informative one), we controlled the number of PP frames parameterized for prepositions in order to examine the effects of sparse data in automatic classification.

4 Classification

4.1 Preparing the Data

A feature vector was constructed for each verb. For example, Valex 1 includes 107, 287 and 305 SCF types for feature sets 1, 2, and 3, respectively. Each feature corresponds to a SCF type. Its value is the relative frequency of the SCF with the verb in question. Some of the feature values are zero because most verbs take only a subset of the possible SCFs.

4.2 Machine Learning Methods

We experimented with five methods for classification: four supervised ones (K nearest neighbours, support vector machines, maximum entropy, Gaussian) and one unsupervised one (cost-based pairwise clustering). To our knowledge, two of these methods (the Gaussian and clustering methods) have not been previously used for verb classification. The free parameters of classifiers were optimised for each feature set by (i) defining the value range (as explained in below sections), and (ii) searching for the optimal value on the training data using 10 fold cross validation (section 5.2).

K Nearest Neighbours. K Nearest Neighbours (KNN) is a memory-based classification method based on the distances between verbs in the feature space. For each verb in the test data, we measure its distance from each verb in the training data. We then assign it the label which is the most frequent among the top K closest training verbs. We use the entropy-based Jensen-Shannon (JS) divergence as the distance measure:

$$JS(P,Q) = \frac{1}{2}\left[D(P\|\tfrac{P+Q}{2}) + D(Q\|\tfrac{P+Q}{2})\right]$$

The range of the parameter K is 2-20.

Support Vector Machines. The Support Vector Machines (SVM) [20] try to find a maximal margin hyperplane to separate between two groups of verb feature vectors. In practice, a linear separator is unlikely to exist in the original feature space. SVM uses a kernel function to map the original feature vectors to a higher dimension space. The 'maximal margin' optimizes our choice of dimensionality to avoid over-fitting. We used Chang and Lin's LIBSVM library [21] to implement the SVM. Following [22], we use the radial basis function as the kernel function:

$$K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2), \gamma > 0$$

γ and the cost of the error term C (the penalty for margin errors) are optimized. The search ranges of [22] are used:

$$C = 2^{-5}, 2^{-3}, \ldots, 2^{15}, 2^{17}\ ; \gamma = 2^{-17}, 2^{-15}, \ldots, 2^{1}, 2^{3}$$

Maximum Entropy. Maximum entropy (ME) constructs a probabilistic model that maximizes entropy on test data subject to a set of feature constraints. If verb x is in class 10.1 and takes the SCF 49 (NP-PP) with the relative frequency of 0.6 in feature function f, we have:

$$f(x,y) = 0.6 \text{ if } y = 10.1 \text{ and } x = 49$$

The expected value of a feature f with respect to the empirical distribution (training data) is:

$$\tilde{\mathcal{E}}(f) \equiv \textstyle\sum_{x,y} \tilde{p}(x,y) f(x,y)$$

The expected value of the feature f (on test data) with respect to the model $p(y|x)$ is:

$$\mathcal{E}(f) \equiv \textstyle\sum_{x,y} \tilde{p}(x) p(y|x) f(x,y)$$

$\tilde{p}(x)$ is the empirical distribution of x in the training data. We constrain $\mathcal{E}(f)$ to be the same as $\tilde{\mathcal{E}}(f)$:

$$\mathcal{E}(f) = \tilde{\mathcal{E}}(f)$$

The model must maximize the entropy $H(Y|X)$:

$$H(Y|X) \equiv -\textstyle\sum_{x,y} \tilde{p}(x) p(y|x) \log p(y|x)$$

The constraint-optimization problem is solved by the Lagrange multiplier [23]. We used Zhang's [24] maximum entropy toolkit for implementation. The number of iterations i (5-50) of the parameter estimation algorithm is optimised.

Gaussian. The Gaussian model assumes that the SCF frequencies of verbs in a class are normally distributed with averages and standard deviations characteristic of the class. We assume the dimensions of a feature vector to be independent of each other. The covariance matrix of the statistical distribution of features is diagonal. Suppose x is a feature vector of a verb, which has d dimensions, and y is the mean vector of the verb class which has N verbs with feature vectors $x_1 \dots x_n$. The likelihood is:

$$p(x|y) = \Pi_{a=1}^{d} \frac{1}{\sqrt{2\pi\sigma_a^2}} \exp\left(-\frac{(x_i-\mu_a)^2}{2\sigma_a^2}\right)$$

The mean and variance of a dimension a are:

$$\mu_a = \frac{\sum_{i=1}^{n} x_i}{N}, \sigma_a^2 = \frac{\sum_{i=1}^{n}(x_i-\mu_i)^2}{N}$$

To predict the class membership of a test verb, we calculate its likelihood for each class and choose the class with the highest likelihood.

Pairwise Clustering. We adopt a cost-based framework for pairwise clustering (PC) [25] where a cost criterion guides the search for a suitable clustering configuration. This criterion is realized through a cost function $H(S, M)$ where

(i) $S = \{\mathrm{sim}(a, b)\}$, $a, b \in A$: a collection of pairwise similarity values, each of which pertains to a pair of data elements $a, b \in A$.
(ii) $M = (A_1, \dots, A_k)$: a candidate clustering configuration, specifying assignments of all elements into the disjoint clusters (that is $\cup A_j = A$ and $A_j \cap A_{j'} = \phi$ for every $1 \le j < j' \le k$).

The cost function is defined as follows:

$$H = -\sum n_j \cdot \mathrm{Avgsim}_j ,$$
$$\mathrm{Avgsim}_j = \frac{1}{n_j \cdot (n_j-1)} \sum_{\{a,b \in A_j\}} \mathrm{sim}(a, b)$$

where n_j is the size of the j^{th} cluster and Avgsim_j is the average similarity between cluster members. The similarity is measured by the JS divergence.

The number of clusters, k, is specified as an input parameter. We varied the value of k from 10 to 35. The best performance was obtained for k values close to the number of gold standard classes ($n = 17$). We report the results for $k = 17$.

5 Experiments

5.1 Methodology for Supervised Methods

We split the data into training and test sets using two methods. The first is 'leave one out' *cross-validation* where one verb is held out as test data, and the remaining N-1 verbs are used as training data. The overall accuracy is the average accuracy of N rounds. The second method is *re-sampling*. For each class, 3 verbs are selected randomly as test data and 9 are used as training data. The process is repeated 30 times and the average result is recorded.

5.2 Measures

Supervised methods are evaluated using accuracy – the percentage of correct classifications out of all the classifications:

$$Accuracy = \frac{truePositives}{\text{Number of verbs}}$$

When evaluating the performance at class level, precision and recall are calculated as follows:

$$Precision = \frac{truePositives}{truePositives + falsePositives}$$
$$Recall = \frac{truePositives}{truePositives + falseNegatives}$$

F-measure is the balance over recall and precision. We report the average F-measure over the 17 classes. Given there are 17 classes in the data, the accuracy of randomly assigning a verb into one of the 17 classes is $1/17 \approx 5.8\%$.

Clustering is evaluated using *unique purity* (uPUR) which evaluates the output as if it were the output of a classifier. For each cluster, the *dominant class* is the class with most cluster members. If a class is dominant in several clusters we choose the cluster with the highest number of class members. This is analogous to the output of a classifier where only one output class can correspond to an actual class. uPUR is the total number of these verbs divided by the number of verbs N. The experiments were run 50 times on each input to get the distribution of performance due to the randomness in the initial clustering.

5.3 Results from Quantitative Evaluation

Table 2 shows the average performance of each classifier and feature set according to 'leave one out' cross-validation when the features are extracted from Valex 1. Each classifier performs considerably better than the random baseline. The simple KNN method produces the lowest accuracy (44.1-54.9). SVM and ME perform better (47.1-57.8 and 47.5-59.8 accuracy, respectively), with GS yielding the best accuracy (49.5-64.2). The clustering method performs similarly with KNN, yielding uPUR of 39.6-51.6.

The performance of all methods improves sharply when moving from feature set 1 to the refined feature set 2: both accuracy and F-measure improve c. 10%. When moving further to feature set 3 (which includes a higher number of low frequency PP features) KNN worsens clearly (c. 5% in accuracy and F-measure) while the other methods perform similarly. With sparser feature sets 4-5, KNN, PC SVM and ME show similar performance than with feature set 3 (the changes in SVM and ME are not statistically significant). GS is the only method which performs the best with the most refined feature set 5 (64.2 accuracy and 62.5 F-measure). ME produces the same results as with feature set 4: 59.8 accuracy and 59.9 F-measure. The differences in accuracy and F-measure of GS and ME are significant at the 3.4σ and 2σ level, respectively.

The resampling results in table 3 reveal that some classifiers perform worse than others when less training data is available[3]. KNN produces considerably lower results with resampling, particularly with the sparse feature set 5: 20.3 F-measure vs. 48.5 with

[3] Recall that the amount of training data is smaller with resampling evaluation, see section 5.2.

Table 2. 'Leave one out' cross-validation results for supervised methods; PC results with $\mathcal{K} = 17$ clusters. The standard deviation of ACC and F is $\sigma = 0.9$. Feature sets extracted from Valex 1.

	Feature set 1		Feature set 2		Feature set 3		Feature set 4		Feature set 5	
	ACC	F	ACC	F	ACC	F	ACC	F	ACC	F
RAND	5.8		5.8		5.8		5.8		5.8	
KNN	44.1	44.0	54.9	53.9	49.5	48.2	49.5	48.3	49.5	48.5
ME	47.5	47.6	59.3	59.9	59.3	60.0	59.8	59.9	59.8	59.9
SVM	47.1	47.8	57.8	57.9	57.8	58.2	57.3	57.5	57.3	57.4
GS	49.5	46.2	59.3	57.1	59.3	56.5	59.8	56.6	64.2	62.5
	uPUR		uPUR		uPUR		uPUR		uPUR	
PC	39.6% ± 1.5		51.4% ± 2.5		51.5% ± 2.6		51.6% ± 2.5		51.2% ± 3.7	

Table 3. Re-sampling results for supervised methods; PC results with $\mathcal{K} = 17$ clusters. The σ line presents the average standard deviation for each measure. Feature sets extracted from Valex 1.

	Feature set 1		Feature set 2		Feature set 3		Feature set 4		Feature set 5	
	ACC	F	ACC	F	ACC	F	ACC	F	ACC	F
RAND	5.8		5.8		5.8		5.8		5.8	
KNN	37.3	36.5	42.7	42.6	27.1	28.2	23.6	24.0	20.5	20.3
ME	47.1	47.0	58.1	58.1	60.1	59.8	59.8	59.6	57.5	57.8
SVM	47.3	47.7	56.8	57.1	54.4	54.6	56.8	56.9	55.6	55.5
GS	39.0	40.0	49.9	51.4	50.0	51.7	52.4	53.2	54.0	55.5
σ	6.0	6.1	6.5	6.8	6.1	6.2	5.7	5.8	5.8	5.9
	uPUR		uPUR		uPUR		uPUR		uPUR	
PC	39.6% ± 1.5		51.4% ± 2.5		51.5% ± 2.6		51.6% ± 2.5		51.2% ± 3.7	

Table 4. 'Leave one out' cross-validation results with the three versions of Valex (feature set 5). The standard deviation of ACC, P, and F is $\sigma = 0.9$.

	Valex 1			Valex 2			Valex 3		
	ACC	P	F	ACC	P	F	ACC	P	F
RAND	5.8			5.8			5.8		
KNN	49.5	47.5	48.5	46.6	42.4	44.4	47.1	44.3	45.7
ME	59.8	59.7	59.9	53.9	54.9	54.4	59.3	60.1	59.7
SVM	57.3	57.5	57.4	55.8	55.2	55.5	53.9	53.5	53.7
GS	64.2	60.8	62.5	53.9	46.8	50.1	56.3	51.0	53.5
	uPUR			uPUR			uPUR		
PC	51.2% ± 3.7			44.9% ± 1.5			48.3% ± 1.0		

cross-validation. Also other methods perform worse. Although a considerable improvement can be seen in GS with the use of more sophisticated feature sets, ME produces the best results with resampling.

These results were obtained using features extracted from Valex 1. Table 4 shows 'leave one out' cross-validation results for all the three versions of Valex with feature set 5. Each method performs the best with the very noisy Valex 1 which includes a lot of adjunct data in addition to argument data. Most methods perform the second best with Valex 3 which includes highly accurate argument data supplemented with high frequency adjunct data. In these results with feature set 5 we can see the biggest difference in GS which yields 62.5 F-measure with Valex 1, 53.5 with Valex 3 and 50.1 with Valex 2. For other methods big differences can be detected between the three lexicons when using less ambitious feature sets 1-3.[4]

5.4 Qualitative Evaluation

Figure 1 shows the F-measure for 17 individual classes when supervised methods are used with feature set 5 extracted from Valex 1. Levin classes 40.2, 29.2, and 37.3 (see table 1) which take fewer prepositions with higher frequency have the best average performance (65% or more) among all the methods, and classes 47.3, 45.4 and 18.1 the worst (40% or less). GS outperforms other methods with 11 of the 17 classes.

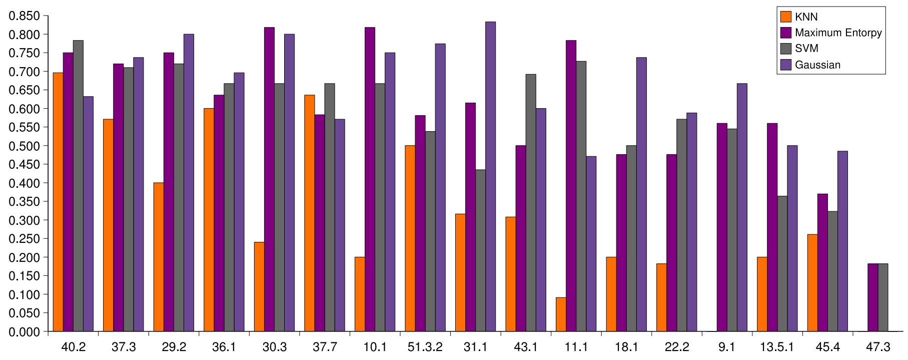


Fig. 1. Class level F-score for feature set 5 (cross-validation)

	10.1	11.1	13.5.1	18.1	22.2	29.2	30.3	31.1	36.1	37.3	37.7	40.2	43.1	45.4	47.3	51.3.2	9.1
10.1	-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11.1	0	-	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
13.5.1	1	1	-	2	0	1	0	0	0	0	0	0	0	1	0	0	0
18.1	0	0	1	-	0	0	1	0	0	0	0	0	0	1	0	0	1
22.2	0	1	0	0	-	0	0	0	1	0	0	0	0	0	0	0	2
29.2	0	0	1	0	0	-	0	0	0	0	0	0	0	0	0	0	1
30.3	0	0	0	0	0	0	-	0	0	1	0	1	0	0	0	0	0
31.1	1	0	0	2	0	0	0	-	0	0	0	0	0	2	0	0	1
36.1	0	0	0	0	2	0	0	0	-	0	0	0	0	0	2	0	0
37.3	0	0	0	0	0	0	1	0	0	-	2	2	0	0	0	1	0
37.7	0	0	0	0	0	0	1	0	0	1	-	0	0	1	2	0	0
40.2	0	0	0	0	0	0	0	0	0	1	0	-	0	0	0	0	0
43.1	0	0	0	1	0	0	0	0	1	0	0	0	-	0	3	1	0
45.4	1	0	0	0	1	1	0	3	1	0	1	0	0	-	1	0	0
47.3	0	0	0	0	1	0	0	0	2	0	0	0	4	1	-	0	0
51.3.2	0	1	1	1	0	0	0	1	0	0	1	0	2	0	2	-	0
9.1	0	0	1	1	3	1	0	0	0	0	1	0	0	0	0	0	-

Fig. 2. Error matrix for ME, a column indicates a correct class and a row a mistakenly predicted class

[4] These figures are not show in table 4 due to space restrictions.

Figure 2 shows the error matrix for ME with feature set 5. Examination of the worst performing class 47.3 (MODES OF BEING INVOLVING MOTION verbs) illustrates well the various error types. For ME 10 of the 12 verbs in this class are classified incorrectly:

- **3** (*flutter, falter, teeter*) in class 43.1 (LIGHT EMISSION verbs): Verbs in 47.3 and 43.1 describe intrinsic properties of their subjects (e.g. *a jewel sparkles*, *a flag flutters*). Their similar alternations and PP SCFs make it difficult to separate them on syntactic grounds.
- **2** (*swirl, waft*) in class 51.3.2 (RUN verbs): 47.3 and 51.3.2 share the meaning component of motion. Their members take similar alternations and SCFs, which causes the confusion.
- **2** (*quiver, vibrate*) in class 36.1 (CORRESPOND verbs): 47.3 and 36.1 are semantically very different, but their members take similar intransitive and PP SCFs with high frequency.
- **2** (*quake, wiggle*) in class 37.7 (SAY verbs): 47.3 differs in semantics and syntax from 37.7. The confusion is due to idiosyncratic properties of individual verbs.
- **1** (*sway*) in class 45.4 (OTHER CHANGE OF STATE verbs): Classes 47.3 and 45.3 are semantically different. Their similar PP SCFs explains the misclassification.

Interestingly, the errors of GS with class 47.3 are very similar. Both methods confuse 47.3 with classes 51.3.2, 37.7 and 43.1, but GS assigns as many as 5 verbs to class 51.3.2. Most errors concern classes which are in fact semantically related. Unfortunately no existing gold standard comprehensively captures the semantic relatedness of Levin classes. This kind of error analysis could be used as a step towards creating one. Other errors concern semantically unrelated but syntactically similar classes – cases which we can try to address with careful feature engineering. Some errors relate to syntactic idiosyncracy. These show the true limits of lexical classification - the fact that the correspondence between the syntax and semantics of verbs is not always perfect.

6 Discussion and Conclusion

In our experiments, rich syntactic features incorporating extensive information about both arguments and adjuncts of verbs proved the most useful for verb classification. This result not only confirms the earlier observations that adjuncts can be useful for the task [4,5], but demonstrates that adjuncts are actually very important for the task. SCFs containing arguments have been used as primary features in manual verb classification (Levin, 2003) for good theoretical reasons. However, adjuncts may be absent in manual work for practical reasons. It may be that they are best identified via general statistics about verb usages in corpora (rather than via specific syntactic tests). This kind of statistical information would be prohibitively difficult to capture manually.

Our best performing method (the new GS method) produced the best results with the challenging feature set 5 which is the most informative feature set but challenging to most methods due to its sparseness. The 64.2 accuracy and 62.5 F-measure produced by GS is especially promising considering that in these experiments focussing on the basic characteristics of syntactic features we performed no sophisticated feature engineering or selection based on the properties of the target classification.

Due to differences in target languages and evaluation procedures, direct comparison of our results against previously published ones is difficult. The closest comparison point is the recent experiment reported in [5] which involved classifying 835 English verbs to 14 Levin classes using SVM. Features were specifically selected via analysis of alternations that are used to characterize Levin classes. Both (i) shallow syntactic features (syntactic slots obtained using a chunker) and (ii) deep ones (SCFs extracted using Briscoe and Carroll's system) were used. The accuracy was 58% with (i) and only 38% with (ii). This experiment is not directly comparable with ours as we classified a smaller number of verbs (204) to a higher number of Levin classes (17) (i.e. we had less training data) and did not select the optimal set of features using Levin's alternations. We nevertheless obtained better accuracy with our best performing method, and better accuracy (47%) with the same method (SVM) when the comparable feature set 1 was acquired using the very same subcategorization acquisition system. [5] does not reveal whether noise was filtered from the system output to obtain a set of SCFs containing mostly arguments. In the light of our experiments it seems likely that this was done, and that using much larger and noisier SCF data including a high proportion of adjuncts explains our better result.

Further experiments are required to determine the optimal set of features. The fact that we obtained the best results using noisy Valex 1 and the second best using highly accurate but filtered Valex 3 suggests that combining these two lexicons into a single lexicon may be the best approach. It would yield a large lexicon with good coverage in adjuncts and high accuracy in arguments. In the future, we also plan to improve the features by e.g. enriching them with additional syntactic information recoverable from the parser output and/or available in VALEX lexical entries (see section 3). Incorporating semantic information e.g. about selectional preferences [4] would also be interesting, although this has proved challenging in earlier works.

We evaluated our features using both supervised and unsupervised methods. This was important since the two types of methods can serve different purposes: unsupervised methods are ideal for class discovery (e.g. in new domains) while supervised methods are useful for supplementing existing classifications with additional members. However, supervised techniques may perform optimally when combined with unsupervised techniques and a large unlabelled data [26]. In the future, we will therefore investigate a semi-supervised approach to verb classification.

Acknowledgement

This work was supported by the EPSRC project 'ACLEX' and the Royal Society, UK.

References

1. Korhonen, A., Krymolowski, Y., Briscoe, T.: A large subcategorization lexicon for natural language processing applications. In: Proceedings of LREC (2006)
2. Merlo, P., Stevenson, S.: Automatic verb classification based on statistical distributions of argument structure. Computational Linguistics 27, 373–408 (2001)
3. Korhonen, A., Krymolowski, Y., Collier, N.: Automatic classification of verbs in biomedical texts. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual meeting of the ACL, pp. 345–352 (2006)
4. Schulte im Walde, S.: Experiments on the automatic induction of german semantic verb classes. Computational Linguistics 32, 159–194 (2006)

5. Joanis, E., Stevenson, S., James, D.: A general feature space for automatic verb classification. Natural Language Engineering (forthcoming, 2007)
6. Dorr, B.J.: Large-scale dictionary construction for foreign language tutoring and interlingual machine translation. Machine Translation 12, 271–322 (1997)
7. Prescher, D., Riezler, S., Rooth, M.: Using a probabilistic class-based lexicon for lexical ambiguity resolution. In: 18th International Conference on Computational Linguistics, Saarbrücken, Germany, pp. 649–655 (2000)
8. Swier, R., Stevenson, S.: Unsupervised semantic role labelling. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, pp. 95–102 (2004)
9. Dang, H.T.: Investigations into the Role of Lexical Semantics in Word Sense Disambiguation. PhD thesis, CIS, University of Pennsylvania (2004)
10. Shi, L., Mihalcea, R.: Putting pieces together: Combining FrameNet, VerbNet and WordNet for robust semantic parsing. In: Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico (2005)
11. Jackendoff, R.: Semantic Structures. MIT Press, Cambridge (1990)
12. Levin, B.: English Verb Classes and Alternations. Chicago University Press, Chicago (1993)
13. Miller, G.A.: WordNet: An on-line lexical database. International Journal of Lexicography 3, 235–312 (1990)
14. Schulte im Walde, S.: Clustering verbs semantically according to their alternation behaviour. In: Proceedings of COLING, Saarbrücken, Germany, pp. 747–753 (2000)
15. Kipper, K., Dang, H.T., Palmer, M.: Class-based construction of a verb lexicon. In: AAAI/IAAI, pp. 691–696 (2000)
16. Briscoe, E.J., Carroll, J.: Automatic extraction of subcategorization from corpora. In: Proceedings of the 5^{th} ACL Conference on Applied Natural Language Processing, Washington DC, pp. 356–363 (1997)
17. Briscoe, E.J., Carroll, J.: Robust accurate statistical annotation of general text. In: Proceedings of the 3^{rd} LREC, Las Palmas, Gran Canaria, pp. 1499–1504 (2002)
18. Boguraev, B., Briscoe, T.: Large lexicons for natural language processing: utilising the grammar coding system of ldoce. Comput. Linguist. 13, 203–218 (1987)
19. Grishman, R., Macleod, C., Meyers, A.: Comlex syntax: building a computational lexicon. In: Proceedings of the 15th conference on Computational linguistics, Morristown, NJ, USA, Association for Computational Linguistics, pp. 268–272 (1994)
20. Vapnik, V.N.: The nature of statistical learning theory. Springer, New York (1995)
21. Chang, C., Lin, J.: LIBSVM: a library for support vector machines (2001)
22. Hsu, W., Chang, C., Lin, J.: A practical guide to support vector classification (2003)
23. Pietra, S.D., Pietra, J.D., Lafferty, J.D.: Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence 19, 380–393 (1997)
24. Zhang, L.: Maximum Entropy Modeling Toolkit for Python and C++ (2004)
25. Puzicha, J., Hofmann, T., Buhmann, J.M.: A theory of proximity-based clustering: structure detection by optimization. Pattern Recognition 33, 617–634 (2000)
26. Ando, R.K., Zhang, T.: A high-performance semi-supervised learning method for text chunking. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 1–9 (2005)

Growing TreeLex

Anna Kupść[1] and Anne Abeillé[2]

[1] Université de Bordeaux, ERSSàB/SIGNES and IPIPAN;
Université Michel de Montaigne, Domaine Universitaire, UFRL
33607 Pessac Cedex, France
[2] Université Paris7, LLF/CNRS
UMR 7110, CNRS-Université Paris 7, Case 7031, 2, pl. Jussieu
75251 Paris Cedex 05, France
akupsc@u-bordeaux3.fr, anne.abeille@linguist.jussieu.fr

Abstract. TreeLex is a subcategorization lexicon of French, automatically extracted from a syntactically annotated corpus. The lexicon comprises 2006 verbs (25076 occurrences). The goal of the project is to obtain a list of subcategorization frames of contemporary French verbs and to estimate the number of different verb frames available in French in general. A few more frames are discovered when the corpus size changes, but the average number of frames per verb remains relatively stable (about 1.91–2.09 frames per verb).

Keywords: Verb valence, subcategorization, treebank.

1 Introduction

The paper presents TreeLex, a subcategorization lexicon for French, automatically extracted from a syntactically annotated corpus.

Information about the combinatory potential of a predicate, i.e., the number and the type of its arguments, is called a subcategorization frame or valence. For example, the verb *embrasser* 'kiss' requires two arguments (the subject and an object), both of them realized as a noun phrase, whereas the predicative adjective *fier* 'proud' selects a prepositional complement introduced by the preposition *de*. This kind of syntactic properties is individually associated with every predicate, both within a single language and cross-linguistically. For example, the English verb *miss* has two NP arguments but the second argument of its French equivalent *manquer* is a PP (and semantic roles of the two arguments are reversed). This implies that subcategorization lexicons which store such syntactic information have to be developed for each language individually.[1] In addition to their importance in language learning, they play a crucial role in many NLP applications related both to parsing, e.g., [4], [6], [24], and generation, e.g., [9], [17].

[1] Work on mapping theory has revealed partial correlations between lexical semantics and subcategorization frames, see for example [10] for linking relations of verbs' arguments. We are not aware of any similar work done for other types of predicates, e.g., adjectives or adverbs.

The (un)availability of such lexical resources is still a bottleneck for text processing. Traditionally, they have been developed manually by human experts, e.g., [21,18] (for English) or [15,16,19,25] (for French), which guarantees their high quality, but they cannot be directly used in NLP applications. With the development of corpora and adaptation of statistical techniques for NLP, more efficient methods became available, which allowed for an automatic construction of syntactic lexicons for many languages (English, Spanish, German, Chinese), cf. [5,13]. Recent years have witnessed also an increased interest in obtaining such resources for French, either by applying statistical techniques, e.g., [2], [7], adapting the existing lexicons, e.g., [14,11], or using heurisitics to extract valence information [23,22,8] for French verbs; a syntactic lexicon of French prepositions has been lately created by [12].

In this paper we present another effort on automatic extraction of a syntactic lexicon for French verbs. The approach we have adopted differs form those mentioned above as it relies on syntactic (and functional) corpus annotations. We use the treebank of Paris7, [1], a journalistic corpus based on articles from *Le monde* (1989–1993), a French daily newspaper. The corpus contains morphological, syntactic and functional annotations for major constituents. The annotations have been manually validated, which makes the corpus a valuable resource for linguistic research but also for NLP applications.

The main goal of the project is to obtain a list of different subcategorization frames of French verbs as well as to enrich corpus annotations with this information. We aim also at estimating the number of verb frames in general and propose different methods to reduce the ambiguity rate.

2 Corpus Annotations

In the corpus, all main syntactic constituents are annotated but their internal structure is not indicated. For example, the boundaries of the adverb phrase *pas encore* ‘not yet’ in Fig. 1 are marked but its components (i.e., words *pas* ‘not’ and *encore* ‘yet’) are treated on a par, i.e., no structural relation between them is indicated.

The adopted annotation schema distinguishes a VP only for infinitive phrases. Instead, for inflected verbs, a verbal nucleus (VN) is defined and it contains the main verb, auxiliaries, negation, pronominal clitics and adverbs which follow the auxiliary. The head verb is not explicitly indicated but we assume that the last verb in VN is the head. Note that pronominal clitics, e.g., the pronominal subject *il* ‘he’ in Fig. 1, are not treated as syntactic NPs but are part of VN.

Syntactic functions are annotated only for verbal dependents. As shown in Fig. 1, the verb *sait* ‘knows’ has the subject (NP) and the object (a subordinate phrase, Ssub) indicated but no relation is specified between the noun *état-major* ‘management’ and its AP modifier *français* ‘French’. Functions are treated as relations between constituents (e.g., VN and NP *une bataille* ‘a battle’ in Fig. 1) and they do not link directly the head and its dependents (i.e., V and NP).

```
<SENT>
  <NP fct="SUJ">L'etat-major
     <AP>francais</AP> </NP>
  <VN>sait</VN>
  <Ssub fct="OBJ">qu'
      <VN fct="SUJ">il a gagne</VN>
      <NP fct="OBJ">une bataille</NP>,
          <COORD>mais
             <AdP>pas encore</AdP>
             <NP>la guerre</NP>
          </COORD>
   </Ssub>.
</SENT>
```

Fig. 1. Example of annotation schema: *L'état-major français sait qu'il a gagné une bataille mais pas encore la guerre* 'The French management knows that they won a battle but not yet the war'

Theoretical approaches use different representations of subcategorization frames. In some models, like LFG [3], the notation based on functional information is preferred (1), while in others, like LADL (lexicon–grammar of [15]), a categorial notation is adopted (2), yet in others, like HPSG [20], a mixed approach is used (3):

(1) <SUJ, OBJ>

(2) N0 V N1

(3) <SUJ:NP, OBJ:NP>

The first two approaches are not fully informative as both functions and categories can have multiple realizations. For example, a subject can be either nominal or sentential, whereas a postverbal NP can be considered either a direct object or an attribute. Since the corpus we are using contains both kinds of information, we adopt a mixed representation (3) in order to obtain more complete information. The functional representation (1) will be used for a comparison.

The list of categories and functions used in the corpus is presented in Tab. 2. The list ignores two functions: MOD, which always corresponds to non-subcategorized elements, and COORD, which represents coordinated phrases, relatively rare in the corpus, and which does not provide the category information. For prepositional complements, P-OBJ, we retain the type of the preposition which introduces the complement. This allows us to normalize verb frames with respect to active and passive forms.

3 Frame Extraction

3.1 Experiment

For extraction of the verb valency, we used the part of the corpus which contains both constituent and functional annotations, i.e., about 20 000 phrases (500 000

SUJ	NP, VPinf, Ssub, VN
OBJ	NP, AP, VPinf, VN, Sint, Ssub
DE-OBJ	VPinf, PP, Ssub, VN
A-OBJ	VPinf, PP, VN
P-OBJ	PP, AdP, VN, NP
ATO	Srel, PP, AP, NP, VPpart, VPinf, Ssub
ATS	NP, PP, AP, AdP, VPinf, Ssub, VPpart, Sint, VN

Fig. 2. Possible categories for every function of a verb. Functions: SUJ (subject), OBJ (direct object), DE-OBJ (indirect object introduced by *de*), A-OBJ (indirect object introduced by *à*), P-OBJ (a prepositional complement introduced by a different preposition), ATO (object's attribute), ATS (subject's attribute)

words). Our experiment was divided into two steps: first, verbs in the main clauses, i.e., verbs with all functions specified, have been used, which resulted in a lexicon of 1362 verb lemma (12 353 occurrences). Then, we complemented annotations for other verbs, e.g., we added missing subjects to imperative and infinitive forms and we completed frames of verbs in relative clauses. This resulted in 2006 verb lemma (25 076 occurrences) in the final verb lexicon.

As a starting point, we used the frames extracted directly from the corpus, without any modification and then we experimented with several methods to compact the frames. First, we separated function tags indicating clitic arguments. If there are several clitics attached to a verb, e.g., in *Il l'a vue* 'He has seen her', the subject *Il* 'he' and the direct object *l'* 'her/it', the two functions are indicated by a single tag `SUJ/OBJ` and they have to be separated. Clitics are not always associated with grammatical functions, e.g., *y* in the idiomatic expression *il y a* 'there is/are' or the reflexive clitic *se* in inherently reflexive verbs such as *s'evanouir* 'to faint'. Such clitics are nevertheless tightly dependent on the verb so we retain them in the subcategorization frames. In order to indicate clitics, we added two more functions: `refl` for reflexive clitics and `obj` for all other clitics. Moreover, a clitic and a constituent can have the same function. For example, in *Paul en mange-t-il beaucoup?* 'Has Paul eaten lots of them?' there are two subjects (*Paul* and *il*) and two objects (*en* and *beaucoup*). Such duplicated functions had to be eliminated. Finally, there are frames which are missing the subject. It has been added to the imperative forms and infinitives in subordinate clauses. There are two lemma which always appear without a subject, *voici* and *voilà* '(t)here is'. They are considered indicative verbs which do not have a subject.

We normalized frames with respect to passive vs. active form. We used a list of 62 verbs which can be inflected with the auxiliary *être* 'be' in order to distinguish past tense (fr. *passé composé*) and passive forms. If a verb appears with the auxiliary *être* 'be' but its past tense form requires another auxiliary (*avoir* 'have'), the form is considered passive and it is transformed to an active form. We add OBJ to the frame (as SUJ is already present), whereas if the PP expressing the agent is present, i.e., P-OBJ introduced by the preposition *par* or *de*, this PP is deleted. If the passive form appears with an ATS complement (the subject's attribute), we rename this function to ATO (the object's attribute). All other functions in the frame (if any) remain unchanged.

In French, syntactic arguments don't have to be realized in a fixed order. For example, the order of complements is relatively free, cf. (4) and (5), and the subject can also appear postverbally (subject inversion). The order in which functions appear in the frames does not reflect their surface order but has been normalized based on obliqueness and they are listed as follows: SUJ, OBJ, A-OBJ, DE-OBJ, P-OBJ, ATS, ATO, obj, refl. For instance, the verb *parle* 'talks' in (4) and (5), has the same subcategorization frame for both sentences (SUJ, A-OBJ, DE-OBJ):

(4) Marie parle [de ce problème] [à Paul].
Mary talks of this problem to Paul

Mary is talking to Paul about this problem

(5) Marie parle [à Paul] [de ce problème].

In some cases, corpus annotations turned out to be insufficient to extract correct frames. For example, only adverbial phrases but not adverbs alone have a grammatical function assigned. Therefore, the adverb *bien* 'well' is not recognized as a complement in *Elle va bien* 'She is doing well'. Then, only locally realized arguments of a verb are annotated so we do not capture dependents realized on a distance, e.g., in *Que peut faire le gouvernement?* 'What can the government do?', we extract (incorrectly) two objects for the verb *peut* 'can' (*que* 'what' and *faire* 'do') and none for the verb *faire*. Such cases are nevertheless quite rare.

3.2 Results

Below we present an analysis how different representations and parametrization techniques influence the number of extracted frames and their ambiguity rate. These results are provided for the initial data set, i.e., frames of verbs in main clauses. The impact of the size of the data set used is discussed in sec. 3.3.

Functional Representation. As indicated in Fig. 3, after neutralization of passive and active forms, we obtain 142 different subcategorization frames, with an average of 1.9 frames per verb lemma. Unsurprisingly the verb with the highest number of frames is *être* 'be' with 26 frames, whereas more than half of the verbs (849 lemmas) have exactly one subcategorization frame. Then we perform several operations in order to eliminate superfluous clitic arguments. We clean the frames so that duplicated functions are removed. After these modifications, we reduced the number of frames almost three times and we obtained 58 frames, with an average of 1.8 frames per verb lemma. If we additionally compact frames where a complement is realized either as an NP or a reflexive clitic, the ambiguity rate drops to 1.72 per verb, although the number of frames remains the same. The verb *être* still appears with the most frames (16) but the number of verbs with a single frame increases to 886.

	# frames	average	max. nr of frames	1 frame	
				%	#
passive	142	1.9	26 (*être*)	62.3%	849
clitics	58	1.8	16 (*être*)	63.1%	859
reflexive	58	1.72	16 (*être*)	65.1%	886

Fig. 3. Functional representation

`être (16 frames | 3842 tokens): SUJ, ATS (1632); SUJ (112); SUJ, OBJ, ATS (66); SUJ, OBJ (46); SUJ, P-OBJ (27); SUJ, DE-OBJ (21); SUJ, DE-OBJ, ATS (14); SUJ, P-OBJ, ATS (9); SUJ, A-OBJ (6); SUJ, A-OBJ, ATS (5); SUJ, OBJ, DE-OBJ (2); SUJ, OBJ, A-OBJ (2); SUJ, OBJ, A-OBJ, ATS (1); SUJ, A-OBJ, obj:en (1); SUJ, OBJ, P-OBJ (1); SUJ, P-OBJ, obj:en (1)`

`avoir (16 frames | 607 tokens): SUJ, OBJ (211); SUJ, OBJ, P-OBJ (65); SUJ, OBJ, ATO (11); SUJ (7); SUJ, A-OBJ (5); SUJ, OBJ, DE-OBJ (5); SUJ, OBJ, obj:y (4); SUJ, OBJ, A-OBJ (4); SUJ, obj:y (3); SUJ, P-OBJ (2); SUJ, A-OBJ, obj:y (1); SUJ, OBJ, P-OBJ, obj:y (1); SUJ, A-OBJ, DE-OBJ (1); SUJ, obj:y_en (1); SUJ, DE-OBJ (1); SUJ, DE-OBJ, P-OBJ (1)`

`faire (12 frames | 205 tokens): SUJ, OBJ (103); SUJ (19); SUJ, OBJ, A-OBJ (11); SUJ, OBJ, DE-OBJ (9); SUJ, ATS, refl (3); SUJ, obj:en (3); SUJ, P-OBJ, refl (2); SUJ, OBJ, P-OBJ (2); SUJ, OBJ, refl (2); SUJ, OBJ, obj:y (1); SUJ, DE-OBJ, ATO (1); SUJ, A-OBJ, refl (1)`

`rendre (12 frames | 34 tokens): SUJ, OBJ, ATO (15); SUJ, ATS (4); SUJ, A-OBJ, refl (3); SUJ, P-OBJ, ATS (2); SUJ, OBJ, A-OBJ (2); SUJ, OBJ (2); SUJ, P-OBJ, refl (1); SUJ, OBJ, DE-OBJ, refl (1); SUJ, OBJ, DE-OBJ, ATO (1); SUJ, OBJ, refl (1); SUJ, OBJ, A-OBJ, DE-OBJ (1); SUJ, obj:me (1)`

`passer (11 frames | 89 tokens): SUJ, P-OBJ (17); SUJ, DE-OBJ (16); SUJ (9); SUJ, OBJ (9); SUJ, A-OBJ (8); SUJ, A-OBJ, DE-OBJ (6); SUJ, OBJ, P-OBJ (2); SUJ, OBJ, refl (2); SUJ, OBJ, A-OBJ (2); SUJ, DE-OBJ, refl (1); SUJ, ATS (1)`

`laisser (10 frames | 43 tokens): SUJ, OBJ (23); SUJ, OBJ, A-OBJ (3); SUJ, OBJ, ATO (2); SUJ, A-OBJ (1); SUJ, OBJ, P-OBJ (1); SUJ (1); SUJ, OBJ, DE-OBJ (1); SUJ, OBJ, refl (1); SUJ, ATO (1); SUJ, OBJ, P-OBJ, refl (1)`

Fig. 4. Subcategorization frames (functional representation) for 6 most ambiguous verbs (10 frames or more)

Only 6 verbs have 10 frames or more and they are the most ambiguous French verbs: *être* 'be', *avoir* 'have', *faire* 'make', *rendre* 'return', *passer* 'pass', *laisser* 'allow'. Their frames with frequency counts are shown in Fig. 4.

As indicated in Fig. 5, the most frequent frames are SUJ–OBJ (more than half of the lemma, i.e., the verb types), SUJ (about a quarter of the lemmas), then SUJ–A-OBJ and SUJ–DE-OBJ and ditransitive verbs. Very few lemmas have a predicative complement but they are frequently used.

frame	# verb types	tokens
SUJ, OBJ	913 (67.0%)	6407 (51.9%)
SUJ, ATS	16 (1.2%)	1951 (15.8%)
SUJ	351 (25.8%)	1035 (8.4%)
SUJ, DE-OBJ	129 (9.5%)	558 (4.5%)
SUJ, OBJ, A-OBJ	162 (11.9%)	517 (4.2%)
SUJ, A-OBJ	103 (7.5%)	359 (2.9%)
SUJ, P-OBJ	85 (6.2%)	233 (1.9%)
SUJ, OBJ, P-OBJ	81 (5.9%)	197 (1.6%)
SUJ, OBJ, DE-OBJ	75 (5.5%)	160 (1.3%)
SUJ, A-OBJ, refl	55 (4.0%)	132 (1.1%)

Fig. 5. 10 most frequent frames (functional representation)

The drawback of the functional approach is that we have lost categorial information available in the corpus. For example, verbs with a sentential complement and verbs with a nominal complement are indistinguishable. Therefore, we turn to a mixed approach in order to obtain more complete information.

Mixed Representation. A mixed representation (with categories and functions), after depassivization, gives a gross total of 783 different subcategorization frames, with an average of 2.47 frames per lemma, and almost 58% of the lemmas which have only one frame. With the clitic factorization described in section 3.1, we obtain 300 different frames, with an average of 2.32 frames per lemma. The number of unambiguous verbs (with only one frame) does not raise much: 803 lemmas, that is almost 59% of the verbs.

We further factorize the subcategorization frames by the neutralization of the lexical value of a prepositional complement (indirect complements introduced by prepositions other than *à* or *de*). The average number of subcategorization frames drops slightly (2.27 frames per lemma) and so does the total number of frames (222). The number of unambiguous verbs (with only one subcategorization frame) remains the same (803). We then neutralize different realizations of the attribute (ATS and ATO) and types of a subordinate clause (interrogative, Sint, vs. subordinate, Ssub). The number of different frames drops to 173, whereas the ambiguity rate achieves 2.21. Next, we regroup frames which differ only in subject realization. For example, if the subject of a verb can be expressed either as a nominal or a clitic argument with all other arguments being the same, the two realizations are merged to form a single frame. This leads to 160 verb frames with 2 frames per verb on average. The final modification, concerning the neutralization of a complement as either a reflexive clitic or an NP, results in 1.91 frames per verb, or 858 unambiguous verbs.

As shown in Fig. 7, there are 12 verbs with more than 10 frames, with a maximum of 27 frames for *être* 'to be'. The general results are presented in Fig. 6. It is clear that the mixed approach is more precise than the functional one, since it comprises ca. 3 times more frames. But the average number of

	# frames	average	max. nr of frames	1 frame %	1 frame #
passive	453	2.47	100 (*être*)	57.9%	783
clitics	300	2.32	86 (*être*)	58.9%	803
prepositions	222	2.27	72 (*être*)	58.9%	803
attribute & subordinate	173	2.21	43 (*être*)	59.0%	804
subject	160	1.99	27 (*être*)	61.2%	833
reflexive	160	1.91	27 (*être*)	62.9%	858

Fig. 6. Mixed representation

```
être (27), avoir (22), faire (17), passer (12), rendre (12), rester (12),
porter (12), laisser (11), aller (10), dire (10), tenir (10), trouver
(10)
```

Fig. 7. 12 Most ambiguous verbs (10 frames or more); mixed representation

frames and the ambiguity rate are comparable. The number of frames may be further reduced if we compact frames with optional complements.

If we consider the most frequent subcategorization frames, we see that, as in the previous approach, most verbs have the direct transitive frame, followed by the strict intransitive one (SUJ, without any complements). We observe as well that verbs with a sentential complement are more frequent than with an infinitival one (both for verb types and tokens).

frame	# verb types	tokens
SUJ:NP, OBJ:NP	854 (62.7%)	4157 (33.6%)
SUJ:NP, ATS:XP	16 (1.2%)	1932 (15.6%)
SUJ:NP, OBJ:Ssub	95 (7.0%)	1186 (9.6%)
SUJ:NP	339 (24.9%)	1011 (8.2%)
SUJ:NP, OBJ:VPinf	40 (2.9%)	839 (6.8%)
SUJ:NP, DE-OBJ:PP	91 (6.7%)	380 (3.1%)
SUJ:NP, OBJ:NP, A-OBJ:PP	120 (8.8%)	348 (2.8%)
SUJ:NP, A-OBJ:PP	79 (5.8%)	223 (1.8%)
SUJ:NP, P-OBJ:PP	80 (5.9%)	218 (1.7%)
SUJ:NP, OBJ:NP, P-OBJ:PP	75 (5.5%)	185 (1.5%)

Fig. 8. 10 most frequent frames (mixed representation)

3.3 More Data

As indicated in Fig. 9, the size of the data set influences the results. If we consider all verbs (in main and subordinate clauses), the number of all frames and the ambiguity rate increase, for both representations. Although these changes are noticeable (8 new frames discovered for the functional and 20 for the mixed approach), they are not dramatic given that the number of verbs considered raises

representation	# lemmas	# frames	average	max. nr of frames	verbs with 1 frame	verbs with ≥ 10 frames
functional	1362	58	1.72	16 (être)	859 (63%)	6 (0.4%)
	2006	66	1.93	21 (être)	1183 (59%)	12 (0.6%)
mixed	1362	160	1.91	27 (être)	833 (61.1%)	13 (0.9%)
	2006	180	2.09	29 (être)	1168 (58.2%)	29 (1.4%)

Fig. 9. Comparison of results: verbs in main clauses (1362 types) vs. all verbs (2006 types)

by almost 70%. Moreover, the frequency of the new frames is very low, e.g., all new functional frames appear only once in the corpus, see Fig. 10, whereas only 6 of the 20 new mixed frames occur more than once, cf. Fig. 11. In particular, it should be verified if these frames are attested on a different data set or whether additional factorization techniques should incorporate them to the existing frames. For example, the reflexive clitic in Fig. 10 might be a result of insufficient factorization. Similarly, the frames with an apparently impersonal subject (`SUJ:il`) might be due to insufficient data: *il* 'it' is either an impersonal or a personal (3sg. masc) clitic. In the latter case, it can be replaced by an NP, hence the subject realization should be specified as SUJ:NP.

```
SUJ, DE-OBJ, P-OBJ, refl (1); SUJ, DE-OBJ, P-OBJ, ATS (1);
SUJ, A-OBJ, DE-OBJ, refl (1); SUJ, A-OBJ, DE-OBJ, ATS (1);
SUJ, A-OBJ, ATS, refl (1); SUJ, OBJ, DE-OBJ, ATS (1);
SUJ, A-OBJ, ATO (1); SUJ, obj:te (1)
```

Fig. 10. 8 additional functional frames with their frequencies

```
SUJ:NP, OBJ:VPinf, DE-OBJ:PP (11); SUJ:il, OBJ:AdP, obj:y (3);
SUJ:NP, OBJ:NP, P-OBJ:PP, refl:CL (3);
SUJ:NP, OBJ:NP, A-OBJ:VPinf, refl:CL (2);
SUJ:il, A-OBJ:PP, DE-OBJ:VPinf (2); SUJ:NP, OBJ:NP, A-OBJ:AP (2)
```

Fig. 11. 6 of the additional mixed frames which occur more than once

The majority of frames detected on the smaller sample are confirmed, i.e., their frequency increases or remains the same: 93% of functional and 83% of mixed frames found in both data sets. For the remaining shared frames, their frequency drops on the bigger data set. The main reason for this apparent paradox is that some frames get 'corrected' by supplementary data: for example, the 'impersonal' *il* subject turns out to be a personal pronoun if the verb appears with an NP subject as well (e.g., the frequency of `SUJ:il, OBJ:NP, A-OBJ:PP` drops from 6 to 3 in the final data set), or the frame has been reclassified with a different frame realized by the same verb (for instance, the initial `SUJ:NP, obj:en` is regrouped with `SUJ:NP, DE-OBJ:PP` which was found for the verb *faire* 'make/do' in the larger sample; this makes the overall frequency of the former frame drop from 7 to 6).

functional representation			mixed representation		
frame	v.types	v.tokens	frame	v.types	v.tokens
SUJ,OBJ	1431 [913]	13461 [6407]	SUJ:NP,OBJ:NP	1387 [854]	10257 [4157]
SUJ	730 [351]	3166 [1035]	**SUJ:NP**	717 [339]	3137 [1011]
SUJ, ATS	17 [16]	2582 [1951]	SUJ:NP,ATS:XP	17 [16]	2561 [1932]
SUJ,OBJ,A-OBJ	224 [162]	1083 [517]	SUJ:NP,OBJ:Ssub	115 [95]	1987 [1186]
SUJ,DE-OBJ	202 [129]	1028 [558]	SUJ:NP,OBJ:VPinf	40 [40]	1138 [839]
SUJ,A-OBJ	155 [103]	733 [359]	SUJ:NP,DE-OBJ:PP	162 [91]	843 [380]
SUJ,P-OBJ	150 [85]	494 [233]	SUJ:NP, OBJ:NP,A-OBJ:PP	183 [120]	770 [348]
SUJ,OBJ,DE-OBJ	186 [75]	468 [160]	SUJ:NP,A-OBJ:PP	128 [79]	518 [223]
SUJ,OBJ,P-OBJ	154 [81]	399 [197]	SUJ:NP,P-OBJ:PP	145 [80]	471 [218]
SUJ,OBJ,ATO	45 [32]	248 [114]	SUJ:NP, OBJ:NP,P-OBJ:PP	149 [75]	387 [185]

Fig. 12. Comparison of 10 most frequent frames

As frequencies change, the final ranking of frames is slightly different as well. For the ten top frames more regrouping occurs among the function-based frames (5 frames get promoted), whereas only strictly intransitive verbs appear more frequently if the mixed representation is considered. Fig. 12 presents a comparison of 10 top frames for both representations (numbers correspond to frequency counts in the bigger and smaller, in square brackets, data sets). Frames which get a higher rank in the final evaluation are boldfaced.

Finally, Fig. 9 shows that the number of most ambiguous verbs is doubled, for both representations. This indicates that frame ambiguity is in fact more common than predicted by our initial sample. The 29 verbs which belong to the class of more than 10 frames (mixed representation) are indicated in Fig. 13; the verbs which entered this class are written in boldface. The numbers in brackets indicate the number of frames associated with these verbs in the bigger and smaller samples.

être (29|27), avoir (22|22), faire (19|17), rester (17|12), passer (16|12), tenir (15|10), porter (15|12), trouver (14|10), **venir (14|9)**, **présenter (13|5)**, rendre (13|12), **attendre (12|8)**, dire (12|10), **vendre (11|6)**, **pouvoir (11|6)**, **voir (11|8)**, aller (11|10), **estimer (10|9)**, **revenir (10|8)**, **engager (10|6)**, laisser (10|11), **demander (10|9)**, **montrer (10|8)**, **devoir (10|8)**, **appeler (10|6)**, **déclarer (10|9)**, **permettre (10|8)**, **assurer (10|4)**, **mettre (10|8)**

Fig. 13. 29 Verbs with more than 10 (mixed) frames in the bigger sample

4 Conclusion

We presented results of an automatic frame extraction from a French treebank. We have succeeded in considerably reducing the number of verb frames by applying different factorization techniques. Despite the important difference in number of frames for the two kinds of representations we adopted, the average number of frames per verb is very similar. This fact speaks in favor of the mixed approach as more informative. Moreover, these numbers do not drastically change with the size of the data set which indicates that the number and types of frames has stabilized. On the contrary, the repertoire of frames for individual verbs is still growing.

We plan different extensions to the work presented here. We envisage extraction of subcategorization frames for other predicates (adjectives, nouns or adverbs). The frames need also to be validated and evaluated as we plan to use them to complete the syntactic annotations in the treebank. The lexicon can be easily integrated with other resources so it can be incorporated into syntactic parsers or NLP applications processing French.

The lexicon is freely available from the authors' web page:
`http://erssab.u-bordeaux3.fr/article.php3?id_article=150`.

References

1. Abeillé, A., Clément, L., Toussenel, F.: Building a treebank for French. In: Treebanks, Kluwer, Dordrecht (2003)
2. Bourigault, D., Frérot, C.: Acquisition et évaluation sur corpus de propriétés de sous-catégorisation syntaxique. In: Actes des 12èmes journées sur le Traitement Automatique des Langues Naturelles (2005)
3. Bresnan, J. (ed.): The Mental Representation of Grammatical Relations. MIT Press Series on Cognitive Theory and Mental Representation. MIT Press, Cambridge (1982)
4. Briscoe, T., Carroll, J.: Generalised probabilistic LR parsing for unification-based grammars. Computational linguistics (1993)
5. Cahill, A., et al.: Parsing with PCFGs and automatic f-structure annotation. In: Butt, M., King, T.H. (eds.) Proceedings of the LFG 2002 Conference, CSLI Publications (2002)
6. Carroll, J., Fang, A.: The automatique acquisition of verb subcategorisations and their impact on the performance of an HPSG parser. In: Proceedings of the 1st International Conference on Natural Language Processing, Sanya City, China (2004)
7. Chesley, P., Salmon-Alt, S.: Le filtrage probabiliste dans l'extraction automatique de cadres de sous-catǵorisation. In: Journé ATALA sur l'interface lexique-grammaire, Paris (2005)
8. Danlos, L., Sagot, B.: Comparaison du Lexique-Grammaire et de Dicovalence: vers une intégration dans le Lefff. In: Proceedings TALN 2007 (2007)
9. Danlos, L.: La génération automatique de textes, Masson (1985)
10. Davis, A., Koenig, J.-P.: Linking as constraints on word classes in a hierachical lexicon. Language 76(1), 56–91 (2000)
11. Falk, I., Francopoulo, G., Gardent, C.: Evaluer SynLex. In: Proceedings of TALN 2007 (2007)
12. Fort, K., Guillaume, B.: Preplex: a lexicon of French prepositions for parsing. In: ACL SIGSEM 2007 (2007)
13. Frank, A., et al.: From treebank resources to LFG f-structures. In: Treebanks, Kluwer, Dordrecht (2002)

14. Gardent, C., et al.: Extraction d'information de sous-catégorisation à partir du lexique-grammaire de Maurice Gross. In: TALN 2006 (2006)
15. Gross, M.: Méthodes en syntaxe. Hermann (1975)
16. Guillet, A., Leclère, C.: La structure des phrases simples en français. Droz, Genève (1992)
17. C.-H. Han, J., Yoon, N., Kim, M.: Palmer, A feature-based lexicalized tree adjoining grammar for Korean. Technical report, IRCS (2000)
18. Hornby, A.S.: Oxford Advanced Learner's Dictionary of Current English, 4th edn. Oxford University Press, Oxford (1989)
19. Mel'cuk, I., Arbatchewsky-Jumarie, N., Clas, A.: 1984, 1988, 1992, 1999. Dictionnaire explicatif et combinatoire du français contemporain. Recherches lexico-sémantiques, vol. I, II, III, IV. Les Presses de l'Université de Montréal
20. Pollard, C., Sag, I.A.: Head-driven Phrase Structure Grammar. Chicago University Press / CSLI Publications, Chicago, IL (1994)
21. Procter, P. (ed.): Longman Dictionary of Contemporary English, Longman, Burnt Mill, Harlow (1978)
22. Sagot, B., Danlos, L.: Améliorer un lexique syntaxique à l'aide des tables du lexique-grammaire. constructions impersonnelles. In: Proceedings of TALN 2007 (2007)
23. Sagot, B., et al.: The lefff 2 syntactic lexicon for french: architecture, acquisition, use. In: Actes de LREC 2006, Gênes, Italie (2006)
24. Surdeanu, M., et al.: Using predicate-argument structures for information extraction (2003)
25. van den Eynde, K., Mertens, P.: La valence: l'approche pronominale et son application au lexique verbal. French Language Studies 13, 63–104 (2003)

Acquisition of Elementary Synonym Relations from Biological Structured Terminology

Thierry Hamon[1] and Natalia Grabar[2]

[1] LIPN – UMR 7030, Université Paris 13 – CNRS, 99 av. J-B Clément, F-93430 Villetaneuse, France
thierry.hamon@lipn.univ-paris13.fr
[2] Université Paris Descartes, UMR_S 872, Paris, F-75006 France
INSERM, U872, Paris, F-75006, France
natalia.grabar@spim.jussieu.fr

Abstract. Acquisition and enrichment of lexical resources have long been acknowledged as an important research in the area of computational linguistics. Nevertheless, we notice that such resources, particularly in specialised domains, are missing. However, specialised domains, *i.e.* biomedicine, propose several structured terminologies. In this paper, we propose a high-quality method for exploiting a structured terminology and inferring a specialised elementary synonym lexicon. The method is based on the analysis of syntactic structure of complex terms. We evaluate the approach on the biomedical domain by using the terminological resource `Gene Ontology`. It provides results with over 93% precision. Comparison with an existing synonym resource (the general-language resource `WordNet`) shows that there is a very small overlap between the induced lexicon of synonyms and the `WordNet` synsets.

1 Background

Acquisition and enrichment of lexical resources have long been acknowledged as an important research in the area of computational linguistics. Indeed, such resources are often helpful for the deciphering and computing semantic similarity between words and terms within tasks like information retrieval (especially query expansions), knowledge extraction or terminology matching.

We make the distinction between terminological and lexical resources. The aim of terminological resources is collecting terms used in a specialised area, describing and organizing them. Within terminologies, terms can be simple (*reproduction*) but mostly complex (*formation of catalytic spliceosome for first transesterification step*; *cell wall mannoprotein synthesis*). They can be linked between them with semantic relations (hierarchical, synonymous, ...). Other features of terms (*i.e.*, definitions, areas of usage) can be precised. As for lexical resources, they gather mostly simple lexical units (*i.e.*, synonyms like *formation*, *synthesis* and *biosynthesis*). These units can belong to common language or be specific to some specialised languages. They can receive descriptions (syntactic, phonetic, morphological, ...) or propose relations between them. Our observation is that units from lexical resources, being simpler linguistic units, are often parts of terms and are spontaneously used during their creation. If terms, usually complex (*i.e.* synonyms like *aromatic amino acid family biosynthesis* and *aromatic amino acid family formation*), can be hardly generalized for being used in various tasks of computational linguistics, their components (*biosynthesis* and *formation* in the given example) are more suitable candidates for the building of a lexicon and their use in natural language processing applications.

Synonym lexicon, as well as lexicon of morphological or orthographic variants, can be used for the task of deciphering semantic relations between terms or words. But not all of these resources are equally well described for various specialised languages and this observation is also true for specialised domains. We are concerned with this remark as our special interest is related to the biomedical domain.

Thus, the morphological description of languages is the most complete and several languages are provided with at least inflectional lexica (widely used within syntactic tools for POS-tagging and lemmatisation [1,2,3]), or even specific databases (such as `Celex` base [4] for inflectional and derivational description of English and German, or `MorTal` [5] for French). As for the biomedical domain, we can mention the widely used `UMLS` Specialized Lexicon [6] for English, and similar resources for German [7] and French [8].

But when one looks for the description of synonymous or orthographic relations, little available resources can be found. If `WordNet` [9] proposes synonym relations for English, the corresponding resources for other languages are not freely available; while the initiative for tuning this resource for the medical area [10] is still ongoing. Moreover, it has been shown that general lexica, for instance `Wordnet`, are insufficient for specialised knowledge extraction [11]. Indeed, additional specialised information is crucial to improve the coverage and the completeness of the extraction based on general-language resources. To find a solution for this, we propose to use specialised terminologies, as several of them are created and continuously updated in biomedical area. In this work, we propose a novel high-quality method for the acquisition of lexical resources of synonyms from structured terminologies. This method is language-independent. It is based on the identification of syntactic invariants. As indicated, we position our research in the domain of biology.

In the following of this paper, we start with the presentation of the material used (sec. 2), we present then the undergoing hypothesis and various steps of the method proposed (sec. 3). We describe and discuss the obtained results (sec. 4) and conclude with some perspectives to this work (sec. 5).

2 Material

2.1 Structured Terminology of Biology: `Gene Ontology`

In the current work, we use the `Gene Ontology` (*GO*) [12] as the original resource from which elementary synonym relations are inferred. The goal of the *GO* is to produce a structured, precisely defined, common, controlled vocabulary for describing the roles of genes and their products in any organism. The project started in 1998 as a collaboration between databases of three model organism: fly *Drosophila*, yeast *Saccharomyce* and mouse. Since then, *GO* is used and enriched by other databases (genomes of plants, animals and micro-organisms).

GO terms describe one of three types of biological meanings, structured into three hierarchical trees: biological processes, molecular functions and cellular components. These trees have been chosen because they represent knowledge useful for the functional annotation of the majority of organisms and can be used for the description of genes and their products from various species. Terms are structured through three types of relations: subsumption `is-a`, partonomy `part-of` and synonymy.

The used version of *GO* contains 18,315 terms linked with 24,537 `is-a` relations and 2,726 `part-of` relations. These terms have 13,850 synonyms. The whole set of terms contains 23,899 terms, both preferred and synonyms.

In our work, we use the synonymous relations between terms.

2.2 General-Language Resource: `WordNet`

`WordNet` [9] is a large lexical database of English, developed and maintained at Princeton University since 1985, and adapted to other languages. Within this database, nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (called *synsets*), each expressing a distinct concept. Synsets are interlinked by means of semantic and lexical relations. The version used provides 81,426 noun synsets, 13,650 verb synsets, 18,877 adjectival and 3,644 adverbial synsets.

`WordNet` synsets are used for the evaluation of coverage of the inferred resource.

3 Methods

3.1 Preliminary Observations: Compositionality of *GO* Terms

Often within *GO*, terms are coined on the same scheme which can be exploited in order to induce the elementary relations between words or simple terms. For instance, the *GO* concept GO:0009073 contains the following series of terms, which show the compositionality through the substitution of one of their components (underlined in the examples):

aromatic amino acid family <u>*biosynthesis*</u>
aromatic amino acid family <u>*anabolism*</u>
aromatic amino acid family <u>*formation*</u>
aromatic amino acid family <u>*synthesis*</u>

On the basis of this set, it is possible to exploit the compositional structure of terms and thus to induce the following paradigm of synonymous words (or simple terms):

biosynthesis, *anabolism*, *formation*, *synthesis*

In the following, we call the series of synonym terms *original* synonym relations; and series of their substituted components *induced* or *elementary* synonym relations.

We propose a method for the generalization of this observation in order to allow acquiring specialised lexicon of elementary synonymous relations. Like in the given examples, the method exploits the compositional structure of terms and relies on existence of structured terminologies. The notion of compositionality, central for this method, assumes that the meaning of a complex expression is fully determined by its syntactic structure, the meaning of its parts and the composition function [13]. We propose to apply this principle for building a lexicon of elementary synonym relations. Moreover, it has been observed that a large part of *GO* terms indeed verify the compositionality principle [14].

In order to be able to exploit this principle, terms are first analysed syntactically into head and expansion components (sec. 3.2), then specific inference rules are applied (sec. 3.3), and the obtained results are evaluated (sec. 3.4).

3.2 Preprocessing of Terminology

The aim of terminology preprocessing step is to provide the syntactic analysis of terms. Such analysis is crucial for our work: the method we propose exploits syntactic dependency relations and is based on syntactic invariants. Hence, each *GO* term must be linguistically analysed in order to prepare and perform syntactic analysis.

In our work, we use the `Ogmios` platform [15], which is suitable for the processing of large amount of data and, moreover, can be tuned to a specialised domain. Through the platform, several types of linguistic processing are performed. First, the `TagEN` [16] tool is applied for the recognition on named entities. Its use at the beginning of linguistic pipeline helps the forthcoming segmentation into words and sentences. Indeed, the recognition of named entities (*i.e.*, gene names, chemical products) allows disambiguating special characters, such as punctuation marks, dashes, slashes, etc, widely used within named entities in biology and often altering the segmentation into word and sentence. After the segmentation, the POS-tagging and lemmatisation are performed with the `GeniaTagger` [17] tool, specifically trained for the biomedical domain.

The step of syntactic parsing of terms is carried out thanks to the rule-based term extractor $\mathrm{Y_AT_EA}$ [18]. The syntactic dependency relations between term components are computed according to assigned POS tags and parsing rules implemented within $\mathrm{Y_AT_EA}$. Thus, each term is considered as a syntactic binary tree (see figure 1) composed of two elements: head component and expansion component. For instance, *anabolism* is the head component of *acetone anabolism* and *acetone* is its expansion component.

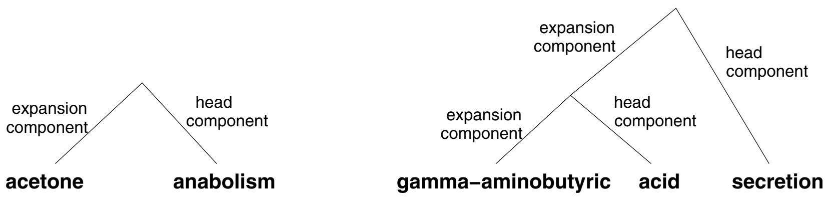


Fig. 1. Parsing tree of the terms *acetone anabolism* and *gamma-aminobutyric acid secretion*

3.3 Acquisition of Synonym Lexicon

The present method is inspired by our previous work [11], where we proposed to apply the semantic compositionality principle for inferring synonymy relations between complex terms. We then postulated that the composition process preserves synonymy and that the compositionality principle holds for complex terms. Roughly, this means that if the meaning $\mathcal{M}$ of two complex terms $A\ rel\ B$ and $A'\ rel\ B$ are given by the following formulas:

$$\mathcal{M}(A\ rel\ B) = f(\mathcal{M}(A), \mathcal{M}(B), \mathcal{M}(rel))$$

and

$$\mathcal{M}(A'\ rel\ B) = f(\mathcal{M}(A'), \mathcal{M}(B), \mathcal{M}(rel))$$

for a given composition function f, and if A and A' are synonymous ($\mathcal{M}(A) = \mathcal{M}(A')$), then the synonymy of the complex terms can be inferred:

$$\begin{aligned} \mathcal{M}(A'\ rel\ B) &= f(\mathcal{M}(A'), \mathcal{M}(B), \mathcal{M}(rel)) && (1)\\ &= f(\mathcal{M}(A), \mathcal{M}(B), \mathcal{M}(rel)) && (2)\\ &= \mathcal{M}(A\ rel\ B) && (3) \end{aligned}$$

In the current work, we assume that the inverse function f^{-1} exists and can be applied for deducing elementary synonym relations given synonymous complex terms. As in the cited work [11], our approach takes into account the internal structure of the complex terms. We assume that the syntactic dependency relation between components is preserved through the compositionality principle. Thus, we can infer elementary synonym relations between components of two terms if:

– parsed terms are synonymous;
– these components are located at the same syntactic position (head or expansion);
– the other components within terms are either synonymous or identical.

The fully parsed terms are represented as a terminological network, within which the deduction of the elementary synonym relations is based on the three following rules:

Rule 1. If both terms are synonymous and their expansion components are identical, then an elementary synonym relation is inferred. For instance,

we can infer the synonym relation {*B-lymphocyte*, *B-cell*} from the original synonym relation between terms:
peripheral B-lymphocyte and *peripheral B-cell*
where the expansion component *peripheral* is identical in both terms.

Rule 2. If both terms are synonymous and their head components are identical, then an elementary synonym relation is inferred. For instance, we infer the synonym relation {*endocytic*, *endocytotic*} from the synonym relation between terms:
endocytic vesicle and *endocytotic vesicle*
where the head component *vesicle* is identical.

Rule 3. If both terms are synonymous and either their head components or expansion components are synonymous, then an elementary synonym relation is inferred. For instance, we infer the synonym relation {*nicotinamide adenine dinucleotide*, *NAD*} from the synonym relation between terms:
nicotinamide adenine dinucleotide catabolism and *NAD breakdown*
where the head components {*catabolism*, *breakdown*} are already known synonyms.

The method is recursive and each inferred elementary synonym relation can then be propagated in order to infer new elementary relations, which allows to generate a more exhaustive lexicon of synonyms.

3.4 Evaluation

We perform manual validation of the inferred elementary relations between words and simple terms. For this, each pair is examined, as well as its source series of synonyms. Accuracy of the inferred pairs is thus computed. Moreover, we make an attempt to compare the inferred resource with `WordNet` synsets and compute the overlap between them. The both sets of synonyms are compared once lemmatised.

4 Results and Discussion

4.1 Preprocessing of Terminology: Ogmios Platform

23,899 *GO* terms have been fully parsed through the platform Ogmios. Thus, 15,863 original synonym relations could be used for inferring elementary relations.

4.2 Acquisition of Synonym Lexicon

The three rules defined for inferring elementary relations have been applied to the terminological network formed with 15,863 original *GO* synonym terms. In this way, 921 pairs of elementary synonym relations have been induced.

Our general observation is that, among these inferred pairs, very few (around ten) are induced from a large number of original *GO* synonyms, while the most

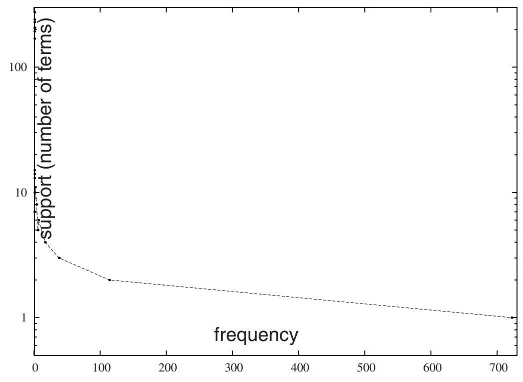


Fig. 2. Support and frequency observed when inferring elementary synonymous relations from *GO* terms (logarithmic scale axis)

of inferred pairs are supported by a small number of *GO* terms. For instance, 274 *GO* synonymous series allow to infer the pair {*breakdown*, *catabolism*}, which thus appears to be a fundamental notion in biology. The pair {*formation*, *synthesis*} is acquired on 240 original terms, the pair {*catabolism*, *degradation*} on 229 term pairs, etc. Pairs like {*adrenaline*, *epinephrine*}, {*gallate*, *gallic acid*}, {*formation*, *growth*}, {*flagella*, *flagellum*}, {*F-actin*, *actin filament*}, {*eicosanoid*, *icosanoid*} are acquired on small number of original term pairs (1 to 3). Such pairs correspond mainly to chemical products, to Latin inflected words or to orthographic variants. These represent nearly 80% (n=722) of the whole number of inferred synonym pairs. They may show smaller semantic acceptance of their paradigms, but this should be verified through their implementation within corpora and applications.

Figure 2 represents this observation graphically by combining figures of support of inferred pairs (number of original *GO* synonyms that allow to infer them) and of frequency of each support value (axis are scaled logarithmically). We can see that the inducing of elementary relations from *GO* terms follows a hyperbolic distribution. Although such distribution is observed in language and is often referred to as Zipf law, we assume that in our experience this situation is also due to the strong policy used by `Gene Ontology` Consortium. As a matter of fact, creation of new terms is governed by *GO* guidelines[1] and the vocabulary (*GODict.DAT*) of words already used within *GO* terms.

Another observation we can make on the basis of this data is that the compositionality is indeed a widely verified principle within *GO* terms, as it was observed in previous work [19,14,20]. In our experience, the large values of support confirm this and attest that the compositionality is indeed applied at large scale for coining new *GO* terms.

Additionally, the acquired synonym pairs can be classified according to their linguistic or semantic types.

[1] *http://www.geneontology.org/GO.usage.shtml*

Linguistic Typology of Synonym Pairs. The linguistic types of elementary synonymous relations can be defined further to a manual analysis:

– Orthographic variants:
  {*synthase*, *synthetase*}, {*leucocyte*, *leukocyte*}, {*sulfate*, *sulphate*}
– Hyphenation variants, which can be considered as part of orthographic variants but have the specificity of being always concerned with the same type of variation (presence or absence of the hyphen):
  {*B-cell*, *B cell*}
– Word ordering: {*gamma-delta T-cell*, *T gamma-delta cell*}
– Abbreviations, which are widely used in biological domain, apply various tactics for the coining of abbreviated terms and words:
  - standard abbreviation through acronym formation:
    {*ER*, *endoplasmic reticulum*}
  - acronym formation at morphological level:
    {*DPH*, *dehydropeptidase*}, {*IL-10*, *interleukin-10*}
  - syllabic abbreviation: {*Eph*, *ephrin*}, {*Gly*, *glycine*}
  - combined abbreviation: {*TGase*, *transglutaminase*},
– Use of symbols, which is also very frequent in biological domain:
  {*1-b-glucosyltransferase*, *1-beta-glucosyltransferase*},
  {*D-isomerase*, *delta-isomerase*},
  {*omega-amidase*, *w-amidase*}
– However, most of the induced synonym pairs link entities for which no common formal features can be observed:
  {*hydroxylase*, *monooxygenase*}, {*vitamin Bh*, *myo-inositol*}, {*cell*, *lymphocyte*}, {*apyrase*, *nucleoside-diphosphatase*}, {*myrosinase*, *sinigrinase*}, {*invertase*, *saccharase*}, {*regulator activity*, *modulator*}, {*Valium*, *diazepam*}

The method we propose is specifically useful for the acquisition of this last type of synonyms which are difficult to detect otherwise, *i.e.* on the basis of their formal feature (internal structure, morphology, etc.).

Semantic Typology of Synonym Pairs. Semantic types of the inferred pairs of synonyms could be defined according to the hierarchical trees of the `Gene Ontology` (biological processes, molecular functions and cellular components) within which the elementary relations have been inferred.

For instance, the synonym series which show an important support:

– *biosynthesis*, *synthesis*, *formation*, *anabolism*
– *breakdown*, *degradation*, *catabolism*

correspond to fundamental biological processes and remain specific to this hierarchical tree of *GO*.

The pair {*cell*, *lymphocyte*} is specific notion of cellular component tree but is, in fact, widespread over all the *GO* terms: the majority of biological processes and molecular functions are located at the cell level. The same observation can be done for {*ER*, *endoplasmic reticulum*} pair, which stands for a cellular component, but is the place of many biological processes and molecular functions.

As for {*DPH*, *dehydropeptidase*}, {*Eph*, *ephrin*} or {*Gly*, *glycine*} pairs, they are molecular functions inferred from few *GO* series of synonyms.

Such semantic typology, based on the hierarchical organization of *GO*, gives some insights into the language usage within biological domain and, more specifically, within *GO*. We assume a more fine-grained typology can be proposed, distinguishing in addition semantic types like phenotype, chemical products, pathological processes, etc.

Contextual Nature of Synonymous Relations. An additional remark can be made on the nature of the synonymous relations. Like in in [21], we consider synonymy as a Boolean rather than as a scaling property. Thus, we define synonymy as a sort of *contextual cognitive synonymy*: X is a cognitive synonym of Y relatively to a context C if (i) X and Y are syntactically identical, and (ii) any grammatical declarative sentence S containing X in the context C has equivalent truth-conditions to another sentence S', which is identical to S except that, in C, X is replaced by Y [21, p.88]. In this way, our synonymy definition is close to that of `WordNet`: as far as we can observe at least one context within which a pair of words is synonymous we record these words as true synonyms in the inferred lexicon.

For instance, the pair {*cell*, *lymphocyte*} can be observed in several synonymous terms within *GO*:

establishment of B cell polarity; *establishment of B lymphocyte polarity*
T cell homeostatic proliferation; *T lymphocyte homeostatic proliferation*
B-cell homeostasis; *B-lymphocyte homeostasis*
T cell mediated cytotoxicity; *T lymphocyte mediated cytotoxicity*

For this reason, this inferred pair of synonyms is counted as correct, even if the common feeling about it would be that *cell* is a more general term than *lymphocyte*.

The validity of such pairs within other contexts should be verified.

4.3 Evaluation

The manual evaluation, performed by a computational scientist, shown that 93.1% (n=857) are correct, 5.4% (n=50) rejected and 1.5% (n=14) remain undecided. This evaluation is supported by the analysis of both inferred and initial synonym pairs.

The efficiency of the proposed method is very high. This is due to the fact that the acquisition is performed on controlled terminological data. Moreover, the inferring rules strongly exploit syntactic scheme within syntactically analyzed terms. Finally, as we observed, the compositionality principle is widely applied for the coining of new *GO* terms. All these factors can but contribute to the acquisition of high-quality synonym pairs.

We attempted a comparison between the induced elementary synonymous pairs and the synsets provided by `WordNet`. Unsurprisingly, the overlap is very low. As a matter of fact, any of the inferred synonym sets can be completely matched with any of the synsets. Although we can find partial overlapping between these two resources. For instance, the inferred set *biosynthesis*, *synthesis*, *formation*, *anabolism* partly overlaps with the following synsets:

– *biosynthesis*
– *biosynthesis*, *biogenesis*
– *constitution*, *establishment*, *formation*, *organization*, *organisation*
– *formation*, *shaping*
– *formation*
– *anabolism*, *constructive metabolism*
– *synthesis*

and shows to have no common meaning with other synsets, for instance *deduction*, *deductive reasoning*, *synthesis*.

Otherwise, there is difference in namings, for instance {*cell*, *lymphocyte*} in the inferred resource and {*lymphocyte*, *lymph cell*} as proposed by `WordNet`. But, usually, the inferred resource proposes more specialised notions: {*ER*, *endoplasmic reticulum*} and *endoplasm* in `WordNet`. Finally, many of these notions do not occur within `WordNet`.

The difference between the two compared resources is not surprising as the purpose, as well as aimed applications, of the `WordNet` and of the `Gene Ontology` are different. `WordNet`, being the only available resources of synonyms, is sometimes applied in specialised domains. For instance, its use for terminology structuring and knowledge extraction shown that such general lexica are insufficient for specialised domains [11] and should be completed with specialised resources. Indeed, specialised domains make use of concepts too specific to occur within a general language lexicon. The common-language resources have been proposed to be adapted to a given domain through corpus-based filtering, even though they do not represent the richness of this specialised language [22]. Another experiences demonstrated that although the suitability of the general-language resources for biomedical area is low [23,24], they can be used as layer which could adapt high technical level information to lay people understanding. In this case, definitions as those proposed by `WordNet` are helpful.

5 Conclusions and Perspectives

Although there is a huge need in various types of linguistic resources, some types of such resources are missing especially in specialised domains. For instance, in many areas of the natural language processing synonym resources are widely needed. In this work, we propose a novel method for filling in the gap and inferring elementary synonymous relations. This method exploits the compositionality principle, when it is verified, and relies on existence of structured terminologies. It applies set of rules based on syntactic dependency analysis within terms.

The proposed method has been applied to `Gene Ontology`, a terminological resource of biology. It provides high-quality results: over 93% of inferred relations prove to be correct. However, the synonymy is as contextual relation and the validity of some inferred pairs should be tested on corpora. The attempted comparison with the available resource of synonym relations, proposed by the `WordNet`, is very low: in the best case scenario, the overlap is partial. But often the inferred notions are missing in `WordNet`.

In the next future, we plan to use the inferred synonym relations for enriching and extending the `Gene Ontology`. We will also test their efficiency with other biomedical terminologies. But we assume this resource can be used in many other applications of computational linguistics.

As we noticed, the method is language-independent, and it is possible to apply it to other languages as far as (1) the required linguistic processing can be realised and (2) synonym relations between complex terms are available. For this purpose, we can use for instance the `UMLS` resource [6], or more specifically the `MeSH` [25] or `Snomed` [26] terminologies which are available in several languages.

References

1. Brill, E.: A Corpus-Based Approach to Language Learning. PhD thesis, University of Pennsylvania, Philadelphia (1993)
2. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: Proceedings of the International Conference on New Methods in Language Processing, Manchester, UK, pp. 44–49 (1994)
3. Namer, F.: FLEMM: un analyseur flexionnel du français á base de règles. Traitement Automatique des Langues (TAL) 41(2), 523–547 (2000)
4. Burnage, G.: CELEX - A Guide for Users. Centre for Lexical Information, University of Nijmegen (1990)
5. Hathout, N., Namer, F., Dal, G.: An experimental constructional database: the MorTAL project. In: Boucher, P. (ed.) Morphology book, Cascadilla Press, Cambridge (2001)
6. NLM: UMLS Knowledge Sources Manual. National Library of Medicine, Bethesda, Maryland (2007), `www.nlm.nih.gov/research/umls/`
7. Schulz, S., et al.: Towards a multilingual morpheme thesaurus for medical free-text retrieval. In: Medical Informatics in Europe (MIE) (1999)
8. Zweigenbaum, P., et al.: Towards a Unified Medical Lexicon for French. In: Medical Informatics in Europe (MIE) (2003)
9. Fellbaum, C.: A semantic network of english: the mother of all WordNets. Computers and Humanities. EuroWordNet: a multilingual database with lexical semantic network 32(2–3), 209–220 (1998)
10. Smith, B., Fellbaum, C.: Medical wordnet: a new methodology for the construction and validation of information. In: Proc. of 20th CoLing, Geneva, Switzerland, pp. 371–382 (2004)
11. Hamon, T., Nazarenko, A.: Detection of synonymy links between terms: experiment and results. In: Recent Advances in Computational Terminology, pp. 185–208. John Benjamins (2001)
12. Gene Ontology Consortium: Creating the Gene Ontology resource: design and implementation. Genome Research 11, 1425–1433 (2001)
13. Partee, B.H.: In: Compositionality. F. Landman and F. Veltman (1984)
14. Ogren, P., et al.: The compositional structure of Gene Ontology terms. In: Pacific Symposium of Biocomputing, pp. 214–225 (2004)

15. Hamon, T., et al.: A robust linguistic platform for efficient and domain specific web content analysis. In: RIAO 2007, Pittsburgh, USA (2007)
16. Berroyer, J.F.: Tagen, un analyseur d”entits nommes: conception, développement et valuation. Mémoire de D.E.A. d'intelligence artificielle, Universit Paris-Nord (2004)
17. Tsuruoka, Y., et al.: Developing a robust part-of-speech tagger for biomedical text. In: Bozanis, P., Houstis, E.N. (eds.) PCI 2005. LNCS, vol. 3746, pp. 382–392. Springer, Heidelberg (2005)
18. Aubin, S., Hamon, T.: Improving term extraction with terminological resources. In: Salakoski, T., et al. (eds.) FinTAL 2006. LNCS (LNAI), vol. 4139, pp. 380–387. Springer, Heidelberg (2006)
19. Verspoor, C.M., Joslyn, C., Papcun, G.J.: The gene ontology as a source of lexical semantic knowledge for a biological natural language processing application. In: SIGIR workshop on Text Analysis and Search for Bioinformatics, pp. 51–56 (2003)
20. Ogren, P., Cohen, K., Hunter, L.: Implications of compositionality in the Gene Ontology for its curation and usage. In: Pacific Symposium of Biocomputing, pp. 174–185 (2005)
21. Cruse, D.A.: Lexical Semantics. Cambridge University Press, Cambridge (1986)
22. Grabar, N., Zweigenbaum, P.: Utilisation de corpus de spécialité pour le filtrage de synonymes de la langue générale. In: Traitement Automatique de Langues Naturelles (TALN) (2005)
23. Bodenreider, O., Burgun, A.: Characterizing the definitions of anatomical concepts in WordNet and specialized sources. In: Proceedings of the First Global WordNet Conference, pp. 223–230 (2002)
24. Bodenreider, O., Burgun, A., Mitchell, J.A.: Evaluation of WordNet as a source of lay knowledge for molecular biology and genetic diseases: a feasibility study. In: Medical Informatics in Europe (MIE), pp. 379–384 (2003)
25. National Library of Medicine Bethesda, Maryland: Medical Subject Headings (2001), `http://www.nlm.nih.gov/mesh/meshhome.html`
26. Côté, R.A.: Répertoire d'anatomopathologie de la SNOMED internationale, v3.4. Université de Sherbrooke, Sherbrooke, Québec (1996)

A Comparison of Co-occurrence and Similarity Measures as Simulations of Context

Stefan Bordag

Natural Language Processing Department, University of Leipzig
sbordag@informatik.uni-leipzig.de

Abstract. Observations of word co-occurrences and similarity computations are often used as a straightforward way to represent the global contexts of words and achieve a simulation of semantic word similarity for applications such as word or document clustering and collocation extraction. Despite the simplicity of the underlying model, it is necessary to select a proper significance, a similarity measure and a similarity computation algorithm. However, it is often unclear how the measures are related and additionally often dimensionality reduction is applied to enable the efficient computation of the word similarity. This work presents a linear time complexity approximative algorithm for computing word similarity without any dimensionality reduction. It then introduces a large-scale evaluation based on two languages and two knowledge sources and discusses the underlying reasons for the relative performance of each measure.

1 Introduction

One way to simulate associative and semantic relations between words is to view each word as a distinct entity. That entity may occur in a linear stream of sentences or other easily observable linguistic units. It is then possible to measure the statistical correlation between the common co-occurrence of such entities (i.e. words) within these units [1,2]. If additional knowledge such as word classes or morphological relatedness is available, this model allows to construct a variety of applications that depend on knowledge about word relatedness, but do not necessarily need this knowledge to be precise. For example, it is sufficient to know the most significant co-occurring word pairs in a corpus to enable the creation of a helpful tool for extraction of collocations, idioms or multi-word-expressions [3,4,5]. Similarly, knowledge about contextual similarity modeled as co-occurrence vector comparisons helps to build thesaurus construction tools such as the Sketch engine [6] or to design specific semi-automatic algorithms that create approximations of a thesaurus [7,8,9,10].

Assuming the simple vector-space model where each word defines a new dimension, the question arises how exactly significant co-occurrence or word similarity is to be modeled. Several variations of the same underlying vector space model were proposed. One is to apply Latent Semantic Indexing (LSI) to the matrix containing the raw co-occurrence counts of words [11]. However, it is unclear what the abstract concepts LSI generates are and whether similarity based on raw frequency counts can achieve the best performance in subsequent applications. Additionally, the gain in computational efficiency by operating on the reduced set of dimensions can be easily outperformed by an approximative algorithm that makes use of the fact that the co-occurrence matrix is sparse, see Section 2.3 below.

Another possibility to model the word space is to apply a significance measure to the co-occurrence counts. Then the ranking of globally most significant word pairs can be used directly as an indicator for possible idiomatic usage of these words [5]. Alternatively, the local ranking of most significant co-occurrences of any word can be used as a condensed contextual representation of that word. These contexts allow to simulate word similarity because obviously, if two words share many significant co-occurrences, then they are probably related to each other. This, in turn, allows to compute rankings of most similar words. These per-word rankings of most significant co-occurrences or most similar words are directly used in applications such as the Sketch engine, automatic thesaurus construction algorithms [12], or document clustering algorithms which accumulate the context representations of the words in a document into a single large vector.

One crucial aspect for all models is the usage of significance and similarity measures. Due to the large number of publications devoted to the development of new measures [13,14,15,16,17], several publications have recently appeared that attempt to measure and compare the performance of some of these measures. However, they either compare only word similarity measures on small evaluations sets (i.e. only 300 nouns [18] or the 80 TOEFL test questions [19]) or are concerned exclusively with the global view [5] which is incompatible with the majority of applications mentioned above. Alternatively, some evaluations are based on creating and then measuring retrieval quality of artificial synonyms [12], a pseudo-disambiguation task [20,21] or comparing the output of the measures to thesauri [22] or a combination of several of several such methods [23,24,2].

Contrary to the previous evaluation efforts this work provides a robust, large-scale evaluation of the most common measures and a discussion of the reasons for the observed differences based on proper statistical significance tests (i.e. not the t-test) to gauge the observed differences.

2 Measures

The measures to be evaluated are inherently divided into two consecutive steps. In order to compare words for similarity, the first operation is to observe co-occurrence frequencies n_{AB} between any word A and B. These are then interpreted by a co-occurrence significance measure, given the individual word frequencies n_A, n_B and the corpus size n. Out of the many possible measures those were chosen that are either frequently used in related work or are statistically well-founded.

2.1 Co-occurrence Measures

As most measures have already been described in great detail, the derivation of most measures is only referenced to in this work. However, for some co-occurrence measures their explicit form with respect to the four observable variables n_{AB}, n_A, n_B and n is given, which in some cases helps to avoid difficulties with too small probability values or with interpretation ambiguities.

The **baseline** for (sentence-wide) co-occurrence significance is to assume that higher frequency means higher significance. In order to relativize a high co-occurrence frequency of A with B with the individual frequencies of the two words, it is possible to compute the **Dice coefficient** [25] as an interpretation of the observed variables.

Assuming that the probability of the occurrence of a word in a sentence $p(A)$ can be approximated by the expression n_A/n then the probability of the co-occurrence of two words A and B in a random corpus should equal $p(A) \cdot p(B) = \frac{n_A \cdot n_B}{n^2}$. In the **Mutual Information** measure [13] this probability is compared to the conditional probability $p(A, B) = \frac{n_{AB}}{n}$ that can be derived from the observed data:

$$sig_{MI}(A, B) = \log_2 \frac{n \cdot n_{AB}}{n_A \cdot n_B} \tag{1}$$

Aware of the problems with MI, especially regarding its preference of low-frequent words, lexicographers modified it (**Lexicographers Mutual Information**) by an additional multiplication with the co-occurrence frequency [6]:

$$sig_{LMI}(A, B) = n_{AB} \log_2 \frac{n \cdot n_{AB}}{n_A \cdot n_B} \tag{2}$$

The **log-likelihood test** [14] uses the generalized likelihood ratio λ to compare two parametrized (binomial in this case) distributions with each other. The first set consists of parameters as expected from the independence assumption. The second derives it's parameters from the observed frequencies. Taking $-2log\lambda$ of the ratio, i.e. the probability of the the observed values, transforms it into a significance value, which is χ^2 distributed, so that the respective thresholds can be used. For example, one degree of freedom and a confidence level of 0.025 means that any value above 5.02 is significant with an error probability of 2.5%. To avoid problems with numerically too extreme probability values [5], it is possible to use the following equivalent and explicit but lengthy form that represents the ratio λ:

$$\lambda = \begin{bmatrix} n \log n - n_A \log n_A - n_B \log n_B + n_{AB} \log n_{AB} \\ + (n - n_A - n_B + n_{AB}) \cdot \log (n - n_A - n_B + n_{AB}) \\ + (n_A - n_{AB}) \log (n_A - n_{AB}) + (n_B - n_{AB}) \log (n_B - n_{AB}) \\ - (n - n_A) \log (n - n_A) - (n - n_B) \log (n - n_B) \end{bmatrix} \tag{3}$$

The significance is then computed as follows:

$$sig(A, B)_{lgl} = -2 \log \lambda \tag{4}$$

This test is only one-sided, in that it does not distinguish significant co-occurrence from significant non-co-occurrence. To amend this, a second significance can be defined:

$$sig(A,B)_{lgl2} = \begin{matrix} -2\log\lambda & \text{if } n_{AB} < \frac{n_A \cdot n_B}{n} \\ 2\log\lambda & \text{else} \end{matrix} \tag{5}$$

If the frequency of most words is much smaller than the corpus size, the Poisson distribution is a good approximation of the binomial distribution, which leads to the **Poisson significance measure** [16]. Using it instead of the binomial distribution results in a formula that contains the term $\ln n_{AB}!$ which is hard to handle numerically for larger n_{AB}. However, in such cases it is possible to use approximations, such as Stirling's formula, which (with $\lambda = \frac{n_A \cdot n_B}{n}$) results in the following explicit form:

$$sig_{ps1}(A,B) \approx n_{AB}\left(\ln n_{AB} - \ln\lambda - 1\right) + \tfrac{1}{2}\ln 2\pi n_{AB} + \lambda \tag{6}$$

Another presumably acceptable [17] approximation $\ln k! = k\ln k - k + 1$ which can be further simplified to $\ln k! = k\ln k$, results in the following significance measure:

$$sig_{ps2}(A,B) \approx n_{AB}\left(\ln n_{AB} - \ln\lambda - 1\right) \tag{7}$$

However, this simplification introduces a systematic error which results in an increasing positive discrepancy for larger n_{AB}. The effect is that this approximation systematically overrates larger co-occurrence frequencies over small ones, which also explains the varying performance in the evaluations below.

The **z-score** and the **t-score** (from the t-test) are two commonly used measures [5]. The z-score divides the difference between the expected and the observed value by the expected value $\frac{n_A \cdot n_B}{n^2}$:

$$sig(A,B)_{z-sc} = \frac{n_{AB} - \frac{n_A \cdot n_B}{n^2}}{\sqrt{\frac{n_A \cdot n_B}{n^2}}} \tag{8}$$

The t-score, applied to this task according to the 'standard way' [26] differs from the z-score only in dividing by the observed value n_{AB} instead of the expected one:

$$sig(A,B)_{t-sc} = \frac{n_{AB} - \frac{n_A \cdot n_B}{n^2}}{\sqrt{n_{AB}}} \tag{9}$$

Except for the Dice coefficient most measures produce values which are comparable to each other only if one of the two words is the same (i.e. in the local ranking case). In particular, it does not make sense to compare $sig(A,B)_{lgl} = 50$ with $sig(C,D)_{lgl} = 10$, because even though the 50 is numerically larger, the corresponding word frequencies, for example $n_A = 1000$ and $n_B = 1000$, might make it less important than the 10 with $n_A = 10$ and $n_B = 10$.

2.2 Similarity Measures

The similarity measures included in the evaluation are standard measures such as the cosine or euclidian distance and can be applied in different ways. It is possible to ignore the significance values computed by the significance measures above by transforming all vectors into binary vectors and then computing the cosine, for example. Alternatively, it is possible to keep the values and use the same measure. The following listing gives the similarity measures used to compare two vectors in the evaluation below (a more detailed listing including formulae can be found in related word, for example [27]):

- baseline (base) - the number of matching non-zero elements in both vectors
- overlap (over) - baseline divided by the minimum of non-zero elements in both vectors
- Dice (Dice) - baseline multiplied by 2 and divided by the amount of non-zero elements in both vectors
- binary cosine (cbin) - angle between the binary versions of both vectors
- cosine (cos) - angle between both vectors
- city block metric (L1) - the sum of the pairwise absolute differences between each value of both vectors
- euclidian distance (L2) - the square root of the pairwise squared differences between each value of both vectors
- Jensen-Shannon divergence (JS) - transforms both vectors into probability distributions, builds a mean distribution and measures the mean divergence of both original distributions to the mean distribution

Note that the JS divergence is defined only when applied on the plain frequency counts of co-occurrence, not on interpreted values. Technically however, any vector can be transformed into a probability distribution, but other than for frequency counts, the result is meaningless.

2.3 Computing Similarity

In order to compute the similarity between all words it is not necessary to compare each word with each other or to reduce the dimensionality of the entire vector space. Obviously, only words that are co-occurrences of co-occurrences of the input word are candidates to share any co-occurrences with it. Due to the power-law distribution of word frequency, for most words this candidate list is short (on the order of less than 100 words). In any case, but especially for the remaining very frequent words it is possible to restrict the search for candidates to use only the most significant co-occurrences to find new candidates.

This results in an algorithm with two approximation parameters which make the complexity of the entire algorithm linear, instead of quadratic. This is combined with programming the vector representations in a way that a vector uses only as much memory as there are non-zero entries in it. Additionally to the linear time-complexity of the algorithm the resulting constant memory requirement also enables computing similarity on arbitrarily large corpora without the need for costly hardware.

In the experiments reported here each word was compared with a maximum of 10 000 other words. Additionally, the maximal amount of values to be compared ordered by decreasing significance was restricted to 200. These approximations affected only 2% of all comparisons but decreased run-time significantly. However, these approximations skew the binary vector similarity measures: Since especially the baseline counts only matching non-zero values it should produce the exact same results irrespective of the underlying co-occurrence significance measure. Yet, since some values in the vectors are ignored depending on the particular co-occurrence measure, the performance varies, as shown below.

3 Evaluation

The experimental setup comprises a corpus and a gold standard and is repeated for two languages, English and German. The gold standards used are, respectively, WordNet [28] and GermaNet [29].

First, for each co-occurrence measure all sentence-wide co-occurrences (including insignificant ones) on the raw occurrences of words without any POS-tagging or other preprocessing are computed (apart from basic tokenization and sentence splitting). Then, for each word and for each measure the ranking of most similar words is obtained, which makes a total of $11 + 11 \cdot 8$ result sets. Then each result set is cut to contain only words that are in the gold standard and among the 100 000 most frequent words. Additionally, the ranking for each remaining word is cut to contain 100 words. The experiment was repeated for the BNC with WordNet and for a German subcorpus of the 'Wortschatz Projekt' with GermaNet. For English this leaves 35 966 input words with 100 output words each and for the German subcorpus 21 686 words. In the gold standards only those words were counted, which occured at least once in the corresponding corpus, i.e. 57 990 out of 146 212 for WordNet and 40 703 out of 52 620 for GermaNet. On average for each valid input word 59.9 relevant output words are found according to WordNet and 33.3 according to GermaNet. In both cases roughly 83% are cohyponyms.

The evaluation consists of evaluating the average quality of the ranking produced for each word. For this purpose, any word that stands in any relation with the input word in the gold standard is counted as relevant, similarly to relevant and irrelevant documents in IR. Both semantic nets were modified so that they also contain the cohyponymy relation (assuming that words sharing a direct hyperonym are cohyponyms). There are some problematic issues concerning this kind of evaluation, most importantly with the unknown upper bounds. On the one hand, an algorithm might compute many correct word pairs, which are all counted as wrong. The smaller the gold standard, the more severe is the effect on the evaluation. On the other hand, the gold standard might contain many annotated word pairs which are not observable in the corpus.

Nevertheless, used on such a large scale, this evaluation gives reliable (with respect to statistical significance) relative performances of the measures. The evaluation measures used are **mean average precision** (MAP) and **precision**. Precision is defined as the number of relevant words found in the ranking, divided by the length of the ranking. It is possible to measure only top 5 words, for

Table 1. Precision for 5 most significant or similar words in % for BNC measured on WordNet for all measure combinations based on 35 966 test words. Word pairs in any relation are counted as relevant. Two groups of measure combinations that do not differ significantly are emphasized.

	base	Dice	MI	LMI	t-sc	z-sc	lgl	lgl2	ps1	ps2
only	1.95	*8.35*	6.35	6.82	1.47	*8.08*	*7.79*	*7.79*	*8.13*	*8.26*
base	5.37	7.95	6.09	7.77	4.99	7.14	**8.79**	**8.78**	**8.65**	10.29
over	5.37	7.56	5.90	5.27	4.80	5.89	5.71	5.50	6.27	7.84
Dice	5.43	8.12	6.12	7.89	5.06	7.35	**8.98**	**8.98**	**8.82**	10.37
cos	3.29	8.09	6.30	8.49	5.63	6.90	7.96	8.80	8.82	9.16
cbin	5.44	8.12	6.10	7.93	5.09	7.29	**8.96**	**8.94**	**8.77**	10.17
L1	5.88	3.74	4.41	6.23	5.97	3.67	5.30	5.53	5.32	4.23
L2	5.70	3.84	3.93	7.07	5.93	3.52	6.36	7.05	6.17	6.03
JS	5.68	3.80	3.37	4.81	5.49	3.52	4.21	4.16	3.36	3.54

example, so that the expected performance in a thesaurus creation scenario is reflected more closely. MAP is defined as the sum of inverse ranks divided by the minimum of relevant words and ranking length and thus represents a combination of precision, recall and ranking quality. If 2 out of 40 relevant words were found at ranking positions 3 and 6 and the algorithm returned 100 words, then MAP in percent in this case is $\frac{\frac{1}{3}+\frac{2}{6}}{min(40,100)} \cdot 100 = 1.6\%$.

3.1 Results

Unsuprisingly, the values in Table 1 and 2 differ and are generally very low. Therefore the following discussion is based on the results of Scheffé's test which is an ANOVA post-hoc test to discover possible interactions in a group of means. Unfortunately, this necessary test has not yet been applied in related discussions. This test produces statements about whether a precision of 8.14% (*ps2_only*) and 7.12% (*MI_only*) differ significantly with an error probability of 0.0097 ($alpha = 0.05$). Usually, the pairwise t-test is applied instead; incorrectly, because the t-test is not able to correctly distinguish groups of possibly related test instances.

When **comparing the performances of co-occurrence significance measures**, Scheffé's test finds three groups of measures. The largest group (group significance 0.247 with $\alpha = 0.05$) includes Dice, z-score, both poisson variants and both log-likelihood variants. The second group is composed of both mutual information variants (group significance 0.807), whereas the remaining group comprises the t-score and the baseline (group significance 0.664). However, when these co-occurrence frequency interpretations are used to compute word similarity, the similarity computations based on the likelihood (and poisson) co-occurrence measures perform significantly better than any other. Surprisingly, the second (unprecise) poisson approximation *ps2* gives the best results and is found to significantly differ from all other measures. Apparently the systematic overrating of high co-occurrence counts helps, if not taken directly (as in the case of the baseline).

The following example illustrates the differences: The input word A ($n_A = 1000$) in a one million sentences corpus co-occurs 100 times with B ($n_B = 500$) and 150 times with C ($n_C = 2000$). According to the Dice coefficient $sig(A, B)_{Dice} = 0.133$ and $sig(A, C)_{Dice} = 0.1$, which means that the less frequent B is a more significant co-occurrence than C. Contrary to that, the second poisson approximation ($sig(A, B)_{ps2} = 429$ and $sig(A, C)_{ps2} = 497$) results in an inversed ranking. The reason why this has adverse effects on similarity computations is sparsity. The less frequent a word in a co-occurrence vector, the less probable it makes a match with a co-occurrence vector of another word.

The sole difference between the two log-likelihood variants as described above is that the second method differentiates between significant inhibition and attraction. The only effect this seems to have is a slight but statistically insignificant improvement of the subsequent similarity rankings.

As expected, the overrating of infrequent words renders the rankings of the mutual information measure nearly useless compared to the other measures. While the lexicographer's modification indeed helps, it does not help enough to produce significantly better results.

Table 2. Precision for 5 most significant or similar words in % for the German subcorpus measured on GermaNet for all measure combinations based on 21 686 test words. Word pairs in any relation are counted as relevant. Two groups of measure combinations that do not differ significantly are emphasized.

	base	Dice	MI	LMI	t-sc	z-sc	lgl	lgl2	ps1	ps2
only	3.47	*8.59*	7.12	6.90	2.84	*8.48*	*7.71*	*7.71*	*8.06*	*8.14*
base	7.44	10.98	8.78	10.40	6.92	9.18	**12.79**	**12.87**	**12.11**	14.63
over	7.25	10.88	8.77	8.06	6.89	8.67	9.21	9.17	9.71	11.18
Dice	7.55	11.24	8.92	10.59	7.03	9.57	**13.17**	**13.24**	**12.45**	14.70
cos	7.99	11.18	9.26	11.86	8.33	9.05	11.27	11.87	11.36	12.33
cbin	7.59	11.25	8.92	10.68	7.08	9.57	**13.15**	**13.21**	**12.36**	14.28
L1	8.64	5.92	7.02	11.61	9.12	6.54	8.68	8.90	7.85	6.74
L2	8.22	6.46	6.61	10.64	8.75	6.15	9.74	10.23	8.81	8.65
JS	7.98	6.82	5.88	7.69	7.92	6.26	6.49	6.53	5.46	5.19

The baseline - ranking according to co-occurrence frequency - is consistently outperformed by every measure except the t-score throughout both experiments. Hence, 'Yes, we can do better than frequency!', to answer the question formulated earlier [30] on a similar topic.

Comparing the similarity measures results in several surprising observations which are independent on which co-occurrence measure they are based on, except for the baseline and the t-score. First, it seems to be impossible to outperform the baseline. Second, Jensen-Shannon divergence is not the best measure, which apparently contradicts the results of other researchers. And third, the introduced approximations when computing similarity appear to be harmful if the co-occurrence measure has a poor performance.

The baseline disregards the computed significance values and only counts matching non-zero elements in the two vectors to be compared. The other measures either additionally take the amount of non-zero elements into account or weight the non-zero elements according to their relative values. Obviously, such additional information either degradates the results (for the overlap, for example) or does not seem to improve them (Dice). A manual examination of several examples revealed that the reason is a combination of data sparsity and Zipfs Law [31]. As could be expected, given an input word A and a set of words whose co-occurrence vectors have at least one matching element with the co-occurrence vector of A, the amount of matches is power-law distributed. That means that most words have one match, fewer two matches, etc. This also means that especially the words with the highest contextual similarity to A have rapidly falling amounts of matches. The first might have 200 matches, the next only 167, then the next only 150, etc.

As mentioned previously, the Jensen-Shannon divergence is applicable only to frequency counts, because only they can be directly transformed into probabilities. Hence, in the results tables only the results for JS applied on the baseline co-occurrence measure are meaningful. Out of all measures applied on the baseline co-occurrence data, the JS is indeed among the best measures, although not significantly. However, using a proper co-occurrence significance measure such as the second poisson approximation and then computing similarity using for example the Dice measure consistently produces approximately twice as good similarity rankings in both experiments.

In fact, these results suggest that any similarity measure based on pure frequency counts is at a disadvantage against comparing interpreted vectors. While it is unclear, whether for example the generalization to fewer concepts in LSI might alleviate the problem, it might also be the case that information-loss reduces the performance even more. A more direct comparison of LSI to the methods here would be necessary.

Finally, exactly the same observations can be reproduced from the MAP values for both experiments. The only difference is that the MAP values are lower and differ more between the two languages. For example, the *ps2-base* measure combination has a MAP value of 2.58 for English and 4.66 for German. The larger difference between the language is apparently due to the differing sizes of the knowledge bases which means that for English a lower recall is achieved. WordNet is about twice as large as GermaNet. Hence under similar circumstances it is to be expected that the same algorithm misses twice as much for English.

Additionally to Scheffé's test, Pearson's correlation coefficient helps to quantify the similarities between the various rankings of each measure combination. In the entire matrix of possible measure combinations only two pairs of combinations did not differ: *lg_only* and *lg2_only* had equal results. The only difference between them is that in 4.7% cases the computed significance is negative (resulting in a different ranking). Given that this difference did not affect the retrieval quality at all suggests that only words or word pairs not annotated in the knowledge sources are affected. In fact, only very frequent function words are affected. Because using the *z-sc* and *ps2* significances to compute similarity yields such differing quality of similarity rankings the main differences between these rankings are probably located in a range which is not measurable using WordNet or GermaNet, i.e. again function words.

On average, Pearson's correlation coefficient between all measure combinations ranges from 0.2 to 0.6. The only exceptions achieving values as high as 0.7 or 0.8 are pairs of similarly motivated co-occurrence measures and equal similarity measures such as *lg_base* and *ps1_base* 0.8 or *lg_base* and *lg2_base* 0.96. Such pairs were also found to not differ significantly by Scheffé's test above. It is interesting that some measures that did not perform well such as the *t-sc*, *z-sc* or both Mutual information variants correlate stronger with the log-likelihood measures with values between 0.6 and 0.7 than among each other (for example 0.42 for *t-sc* and *z-sc*). Supporting the finding that using *ps2* for similarity computations significantly outperforms all other co-occurrence measures the Pearson coefficient to the most similar similarity rankings is only 0.79 to *lg_base*. Roughly speaking, according to the results of Scheffé's test, any correlation values below 0.8 entails significant difference with an error probability of less than 0.05.

4 Conclusions

The approximations used in the similarity computation algorithm introduced in this work were shown to have a strong positive effect on the time-complexity of computing similarity. It was also shown that the approximations are only harmful, if the underlying co-occurrence significance measure was ill-chosen.

The interactions between the included measures have been fleshed out and underpinned with sound statistical tests. There are strong indications that any similarity measure dependent on raw frequency counts such as probability distribution divergence measures can be easily outperformed by much simpler comparisons based on co-occurrence significance values instead of frequency counts.

Further research should examine the interactions of measure performance with other factors such as corpus size or word frequency. The evaluation method employed in this work can easily be used to measure the relative performance at computing specific relations. It can be expected, for example, that co-occurrence rankings contain more syntagmatic relations, whereas similarity rankings should be more paradigmatic (see also [2,27]. Additionally, a direct comparison of the effects of the various measures on using them for local rankings as in this work or modified versions for global rankings is necessary as well. Especially in order to explore the discrepancy between the reported performance figures for example for the t-score which was found to be the worst measure in this evaluation but one of the best in other evaluations [5] using global rankings.

References

1. Finch, S.P.: Finding Structure in Language. PhD thesis, University of Edinburgh, Edinburgh, Scotland, UK (1993)
2. Sahlgren, M.: The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces. PhD thesis, Swedish Intitute of Computer Science, Stockholm, Sweden (2006)
3. Smadja, F.: Retrieving collocations from text: Xtract. Computational Linguistics 19, 43–177 (1993)
4. Lin, D.: Extracting collocations from text corpora. In: Proceedings of the First Workshop on Computational Terminology (1998)
5. Evert, S.: The Statistics of Word Cooccurrences: Word Pairs and Collocations. PhD thesis, University of Stuttgart, Stuttgart, Germany (2004)
6. Kilgarriff, A., et al.: The sketch engine. In: Proceedings of Euralex, Lorient, France, pp. 105–116 (2004)
7. Riloff, E., Shepherd, J.: A corpus-based approach for building semantic lexicons. In: Cardie, C., Weischedel, R. (eds.) Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP, Somerset, NJ, USA, Association for Computational Linguistics (ACL 1997) pp. 117–124 (1997)
8. Roark, B., Charniak, E.: Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In: Proceedings of The 17th International Conference on Computational Linguistics (COLING/ACL), Montreal, Quebec, Canada, pp. 1110–1116 (1998)
9. Widdows, D.: Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In: Proceedings of the Human Language Technology Conference (HLT) of the NAACL, Edmonton, Canada, pp. 276–283 (2003)
10. Rohwer, R., Freitag, D.: Towards full automation of lexicon construction. In: Proceedings of Computational Lexical Semantics Workshop at the HLT/NAACL, Boston, MA, USA (2004)
11. Dumais, S.T.: Latent semantic indexing (LSI). In: Harman, D.K. (ed.) Overview of the Third Text Retrieval Conference (TREC), Gaithersburg, MD, USA, National Institute of Standards and Technology, pp. 219–230 (1995)
12. Grefenstette, G.: Explorations in Automatic Thesaurus Discovery. Kluwer Academic Press, Boston (1994)
13. Church, K.W., et al.: Using statistics in lexical analysis. In: Zernik, U. (ed.) Lexical Acquisition: Exploiting On-Line Resources to Build up a Lexicon, pp. 115–164. Lawrence Erlbaum, Hillsdale (1991)
14. Dunning, T.E.: Accurate methods for the statistics of surprise and coincidence. Computational Linguistics 19, 61–74 (1993)
15. Lee, L.: Measures of distributional similarity. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL), College Park, MD, USA, pp. 25–32 (1999)
16. Holtsberg, A., Willners, C.: Statistics for sentential co-occurrence. Working Papers 48, 135–148 (2001)
17. Quasthoff, U., Wolff, C.: The poisson collocation measure and its applications. In: Second International Workshop on Computational Approaches to Collocations, Vienna, Austria (2002)
18. Curran, J.R.: From Distributional to Semantic Similarity. PhD thesis, Institute for Communicating and Collaborative Systems, School of Informatics. University of Edinburgh, Edinburgh, Scotland, UK (2003)

19. Terra, E., Clarke, C.L.A.: Frequency estimates for statistical word similarity measures. In: Proceedings of the Human Language Technology Conference (HLT) of the NAACL, Edmonton, Canada, pp. 165–172 (2003)
20. Gale, W., Church, K.W., Yarowsky, D.: Work on statistical methods for word sense disambiguation. In: Intelligent Probabilistic Approaches to Natural Language. Fall Symposium Series, pp. 54–60 (1992)
21. Schütze, H.: Context space. In: Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language, pp. 113–120. AAAI Press, Menlo Park (1992)
22. Lin, D.: Automatic retrieval and clustering of similar words. In: Proceedings of The 17th International Conference on Computational Linguistics (COLING/ACL), pp. 768–774 (1998)
23. Weeds, J., Weir, D.: Co-occurrence retrieval: A flexible framework for lexical distributional similarity. Computational Linguistics, 439–475 (2005)
24. Weeds, J.: The reliability of a similarity measure. In: Proceedings of the 5th UK Special Interest Group for Computational Linguistics (CLUK), Manchester, UK (2005)
25. Smadja, F., McKeown, K.R., Hatzivassiloglou, V.: Translating collocations for bilingual lexicons: A statistical approach. Computational Linguistics 22, 1–38 (1996)
26. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
27. Bordag, S.: Elements of Knowledge-free and Unsupervised lexical acquisition. PhD thesis, Department of Natural Language Processing, University of Leipzig, Leipzig, Germany (2007)
28. Fellbaum, C.: A semantic network of English: The mother of all WordNets. Computers and the Humanities 32, 209–220 (1998)
29. Hamp, B., Feldweg, H.: GermaNet - a lexical-semantic net for German. In: Proceedings of workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications at the ACL, Madrid, Spain (1997)
30. Krenn, B., Evert, S.: Can we do better than frequency? a case study on extracting pp-verb collocations. In: Proceedings of the Workshop on Collocations at the ACL, Toulouse, France, pp. 39–46 (2001)
31. Zipf, G.K.: Human Behaviour and the Principle of Least-Effort. Cambridge MA edn. Addison-Wesley, Reading (1949)

Various Criteria of Collocation Cohesion in Internet: Comparison of Resolving Power[*]

Igor A. Bolshakov[1], Elena I. Bolshakova[2],
Alexey P. Kotlyarov[1], and Alexander Gelbukh[2]

[1] Center for Computing Research (CIC)
National Polytechnic Institute (IPN), Mexico City, Mexico
{igor,gelbukh}@cic.ipn.mx
[2] Moscow State Lomonosov University
Faculty of Computational Mathematics and Cybernetics, Moscow, Russia
bolsh@cs.msu.su, koterpillar@gmail.com

Abstract. For extracting collocations from the Internet, it is necessary to numerically estimate the cohesion between potential collocates. Mutual Information cohesion measure (MI) based on numbers of collocate occurring closely together (N_{12}) and apart (N_1, N_2) is well known, but the Web page statistics deprives MI of its statistical validity. We propose a family of different measures that depend on N_1, N_2 and N_{12} in a similar monotonic way and possess the scalability feature of MI. We apply the new criteria for a collection of N_1, N_2, and N_{12} obtained from AltaVista for links between a few tens of English nouns and several hundreds of their modifiers taken from Oxford Collocations Dictionary. The nounits own adjective pairs are true collocations and their measure values form one distribution. The nounalien adjective pairs are false collocations and their measure values form another distribution. The discriminating threshold is searched for to minimize the sum of probabilities for errors of two possible types. The resolving power of a criterion is equal to the minimum of the sum. The best criterion delivering minimum minimorum is found.

1 Introduction

During the two recent decades, the vital role of collocationsin any their definitionwas fully acknowledged in NLP. Thus great effort was made to develop methods of collocation extraction from texts and text corpora. As pilot works we can mention [3,6,17,18]. However, up to date we have no large and humanly verified collocation databases for any language, including English. The only good exception is Oxford Collocations Dictionary for Students of English (OCDSE) [11], but even in its electronic version it is oriented to human use rather than to NLP. So the development of the methods of collocation extraction continues [4,5,9,12,13,14,15,16,19].

[*] Work done under partial support of Mexican Government (CONACyT, SNI, CGEPI-IPN) and Russian Foundation of Fundamental Research (grant 06-01-00571).

The well-known numerical measure of collocate cohesion used to extract collocations from text corpora is Mutual Information [10]. It is based on the ratio $(S \cdot N_{12})/(N_1 \cdot N_2)$ that includes numbers of collocates occurring closely together (N_{12}) and apart (N_1 and N_2), as well as the corpus size S.

However, the corpora, even the largest ones, suffer from data scarceness. Meanwhile, Internet search engines are considered more and more frequently as a practically unlimited source of collocations [7,8]. The transition to the Web as a huge corpus forces to revise all statistical criteria, since only numbers of relevant Web pages can be obtained from the search engines. The same words entering a page are indistinguishable in the page statistics, being counted only once, and the same page is counted repeatedly for each word included. Hence, Mutual Information measure is deprived of its statistical status. Therefore it is worthwhile to consider other cohesion measures (hereafter, we name them merely criteria) that depend on N_1, N_2, and N_{12} – now measured in pages – in a similar monotonic manner and retain so-called scalability feature of MI. Scalability is preserving the numeric value of a function with proportional changes of all its numeric arguments. This feature is required to diminish influence of systematic and stochastic variations of Internet statistics, since in each search engine the numbers N_1, N_2, and N_{12} for already well-known words are growing nearly proportionally over time.

The criteria to be chosen should have the most possible resolving power. It means that they should distinguish in a better way whether a given collocate pair is a true collocation or merely a pair casually occurred together. We could estimate the resolving power by the sum of probabilities for errors of the following two types: when a criterion considers a true collocation as false or when it considers a false collocation as true. So our plan is as follows.

We select a family of plausible criteria and prove that they possess the scalability and monotony against N_1, N_2, and N_{12}. Then we get a large set of triples N_1, N_2, N_{12} from AltaVista for collocate pairs formed by 32 English nouns and 1964 modifiers (mainly adjectives) that are recorded for these nouns in OCDSE. We consider the pairs that link the nouns with their own modifiers as true collocations, while 'noun–an alien modifier' pairs are considered false collocations. Some modifiers are common for several nouns, thus introducing errors in the attribution of some pairs. However, we neglect these facts since they affect all the criteria in a similar way.

In our experiments, the criterions values for 'noun–an alien modifier' pairs form one distribution, while 'noun–its own modifier' pairs form another. For the true pairs, any criterion usually gives greater values. A threshold is searched that minimizes the sum of probabilities for errors of the two types: attributing a false collocate pair to true collocations or a true collocate pair to false collocations. Resolving power of a criterion is defined to be that minimum. The best criterion delivers minimum minimorum.

It is shown that the best criterion unites N_1 and N_2 in the so-called harmonic mean. However, the remaining criteria under comparison give rather close results.

2 Various Numerical Criteria of Word Cohesion

Let us take the words W_1 and W_2 in a text corpus as would-be collocates and consider their occurrences and co-occurrences at a short distance as random events. Then their co-occurrence should be considered significant if the relative frequency N_{12}/S (= empirical probability) of the co-occurrence is greater than the product of relative frequencies N_1/S and N_2/S for the collocates taken apart (S is the corpus size in words). Using logarithm, we have the criterion of word cohesion known as Mutual Information [10]:

$$MI_{12} = \log \frac{S \cdot N_{12}}{N_1 \cdot N_2} \tag{1}$$

MI has an important feature of scalability: if all its building blocks S, N_1, N_2, and N_{12} are multiplied by the same positive factor, MI retains its value.

In the Internet we cannot evaluate events directly by numbers of words, since only Web page counts are available. Of course, we can re-conceptualize MI with all N being counts of the pages with relevant words or word combination and with S as the amount of pages indexed by the search engine. However, now N/S is not the empirical probabilities of word occurrence. We only cherish the hope that the ratio N/S is monotonically connected with the corresponding empirical probability for word occurrence.

An additional headache with MI is the page total S. Its evaluation is a separate task, necessitating several Internet queries. The substitution of S by the number of pages for the most frequent word in the given language (it is always an auxiliary word) does help [2], but the immanent Internet trend of volume growth keeps this additional measurement necessary. In such a situation, we are free to consider several different criteria built from the same numbers except of S, which we strive to exclude from the game. The sought-for criteria should:

1. Depend only on N_1, N_2, and N_{12};
2. Depend on N_{12} in a monotonously increasing manner;
3. Depend on N_1 and N_2 in a monotonously decreasing manner;
4. Depend on N_1 and N_2 in the same way, since we have no reason to consider any collocate more influential;
5. Be scalable.

So we change the ratio under the logarithm in (1) into the ratio N_{12}/M_{12}, where M_{12} is a specific mean value for the N_1 and N_2:

$$M_{12} = F^{-1}\left(\frac{F(N_1) + F(N_2)}{2}\right) \tag{2}$$

In (2), $F()$ is a monotonous function, and $F^{-1}()$ is its inverse. The features 1, 2, and 4 are evidently satisfied. The monotonous increment of M_{12} with growth of N_1 or N_2 (feature 3) can be shown through differentiating M_{12} by its arguments N_1 or N_2. It is interesting that the increment is valid even for any monotonously decreasing $F()$.

Table 1. Various types of the mean value

$F(z)$	M_{12}	**Name of** M_{12}
$\log z$	$\sqrt{N_1 N_2}$	Mean geometric
$1/z$	$2N_1N_2/(N_1+N_2)$	Mean harmonic
$\sqrt{z}$	$((\sqrt{N}_1+\sqrt{N}_2)/2)^2$	Mean square root
z	$(N_1+N_2)/2$	Mean arithmetic
z^2	$\sqrt{((N_1^2+N_2^2)/2)}$	Mean quadratic

However, the feature of scalability is not immanent for all types of $F()$, so we take only the specific group: $F(x) = \log x$ or $F(x) = x^p$, where p is positive. For them the scalability can be proved easily. Within the selected group, M_{12} coincides with well-known mean values (cf. Table 1). When collocates occur only together, so that $N_1 = N_2 = N_{12}$, the ratio N_{12}/M_{12} for all $F()$ in the group is equal to its maximum value 1. If these words never meet each other as close neighbors ($N_{12} = 0$), the ratio reaches its minimum value 0. When both words occur with nearly the same frequency, N_{12}/M_{12} is equal to N_{12}/N_1, which is usually a very small quantity.

To investigate the statistics of N_{12}/M_{12} in a more convenient way, we select logarithmic scale for it, just as for MI in (1), with the logarithmic base equal to 2 and an additive constant 16. Thus the collocation cohesion measure takes the form

$$CC = 16 + \log_2 \frac{N_{12}}{M_{12}} \tag{3}$$

The M_{12} in (3) is taken from Table 1, where the third column contains the name of the corresponding criterion.

The transformations in (3) put the maximum value to 16, while zero on the scale now corresponds to $N_{12} \approx N_1/65000$ in the case of $N_1 \approx N_2$. Previous research [1,2] of the geometric criterion with rather vast Web statistics gives evidence that the overwhelming majority of CC values for true collocations are in the interval $(0 \ldots 16)$. The minimal CC value goes to $-\infty$ because of the logarithm, so we may formally replace it by a large negative constant. We take -16, since this value was never reached for any positive N_{12} in our previous experiments.

It should be emphasized that all these scaling tricks in no way affect the further results. They merely expand the relevant scale interval and thus make it convenient for visual representation.

3 Modifier Sets Taken for Evaluations

We take as collocate pairs English nouns with their modifiers – both adjectives and nouns in attributive use – from OCDSE. The nouns were picked up in a rather arbitrary manner, with preference to those with larger modifier sets (cf. Table 2). The convenience of modifiers is that in English they frequently come just before its noun in texts, thus forming bigrams. A deeper research for

Table 2. Selected nouns and sizes of their modifier sets

SN	Noun	MSet Size	SN	Noun	MSet Size
1	answer	44	17	effect	105
2	chance	43	18	enquiries	45
3	change	71	19	evidence	66
4	charge	48	20	example	52
5	comment	39	21	exercises	80
6	concept	45	22	expansion	44
7	conditions	49	23	experience	53
8	conversation	52	24	explanation	59
9	copy	61	25	expression	115
10	decision	40	26	eyes	119
11	demands	98	27	face	96
12	difference	53	28	facility	89
13	disease	39	29	fashion	61
14	distribution	58	30	feature	51
15	duty	48	31	flat	48
16	economy	42	32	flavor	50

distant modifier pairs and collocations of other types necessitates considering word interval between collocates, and this essentially tangles the problem of evaluations of collocate co-occurrences through Internet search engines [2]. For these 32 nouns, total amount of modifiers, including repeated ones, is 1964 (1302 without repetitions). The mean modifier group size equals 61.4, varying from 39 (for *comment* and *disease*) to 119 (for *eyes*). The second and the third ranks determined by the set sizes correspond to *expression* (115) and *effect* (105).

Some nouns (*conditions*, *demands*, *enquiries*, *exercises*, and *eyes*) were taken in plural form in the experiments, since they are used with the recorded modifier sets in plural more frequently than in singular.

We have limited the number of nouns to 32 units, since the total amount of queries to the Web grows approximately as a square of this number. Taking into account the well-known limitations of Internet search engines, on the one hand, and the general trend of statistics growth, on the other hand, we have coped with ca. 50,000 accesses to AltaVista within a week, but we could not afford a greater task.

4 On Calculation of Resolving Powers

Our method of evaluation of the resolving power for various criteria is as follows. Let n_i, $i = 1 \ldots 32$, be nouns under research, and $M_{own}(n_i)$ be the sets of its own modifiers m_p. The set $M_{alien}(n_i)$ of modifiers m_q that are alien to n_i can be expressed by the formula

$$M_{alien}(n_i) = \bigcup_{j=1\ldots32, j\neq i} M_{own}(n_j)$$

We consider our five criteria, performing the following steps for each of them:

1. Calculate CC values for all pairs (n_i, m_p) and all i, forming the first distribution D_1.
2. Calculate CC values for all pairs (n_i, m_q) and all i, forming the second distribution D_2. It frequently contains the value $-\infty$ that corresponds to the collocate pairs never meeting together in the Internet closely (zero N_{12} value).
3. Changing threshold T by small steps, calculate the probability P_1 of D_1 tail in the region lower that T (this is the error of the first type, attributing a true collocate pair to false collocations), and the probability P_2 of D_2 tail in the region greater that T (this is the error of the second type, attributing a false collocate pair to true collocations) – cf. Figure 1. The minimal value of the sum $P_1 + P_2$ is the resolving power RP of the given criteria.

The RP values are then compared to each other and the minimum minimorum found, thus delivering the best criterion (champion). Note that $M_{alien}(n_i)$ can include some members of $M_{own}(n_i)$. The intersection of the sets increases the overlay of the distributions, but it does not eliminate their difference. Since the overlays affect the criteria in the same manner, they cannot change the champion.

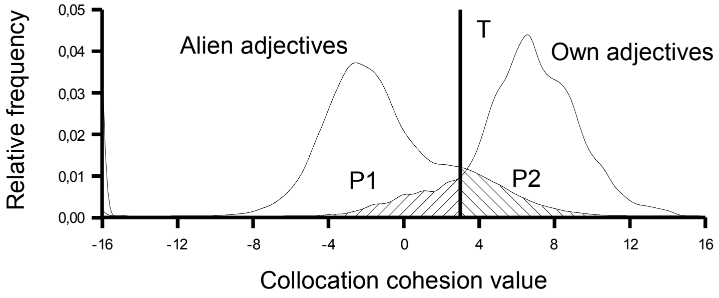


Fig. 1. Two distributions and the threshold

5 Experiment and Discussion of Various Criteria

The results of our calculation are given on Table 3. The best resolving power is delivered by the harmonic criterion with RP equal to 0.25 around the threshold 3.5. The worst is the quadratic criterion with RP equal to 0.30. We can see that the champion seems rather good, but the losers are not so far after. Moreover, shifts of thresholds in the intervals ± 2 centered at the minimums do not change the RP values significantly. All this means that collocation extraction from the Internet may be performed by any of these criteria with comparable results.

Our calculations also show that if CC for the champion is greater than 9.5, this pair is an obviously true collocation; and if it is lower than -3.5, the pair is an obviously false collocation.

Table 3. Resolving power of various criteria

Criterion Name	$F(z)$	Threshold for Minimum	Resolving Power
Geometric	$log z$	3.0	0.27
Harmonic	$1/z$	**3.5**	**0.25**
Square-root	$\sqrt{z}$	1.0	0.28
Arithmetic	z	1.5	0.29
Quadratic	z^2	1.5	0.30

The best criteria can be represented as (4)

$$CC = 16 + \log\left(\frac{N_{12}}{N_1 N_2}\frac{N_1 + N_2}{2}\right) \tag{4}$$

The comparison of (4) with (1) shows that the champion merely takes $2^{15}(N_1 + N_2)$ instead of S, with re-conceptualization of all numbers as measured in Web pages.

It is remarkable that the threshold 3.5 determined for the champion proved to be highly close to the threshold obtained in [2] for distinguishing true collocations from corresponding malapropos collocate pairs. To give a tip on the problem, let us consider a text with the malapropos phrase *travel about the* ***word***, where the intended ***world*** is erroneously replaced by the similar (paronymous) word ***word***. It is necessary to detect the pair *travel ...* ***word*** as false collocation and to propose the true collocation *travel ...* ***world*** as its correction. The detection of malapropos pairs and the search of their possible corrections can be done by means of cohesion measurement in the Internet, and appropriate experiments were carried out with representative sets of Russian malapropisms and with the aid of Yandex search engine.

Therefore, in [2] the close value of the threshold has been obtained for the definition of false collocations as malapropos pairs, for the different natural language, and for the different criterion (namely, the geometric one, cf. Table 3). This proves that the results of distinguishing correct collocations depend on natural language or criterion rather weakly.

6 Conclusions

We have proposed a family of numerical criteria to measure cohesion between words encountered in the Internet. All five criteria depend only on number of Web pages containing would-be collocates. The thresholds are found that minimize the sum of probabilities of errors of the two following types: considering a true collocation as false or considering a false collocation as true. The minimum is called resolving power RP of the given criteria. The best criterion delivers minimal RP among the peers. Its formula includes so-called harmonic mean for numbers of pages with collocate occurrences considered together or apart.

However, the remaining four criteria give comparable results. Therefore, each criterion among the considered ones may be taken for collocation extraction from the Internet with nearly the same results. Further search of better criteria seems ineffective. The proposed criteria are applicable to different problems of computational linguistics, among them malapropism detection and computer-aided acquisition of collocations from the Internet.

References

1. Bolshakov, I.A., Bolshakova, E.I.: Measurements of Lexico-Syntactic Cohesion by means of Internet. In: Gelbukh, A., de Albornoz, Á., Terashima-Marín, H. (eds.) MICAI 2005. LNCS (LNAI), vol. 3789, pp. 790–799. Springer, Heidelberg (2005)
2. Bolshakova, E.I., Bolshakov, I.A., Kotlyarov, A.P.: Experiments in Detection and Correction of Russian Malapropisms by means of the Web. International Journal on Information Theories & Applications 12(2), 141–149 (2005)
3. Church, K., Hanks, P.: Word association norms, mutual information, and lexicography. Computational Linguistics 16(1), 22–29 (1990)
4. Evert, S., Krenn, B.: Methods for the qualitative evaluation of lexical association measures. In: Proc. 39th Meeting of the ACL 2001, pp. 188–195 (2001)
5. Wu, H., Zhou, M.: Synonymous Collocation Extraction Using Translation Information, `http://acl.ldc.upenn.edu/P/P03/P03-1016.pdf`
6. Ikehara, S., Shirai, S., Uchino, H.: A statistical method for extracting uninterrupted and interrupted collocations from very large corpora. In: Proc. COLING 1996 Conference, pp. 574–579 (1996)
7. Keller, F., Lapata, M.: Using the Web to Obtain Frequencies for Unseen Bigram. Computational linguistics 29(3), 459–484 (2003)
8. Kilgarriff, A., Grefenstette, G.: Introduction to the Special Issue on the Web as Corpus. Computational linguistics 29(3), 333–347 (2003)
9. Krenn, B., Evert, S.: Can we do better than frequency? A case study on extracting pp-verb collocations. In: Proc. ACL Workshop on Collocations (2001)
10. Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
11. Oxford Collocations Dictionary for Students of English. Oxford University Press (2003)
12. Pearce, D.: Synonymy in collocation extraction. In: Proc. Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations. NAACL 2001, Pittsburgh, PA (2001), `http://citeseer.ist.psu.edu/pearce01synonymy.html`
13. Xu, R., Lu, Q.: Improving collocation extraction by using syntactic patterns. In: Proc. IEEE Int. Conf. Natural Language Processing and Knowledge Engineering, IEEE NLP-KE apos.05, pp. 52–57 (2005)
14. Seretan, V., Wehrli, E.: Accurate collocation extraction using a multilingual parser. In: Proc. 21st Int. Conf. Computational Linguistics and 44th Annual Meeting of the ACL, Sydney, Australia, pp. 953–960 (2006)
15. Seretan, V., Wehrli, E.: Multilingual collocation extraction: Issues and solutions. In: Proc. Workshop on Multilingual Language Resources and Interoperability, Sydney, Australia, pp. 40–49 (2006)
16. Seretan, V., Nerima, L., Wehrli, E.: A tool for multi-word collocation extraction and visualization in multilingual corpora. In: Proc. 11th EURALEX International Congress EURALEX 2004, Lorient, France, pp. 755–766 (2004)

17. Smadja, F.: Retreiving Collocations from text: Xtract. Computational Linguistics 19(1), 143–177 (1990)
18. Smadja, F.A., McKeown, K.R.: Automatically extracting and representing collocations for language generation. In: Proc. 28th Meeting of the ACL, pp. 252–259 (1990)
19. Wermter, J., Hahn, U.: Collocation Extraction Based on Modifiability Statistics. In: Proc. 20th Int. Conf. Computational Linguistics COLING 2004, pp. 980–986 (2004)

Why Don't Romanians Have a Five O'clock Tea, Nor Halloween, But Have a Kind of Valentines Day?

Corina Forăscu

University Al.I. Cuza of Iaşi, Faculty of Computer Science
Research Institute for Artificial Intelligence, Romanian Academy
16, Gen. Berthelot, Iaşi – 700483, Romania
corinfor@info.uaic.ro

Abstract. Recently the focus on temporal information in NLP applications has increased. Based on general temporal theories, annotations and standards, the paper presents the steps performed towards obtaining a parallel English-Romanian corpus, with the temporal information marked in both languages. The automatic import from English to Romanian of the TimeML markup has a success rate of 96.53%. The paper analyzes the main situations that appeared during the automatic import: perfect or impossible transfer, transfer with amendments or for the language specific phenomena. This corpus study permits to decide how import techniques can be used on the temporal domain.

1 Introduction

The temporal information is expressed in natural language through:

- Time-denoting temporal expressions – references to a calendar or clock system, expressed by NPs, PPs, or AdvPs, as in *Friday; yesterday; the previous month*.
- Event-denoting temporal expressions – explicit/implicit/vague references to an event; syntactically they are realized through:
  - sentences – more precisely their syntactic head, the main verb, as in *She <u>flew</u> as the first ever co-pilot.*
  - noun phrases, as in *She followed a normal <u>progression</u> within NASA.*
  - adjectives, predicative clauses or prepositional phrases, as in : *Many experts thought was once <u>invincible</u>.*

Recent work in document analysis started focusing on the temporal information in documents, mainly for their use in many practical Natural Language Processing (NLP) applications such as:.

- linguistic investigation, lexicon induction, and translation using very large annotated corpora;
- question answering (questions like “when”, “how often” or “how long”);
- information extraction or information retrieval;
- machine translation (translated and normalized temporal references; mappings between different behavior of tenses from language to language);
- discourse processing: temporal structure of discourse and summarization (temporally ordered information, biographic summaries).

The paper studies how well general temporal theories and annotation schema, developed mainly for English, can be applied to other languages – with emphasis on Romanian. A preliminary study, presented in this paper, shows promising results.

In order to have linguistic evidence of how temporal information is really used in Romanian, as source of evidence to inform and substantiate the theory, we used the TimeML 1.2. annotation standard [26] together with the TimeBank 1.2. corpus [19], an English news corpus manually annotated and widely used in the temporal community. The manual temporal annotation is very time consuming, incomplete, expensive [18] and error-prone, hence it would be useful to use some help or back-up from the same annotation applied to a parallel text. Previous experiments of manual temporal annotation on Romanian texts [9] leaded to the same conclusions[1].

Section 2 gives a brief state of the art in the field of temporal annotations and information in NL. In section 3 the TimeML standard and the TimeBank corpus are briefly presented. The next section details the work toward obtaining the parallel English-Romanian corpus, the pre-processing, alignments and annotation import performed on it, as well as an analysis of the main situations that appeared during the automatic import. As the paper presents "work-in-progress", the last section presents the conclusions and discusses future plans with regard to the corpus in order to see how temporal linguistic theories can be applied to Romanian, and applications to be developed by using it.

2 Time in Natural Language

The early work in the field of temporal information is based on Allen's 13 temporal binary relations between time intervals [1], and has used meaning representations augmented with temporal variables: in [20], the verb tenses are classified according to the ordering of three parameters: the points of speech, of the event and of reference.

The Message Understanding Conference, MUC-7[2] of 1998 has fostered the work in the field of temporal annotation. Main activities connected to temporal information and different types of temporal annotation schemes have been developed since then [15]. The most used annotation schemes are TIMEX2 [7] and TimeML [21]. TIMEX2 is a component technology in ACE[3], conceived to fill the temporal attributes for extracted relations and events. TimeML is more complex and it treats unitarily the temporal aspects of texts, hence it is useful in much more applications.

The TimeML standard integrates together two annotation schemes: TIMEX2 and Sheffield STAG ([22] – as a first complete mention of STAG; continuously improved), a fine-grained annotation scheme capturing events, times and temporal relations between them, as well as other emerging work [14].

Corpora with the temporal information marked, as well as temporal taggers have been created mainly for English, but French, German, Spanish, Chinese, Arabic and Korean[4] start to become prominent languages in the field.

[1] We thank our reviewers for their suggestions.
[2] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/
[3] http://www.nist.gov/speech/tests/ace/index.htm
[4] http://complingone.georgetown.edu/~linguist/

Regarding the Semantic Web, some significant efforts have been invested in order to develop ontologies of time, for expressing the temporal content of web pages and temporal properties of web pages and web services (SUMO[5], CYC[6] among others). DAML-Time ontology [10] is a collaborative effort towards standardizing the basic topological temporal relations on instants and intervals, measures of duration, clock and calendar units, months and years, time and duration stamps, including temporal aggregates (*for the last four years*), deictic time (*now*) and vague temporal concepts *(recently, soon*).

TIMEX2 scheme is compatible with the KSL-Time ontology[7], while TimeML is mapped onto the DAML-Time Ontology [11], hence advanced inferential capabilities based on information extracted from text are better supported.

3 TimeML 1.2.1. and TimeBank 1.2.

The TimeML standard has been developed for the purpose of automatically extracting information about the event-structure of narrative texts, and has been applied mainly to English news data. The mark-up language consists of a collection of tags intended to explicitly outline the information about the events reported in a given text, as well as about their temporal relations.

The TimeML metadata standard marks:

- Events through the tags:
  - EVENT: it indicates situations that happen or occur, states or circumstances in which something obtains or holds true: *Female pilots were held up until now by the lack of piloting opportunities for them in the military.*
  - MAKEINSTANCE: it marks how many different instances or realizations a given event has; the tag also carries the tense and aspect of the verb-denoted event: *But they still have catching up to do two hundred and thirty four Americans have flown in space, only twenty six of them women.*
- Temporal anchoring of events through the tags:
  - TIMEX3: it marks: times of a day, dates – calendar dates or ranges, durations: *25 October*; *two days*; *nowadays.*
  - SIGNAL: it marks function words that indicate how temporal objects are to be related to each other: *before, after, until.*
- Links between events and/or timexes through the tags:
  - TLINK – Temporal Link – indicates 13 types of temporal relations between two temporal elements (event-event, event-timex).
  - ALINK – Aspectual Link (of type Initiation, Culmination, Termination, Continuation) – marks the relationship between an aspectual event and its argument event.
  - SLINK – Subordination Link (of type Modal, Factive, Evidential, Negative) – marks contexts introducing relations between two events.

[5] http://ontology.teknowledge.com/rsigma/arch.html#Temporal
[6] http://www.cyc.com/cycdoc/vocab/time-vocab.html
[7] http://www.ksl.stanford.edu/ontologies/time/

The creation of the TimeBank corpus started in 2002 during the TERQAS[8] workshop, and it should be considered preliminary. [4] proves that the corpus still needs improvements and reviews. The dimension of the corpus (4715 sentences with 10586 unique lexical units, from a total of 61042 lexical units) might be too small for robust statistical learning and the annotation inconsistencies (incomplete or inconsistent temporal or subordination links, perfectible event classification, incomplete annotation of the tense and aspect for some event) require corrections. Now it consists of 183 news report documents, with XML markups for document format and structure information, sentence boundary information, and named entity recognition (ENAMEX, NUMEX, CARDINAL from MUC-7). TimeBank 1.2. is temporally annotated according to the TimeML 1.2.1 standard. Some statistics automatically performed on the corpus are shown in table 1.

Table 1. Statistics on English TimeBank 1.2

TimeML tags	#
events	7935
instances	7940
Timexes	1414
Signals	688
alinks	265
slinks	2932
Tlinks	6418
TOTAL	**27592**

The corpus is distributed through LDC[9] [19] and it can be browsed online[10].

4 Towards a Parallel, Temporal Annotated Corpus

In the following subsections we detail the main steps towards the targeted corpus: translation, preprocessing, alignment, and annotation import. The encountered problems and their (possible) solutions, as well as the main situations we faced during the automatic import are discussed.

4.1 Corpus Translation

The TimeBank corpus was distributed for translation to two Master students in Computational Linguistics with strong background in English and Romanian philology and translation. Even if the translation is never perfect and it does not always reflect the accurate Romanian language and its specific phenomena, the solution was adopted because it was feasible with students (unfortunately not with

[8] Time and Event Recognition for Question Answering Systems - available at http://www.timeml.org/site/terqas/index.html

[9] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T08

[10] http://www.timeml.org/site/timebank/browser_1.2/index.php

professionals) and, as it will be shown, the automatic import can be a solution to have the temporal information marked.

As the next step is the alignment of the English and Romanian versions of the corpus, a minimal set of translation recommendations was elaborated, in order not only to ensure a literal translation - one which keeps as close as possible to the original version, but also to permit a best-possible word-alignment process. Some basic translation principles are the followings:

- The sentences are translated in a 1:1 correspondence, whenever the language permits it, so that the sentence-alignment is directly obtained through translation.
- The translation equivalents have as much as possible the same part-of-speech; when the English word has a Romanian cognate (en. *manually* – ro. *manual*), this is used in translation, and not its Romanian paraphrase (*de/cu/la mână – by hand*).
- All words are translated and stylistic variations are avoided, so as not to introduce words or expressions without an English equivalent.
- The tense of verbs is mapped onto its corresponding Romanian one, the modifications being accepted only on linguistic grounds, but not stylistic.
- The format of the dates, moments of day and numbers conforms to the norms of written Romanian.

A first automatic import showed that the translation need more improvements, mainly because more than a half of the articles didn't include in the Romanian version the header of the file, the place where the "creation time" of the document is mentioned. The absence of this part of the text does not permit to import the temporal links of all events related to the "creation time". Therefore a manual check performed on the parallel corpus allowed us to detect and correct also some other lacks and inconsistencies in the way the translators worked.

In the 4715 sentences (translation units) of the current version of the Romanian corpus there are 65375 lexical tokens, including punctuation marks, representing 12640 lexical types.

4.2 Preprocessing the English-Romanian TimeBank

In order to run the lexical aligner, the English and Romanian raw texts have to be preprocessed so as to obtain the corpus in the required format. Thus, the texts are tokenized, POS-tagged, lemmatized and chunked using the TTL[11] module [13]. This module assembles the bitext in an XML format similar to the XCES one [12]. Following, there is a brief description of the preprocessing operations:

1. The tokenization closely follows the MtSeg model [2], dealing also with multi-word expressions (*parte de vorbire - part of speech*) and clitic splitting (*arătat-o – showed it*) by using specific lists for every language.
2. POS tagging implements the TnT POS tagger [3], enriching it with some heuristics to determine the part-of-speech of an unknown word; only open-class words are considered because the grammatical categories of functional words are thought to be known for a given language.

[11] **T**okenizing, **T**agging and **L**emmatizing free running texts.

3. The lemmatization is a stochastic process which automatically learns lemmatization rules from a lexicon containing triples word form, lemma and POS tag. Based on these three steps, the alignment can be done only word-by-word.
4. Chunking: non-recursive chunks are recognised using a set of regular expressions defined over sequences of POS tags: noun phrases, adjectival phrases (*cea mai frumoasă – the most beautiful*), adverbial phrases, prepositional phrases (*în decembrie – in December*) and verb complexes (*vor pleca – will go*). The chunking permits to have n-m alignments.

4.3 Lexical Alignment

Because the COWAL combined word alignment software [24] is currently under major optimization, only YAWA, one of the two word aligners of the COWAL, was used. YAWA is a four stage lexical aligner[12] that uses bilingual translation lexicons [25] and phrase boundaries detection to align words of a given bitext. In each of the first three stages, YAWA adds new links to those already created in the previous steps, without deleting from the existing ones. The evaluation scores of the alignment stages described below are computed over the data in the Shared Task on Word Alignment [17], Romanian-English track organized at the ACL2005.

1. Content words alignment: the open-class words (nouns, verbs, adjectives and adverbs) are aligned using translation lexicons [25]. After this stage, YAWA has a high precision, but the recall is improved during the next steps: P = 94.08%, R = 34.99%, F = 51.00%.
2. Inside-Chunks alignment: after a chunk-to-chunk matching based on the first stage, YAWA uses simple empirical rules to align the words within the corresponding chunks; for example a Romanian noun aligned to an English one preceded by an English determiner will be also linked to the determiner (*fata – the girl*); a Romanian auxiliary verb followed by a participle will be linked to an English main verb (*am aflat – learned*) The evaluation after this step gives P = 89.90%, R = 53.90%, F = 67.40%.
3. Alignment in contiguous sequences of unaligned words [24]: using the POS-affinities of these unaligned words and their relative positions, YAWA attempts to heuristically match them; unaligned chunks surrounded by aligned chunks get probable phrase alignment.
4. Correction phase: the wrong links introduced mainly in stage 3 are now removed.

The current evaluation [26] of YAWA (P = 88.80%, R = 74.83%, F = 81.22%) shows a significant improvement over the accuracy reported in [17]. The COWAL combiner of YAWA and MEBA word aligners was rated the best out of 37 systems participating in the Shared Task [23], with the following evaluation scores: P = 87.17%, R = 70.25%, F = 77.80%. As YAWA has already achieved a very good accuracy it can be successfully used on its own.

The automatic alignment performed on 181 files (out of 183) in the TimeBank parallel corpus produced 91714 alignments out of which 25346 are NULL-alignments. In order to obtain an optimal transfer of the temporal annotations from the

[12] Currently, YAWA only supports Romanian to English lexical alignment.

English version onto the Romanian one, all the alignments were manually checked using MTKit [5]. Most of the wrong alignments are due to incorrect tokenization of some numbers and values, incorrect POS-tagging mainly for Romanian possessive pronouns, English negations (*no*, *n't*, *neither*) and English adjectives (*lower*, *smallest*).

4.4 Import of the Temporal Mark-Up

The translation of the English part of the TimeBank corpus followed the sentence XML structure, and hence it was possible to parse the English corpus and for every sentence XML tag, to extract its content and replace it with the Romanian translation. Due to the nature of the Romanian translations, within a sentence we can not assume that the word ordering in English is completely preserved into Romanian and also that English received a literal (almost word by word) translation into Romanian. Thus, we need to use the Romanian to English lexical alignment to transfer the XML markup from English to Romanian because, otherwise, we could obtain the Romanian translation in a shuffled form if the word order was not preserved. The transfer algorithm goes as follows:

Romanian	English
`<s>`	`<s>`
`<SIGNAL sid="s72">`**O dată ce**`</SIGNAL>`	`<SIGNAL sid="s72">`**Once**`</SIGNAL>`
`<ENAMEX TYPE="PERSON">`**colonelul**	`<ENAMEX TYPE="PERSON">`**Colonel**
Collins`</ENAMEX>`	**Collins**`</ENAMEX>`
a fost	**was**
`<EVENT class="I_ACTION" eid="e22"`	`<EVENT eid="e22" class="I_ACTION">`
aleasă`</EVENT>`	**picked**`</EVENT>`
ca	**as a**
`<EVENT class="STATE" eid="e52">`	`<ENAMEX TYPE="ORGANIZATION">`**NASA**
astronaut`</EVENT>`	`</ENAMEX>`
`<ENAMEX TYPE="ORGANIZATION">` **NASA**	`<EVENT eid="e52" class="STATE">`
`</ENAMEX>`	**astronaut**`</EVENT>`
`</s>`	`</s>`

Fig. 1. An example of the XML markup transfer from English to Romanian

For every pair of sentences ($\mathbf{S}_{ro}$; $\mathbf{S}_{en}$) from the TimeBank parallel corpus with the $\mathbf{T}_{en}$ English equivalent sentence ($\mathbf{T}_{en}$ is the same sentence – same raw text – as $\mathbf{S}_{en}$, with the exception that $\mathbf{T}_{en}$ has the XML structure that we want to transfer) do:

- construct a list **E** of pairs of English text fragments with sequences of English indexes from $\mathbf{S}_{en}$ and $\mathbf{T}_{en}$. Due to the fact that the tokenization of $\mathbf{S}_{en}$ is different from that of $\mathbf{T}_{en}$, the list **E** is needed in order to map English text fragments from $\mathbf{T}_{en}$ with sequences of indexes from $\mathbf{S}_{en}$ so as to be able to use the Romanian lexical alignments which exist relative to these indexes. For instance, looking at Figures 1 and 2: **E** = *{<"Once";1>, <"Colonel Collins";2,3>, <"was";4>, <"picked"; 5>, <"as";6>, <"a";7>, <"NASA";8>, <"astronaut";9>}*.

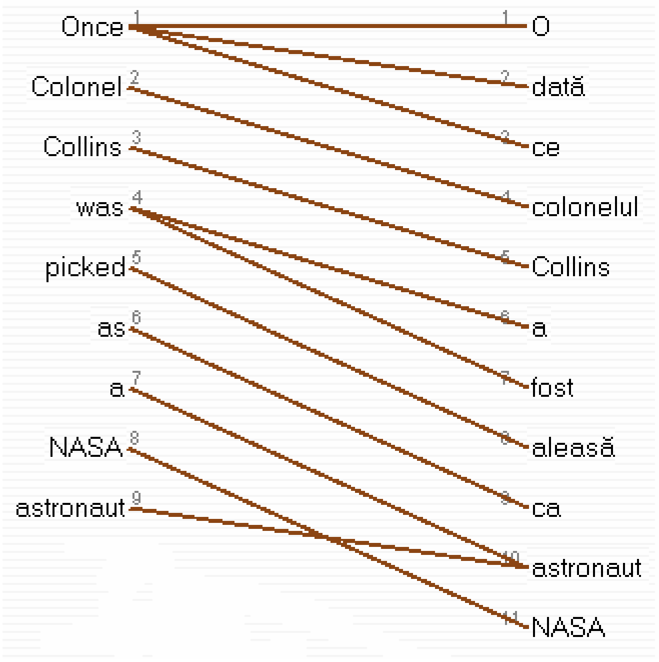


Fig. 2. The lexical alignment for the sentences in Figure 1

- add to every element of **E** the XML context in which that text fragment appeared. For instance, the first element of **E**, *<"Once";1>*, appears in the `s` and `SIGNAL` contexts and 3rd element, *<"was";4>* appears only in the `s` context. Thus the list **E** becomes **E** = *{<"Once";1; s,SIGNAL>, <"Colonel Collins";2,3; s,ENAMEX>, <"was";4; s>, <"picked"; 5; s,EVENT>, <"as";6; s>, <"a";7; s>, <"NASA";8; s,ENAMEX>, <"astronaut";9; s,EVENT>}*. For every tag, its attributes – if present – are stored.
- construct the list **RW** of Romanian words along with the transferred XML contexts using **E** and the lexical alignment between $\mathbf{S}_{ro}$ and $\mathbf{S}_{en}$. If a word in $\mathbf{S}_{ro}$ is not aligned, the top context for it, namely s, is considered. Using the example in Figures 1 and 2, **RW** = *{<"O dată ce";1,2,3; s,SIGNAL>, <"Colonel Collins";4,5; s,ENAMEX>, <"a fost";6,7; s>, <"aleasă"; 8; s,EVENT>, <"ca";9; s>, <"astronaut";10; s,EVENT>, <"NASA";11; s,ENAMEX> }*.
- construct the final list **R** of Romanian text fragments from **RW** by conflating adjacent elements of **RW** that appear in the same XML context. Output the list in XML format (Figure 1 - the result of XML markup transfer).

A TimeBank document can be seen as having three parts: the header, the text and the time and event descriptions (instances and links between temporal entities). The transfer procedure is designated for the header and the text parts only. The time and event descriptions make use of the EVENT, TIMEX3 and SIGNAL IDs from the first two parts (see MAKEINSTANCE, ALINK, TLINK and SLINK tags). For these

Table 2. Statistics on the Romanian TimeBank 1.2

TimeML tags	**#**	**% transferred**
events	7703	97.07
instances	7706	97.05
timexes	1356	95.89
signals	668	97.09
alinks	249	93.96
slinks	2831	96.55
tlinks	6122	95.38
TOTAL	**26635**	**96.53**

descriptions the transfer kept only those XML tags from the English version whose IDs belong to XML structures that have been transferred to Romanian. In Table 2 there is a statistic of the resulting Romanian TimeBank corpus in terms of all TimeML transferred markups.

4.5 Import Analysis

In order to reach one of the initial goals – to have an English-Romanian parallel corpus, temporally annotated in both languages, as a basis for further researches – the annotation transfer started to be evaluated by using the manually corrected markups in the parallel corpus. This work permits to analyze the situations of perfect transfer and compare them with those situations in which:

- the temporal annotation transfer has to be done with some amendments when the temporal constructions in the two languages are not similar but they can be transferred using special developed rules, or
- it has to deal with language specific phenomena, such as the treatment of clitics or the PRO-drop phenomenon, specific to Romanian but not to English,
- or the transfer can not be performed.

We performed a preliminary study, using five files of the corpus – about 3% of the corpus. In the followings we will not refer to the offline markups (MAKEINSTANCE, ALINK, TLINK and SLINK tags), because they are automatically imported only if the elements they are linked to are present in the text, e.g. a TLINK is imported into Romanian if the events, temporal expressions and/or signals it uses are already imported. The table 3 summarizes the four situations encountered during the annotation import.

The amendments that need to be done in the automatic import of the EVENT tag are due to the TimeML rule stating that in cases of phrases, the EVENT tag should mark only the head of the construction. This is the case for Romanian reflexive verbs (the reflexive pronoun was marked inside the EVENT tag: *(to) withdraw – (să) se retragă*), Romanian verbal collocations (<u>avea</u> permisiunea – <u>permit</u>), compound verb phrases (*să se* <u>*îndoiască*</u> – <u>*doubt*</u>).

The only language specific phenomenon that occurred in the files we used is the intercalation of an adverb/conjunction between the verbs forming a verb phrase: ***also***

Table 3. Situations in the automatic import

Transfer \ Tags	EVENT	TIMEX3	SIGNAL
Perfect	202	33	17
With amendments	23	3	-
Based on language specific phenomena	2	-	-
Impossible	2	-	1

<u>*said*</u> – <u>*au*</u> ***mai*** <u>*spus*</u>; *(he)* ***also*** <u>*criticised*</u> – <u>*a*</u> ***şi*** <u>*criticat*</u>, situations where the EVENT tags where automatically imported also on the auxiliary Romanian verb.

The situations when the transfer of events was impossible are due to missing translations (*forces that harbor ill intentions – forţe străine cu intenţii rele*), non-lexicalisations in Romanian (<u>*give*</u>$_1$ *the* <u>*view*</u>$_2$ – <u>*arată*</u>$_1$), or missing alignments (these situations were corrected).

With respect to the TIMEX3 tag there was a case of missing alignment: for the English temporal expression *some time* in the Romanian translation, *un timp mai lung*, the alignment didn't include the adjectival phrase *mai lung*. The other situations of amendments has to consider the wrong marking of the Romanian prepositions as part of TIMEX3, for example <u>*eight years*</u> *(war) – (războiul)* <u>*de*</u> <u>*opt ani*</u>.

The only impossible transfer of a SIGNAL tag is due to non-lexicalisation in Romanian: *on Tuesday – marţi*, where the preposition *on* is marked as a SIGNAL in the English corpus, but it is not present in the Romanian text.

The study permitted to identify temporal elements not (yet) marked in the English TimeBank 1.2. For the events we suggest 35 new elements: 4 of the REPORTING class (*say, said*), 11 belonging to the STATE class (*belongs, look, ceiling, staying, war, policies*), 15 OCCURENCEs (nouns: *missions, training, fight, (mediation) effort, demarcation, move*, as well as verbs: *supervising, leading, include*), and 5 from the I_STATE class (*like, think*). The main idea to propose these events is that each sentence expresses an event, even if not so well temporally anchored. We found two temporal expressions (TIMEX3 tag) for which the value is PAST_REF – meaning that the expressions do not have a specific value, but they can be normalised according to the extended ISO 8601 standard used in TimeML: *once, not that long ago*. The four SIGNALs that we found not-marked are most probably due to inevitable manual annotation mistakes: *several, time and again, after, on*. These observations are consistent with the conclusions of the TimeBank developers[13]: the corpus still needs improvements and reviews [4] especially with respect to: event classes, incomplete temporal markup and linking, incomplete subordinated linking.

5 Conclusions and Future Work

The research proves that the automatic import of the temporal annotations from English to other language (here – Romanian) is a worth doing enterprise with a very

[13] http://www.timeml.org/site/timebank/timebank.html

high success rate (in our experiments the transfer success rate was as high as 96.53%). The most important conclusion of the described work is that, as the manual annotation of the temporal expressions, events and their links is very time-consuming and expensive [9,18], the automatic transfer of annotations represents a solution, provided a parallel corpus involving the target language exists, the source language displays temporal annotation, and adequate processing tools are available.

This study opens the possibility to decide, based on corpus-evidence, how well the temporal theories can be applied to other languages, here with emphasis on Romanian. In particular, the grammatical category of "aspect" of Romanian verbs could be better defined.

The best methods developed until now – machine learning-based or rule-based – will be studied, in order to create or adapt a temporal tagger – such as TARSQI [26] - for Romanian, or even a language independent one.

The temporal annotated data together with time ontologies will be used to represent the temporal structure of the discourse and its possible relations with other discourse structures, such as, for example, Rhetorical Structure [16] or Veins Theory [6,8].

The cooperation with specialists in the NLP field will result in developing other specific applications, using various language and/or web resources: Reasoning with extracted temporal information, Temporal Summarization, Temporal Discourse Structure, Temporal Question-Answering, and Machine Translation.

References

1. Allen, J.F.: Towards a General Theory of Action and Time. Artificial Intelligence 23, 123–154 (1984)
2. Armstrong, S.: Multext: Multilingual Text Tools and Corpora. Lexikon und Text, 107–119 (1996)
3. Brants, T.: TnT – a statistical part-of-speech tagger. In: Proceedings of the 6th Applied NLP Conference, ANLP-2000, Seattle, WA, pp. 224–231 (2000)
4. Boguraev, B., Ando, R.: Analysis of TimeBank as a Resource for TimeML Parsing. In: Proceedings of LREC 2006, Genoa, Italy, pp. 71–76 (2006)
5. Ceauşu, A.: Integrated platform for Statistical Machine Translation system development (MTkit). Microsoft Imagine Cup (2005)
6. Cristea, D., Ide, N., Romary, L.: Veins Theory. An Approach to Global Cohesion and Coherence. In: Proceedings of COLING/ACL- 1998, Montreal, Canada, pp. 281–285 (1998)
7. Ferro, L., Gerber, L., Mani, I., Sundheim, B., Wilson, G.: TIDES 2005 Standard for the Annotation of Temporal Expressions (2005)
8. Forăscu, C., Pistol, I., Cristea, D.: Temporality in Relation with Discourse Structure. In: Proceedings of LREC-2006, Genoa, Italy, pp. 65–70 (2006) ISBN 2-9517408-2-4
9. Forăscu, C., Solomon, D.: Towards a Time Tagger for Romanian. In: Proceedings of the ESSLLI Student Session, Nancy, France (2004)
10. Hobbs, J.: Toward an Ontology for Time for the Semantic Web. In: Proceedings of the LREC 2002 Workshop Annotation Standards for Temporal Information in Natural Language, Las Palmas, Spain, pp. 28–35 (2002)
11. Hobbs, J., Pustejovsky, J.: Annotating and Reasoning about Time and Events. In: Proceedings of the AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning, Stanford, California (2003)

12. Ide, N., Bonhomme, P., Romary, L.: XCES: An XML-based Encoding Standard for Linguistic Corpora. In: Proceedings of the Second International Language Resources and Evaluation Conference, pp. 825–830 (2000)
13. Ion, R.: Word Sense Disambiguation Methods Applied to English and Romanian. (in Romanian) PhD thesis. Romanian Academy, Bucharest (2007)
14. Katz, G., Arosio, F.: The Annotation of Temporal Information in Natural Language Sentences. In: Proceedings of the ACL-2001 Workshop on Temporal and Spatial Information Processing, ACL-2001, Toulose, France, pp. 104–111 (2001)
15. Mani, I., Pustejovsky, J., Gaizauskas, R. (eds.): The Language of Time: A Reader. Oxford University Press, Oxford (2005)
16. Mann, W.C., Thompson, S.A.: Rhetorical structure theory: Description and construction of texts structures. In: Kempen, G. (ed.) Natural Language Generation, pp. 85–96. Martinus Nijhoff Publisher, Dordrecht (1987)
17. Martin, J., Mihalcea, R., Pedersen, T.: Word Alignment for Languages with Scarce Resources. In: Proceeding of the ACL2005 Workshop on Building and Using Parallel Corpora: Datadriven Machine Translation and Beyond. Ann Arbor, Michigan, pp. 65–74 (2005)
18. Pustejovsky, J., Belanger, L., Castaño, J., Gaizauskas, R., Hanks, P., Ingria, B., Katz, G., Radev, D., Rumshisky, A., Sanfilippo, A., Sauri, R., Setzer, A., Sundheim, B., Verhagen, M.: NRRC Summer Workshop on Temporal and Event Recognition for QA Systems (2002)
19. Pustejovsky, J., Verhagen, M., Sauri, R., Littman, J., Gaizauskas, R., Katz, G., Mani, I., Knippen, B., Setzer, A.: TimeBank 1.2. Linguistic Data Consortium (2006)
20. Reichenbach., H.: The tenses of verbs. In: Reichenbach, H. (ed.) Elements of Symbolic Logic, Section 51, pp. 287–298. Macmillan, New York (1947)
21. Sauri, R., Littman, J., Knippen, B., Gaizauskas, R., Setzer, A., Pustejovsky, J.: TimeML Annotation Guidelines, Version 1.2.1 (2006)
22. Setzer, A.: Temporal Information in Newswire Articles: an Annotation Scheme and Corpus Study. PhD dissertation. University of Sheffield (2001)
23. Tufiş, D., Ion, R., Ceauşu, A., Ştefănescu, D.: Combined Aligners. In: Proceedings of the ACL 2005 Workshop on Building and Using Parallel Corpora: Data-driven Machine Translation and Beyond, Ann Arbor, Michigan pp. 107–110 (2005)
24. Tufiş, D., Ion, R., Ceauşu, A., Ştefănescu, D.: Improved Lexical Alignment by Combining Multiple Reified Alignments. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006) Trento, Italy pp. 153–160 (2006)
25. Tufiş, D., Barbu, A.M.: Revealing translators knowledge: statistical methods in constructing practical translation lexicons for language and speech processing. International Journal of Speech Technology (5), 199–209 (2002)
26. Verhagen, M., Mani, I., Sauri, R., Littman, J., Knippen, R., Bae Jang, S., Rumshisky, A., Phillips, J., Pustejovsky, J.: Automating Temporal Annotation with TARSQI. In: Proceedings of the 43rd Annual Meeting of the ACL, Ann Arbor, Michigan, pp. 81–84 (2005)

SIGNUM: A Graph Algorithm for Terminology Extraction

Axel-Cyrille Ngonga Ngomo

University of Leipzig, Johannisgasse 26, Leipzig D-04103, Germany
ngonga@informatik.uni-leipzig.de
http://bis.uni-leipzig.de/AxelNgonga

Abstract. Terminology extraction is an essential step in several fields of natural language processing such as dictionary and ontology extraction. In this paper, we present a novel graph-based approach to terminology extraction. We use SIGNUM, a general purpose graph-based algorithm for binary clustering on directed weighted graphs generated using a metric for multi-word extraction. Our approach is totally knowledge-free and can thus be used on corpora written in any language. Furthermore it is unsupervised, making it suitable for use by non-experts. Our approach is evaluated on the TREC-9 corpus for filtering against the MESH and the UMLS vocabularies.

1 Introduction

Terminology extraction is an essential step in many fields of natural language processing, especially when processing domain-specific corpora. Current algorithms for terminology extraction are most commonly knowledge-driven, using differential analysis and statistical measures for the extraction of domain specific termini. These methods work well, when a large, well-balance reference corpus for the language to process exists. Yet such datasets exist only for a few of the more than 6,000 languages currently in use on the planet. The need is thus for knowledge-free approaches to terminology extraction. In this work, we propose the use of a graph-based clustering algorithm on graphs generated using techniques for the extraction of multi-word units (MWUs). After presenting work related to MWU extraction, we present the metric for MWU extraction used: SRE. This metric is used to generate a directed graph on which SIGNUM is utilized. We present the results achieved using several graph configurations and sizes and show that SIGNUM improves terminology extraction. In order to evaluate our approach, we used the Medical Subject Headings (MESH), with which the TREC-9 collection was tagged, and the Unified Medical Language System (UMLS) vocabularies as gold standards. Last, we discuss some further possible applications of SIGNUM and the results generated using it.

2 Related Work

Depending on the amount of background knowledge needed, two categories of approaches for MWU extraction can be distinguished: knowledge-driven and knowledge-free approaches.

Knowledge-driven approaches fall into two main categories: *syntactic* and *hybrid approaches*. Syntactic approaches use linguistic patterns to extract MWU. LEXTER [2] uses an extensive list of predefined syntactic patterns to segment sentences in their components and identify potential nominal phrases and collocations. Furthermore a list of nouns that use certain prepositions as complements is used to filter the initial segmentation results. By applying a learning approach, LEXTER is then able to improve its results. A similar but semi-automatic strategy is implemented in Termight [4], which uses a combination syntactic patterns and frequency analysis for MWU extraction, the most frequent syntactic patterns being seen as more relevant. Purely syntactic approaches are always language-specific due to the patterns they necessitate. In order to improve their flexibility *hybrid approaches* were introduced. They combine syntax and statistics for MWU extraction either by first applying a numerical preprocessing to detect potential candidates for MWU and pruning the resulting list using linguistic patterns (see e.g., XTRACT [19]) or by processing the input in the reserve order, first using syntactic patterns such as NOUN NOUN and ADJ NOUN and subsequently filtering the results using numerical models (see e.g., [11]). Still they have the restrictions of syntactic approaches as they are language-specific as well.

Most *knowledge-free approaches* use probabilistic metrics (e.g., the pure occurrence frequency [9], the Dice formula [6], Pointwise Mutual Information [3], the Symmetric Conditional Probability [8]) to compute the significance of collocations. Schone [17] proposed an approach based on Latent Semantic Analysis to compute the semantic similarity of terms. The extracted similarity values are used to improve the score function during the MWU extraction. This technique shows some improvement, yet is computationally very expensive. Another approach proposed later by Dias [5] yields comparable improvement and is computationally cheaper. Dias uses pattern distributions over positional word n-grams to detect MWUs. He defines a new metric called Mutual Expectation (ME). ME models the non-substitutability and non-modifiability of domain-specific MWU. SRE is similar to ME, yet yields a further component, which takes the distribution of MWU over documents into consideration, modeling their specificity.

3 Smoothed Relative Expectation

To compute n-gram scores, we used the Smoothed Relative Expectation (SRE) [15] given by

$$SRE(w) = p(w)\frac{e^{-\frac{(d(w)-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}}\frac{nf(w)}{\sum_{i=1}^{n} f(c_1...c_i * c_{i+2}...c_n)}, \tag{1}$$

where

- $w = c_1...c_n$,
- $*$ is the wildcard symbol,
- $d(w)$ returns the number of documents in which w occurs,
- μ and σ^2 are the mean and the variance of the occurrence of an n-gram in a document respectively,
- $p(w)$ is the probability of occurrence of w in the whole corpus,
- $f(w)$ is the frequency of occurrence of w in the whole corpus and
- $c_1...c_i * c_{i+2}...c_n$ are all patterns such that $ham(w, c_1...c_i * c_{i+2}...c_n) = 1$.

SRE computes the expectation of a given word combination relatively to other word combinations at a Hamming distance [10] of 1 and combines it with their distribution over the documents in the corpus. It can be used to compute n-grams of all lengths. The SRE metric was compared to five other state-of-the-art metrics (DICE = dice coefficient, FR = frequency, ME = Mutual Expectation, PMI = Pointwise Mutual Information, SCP = Symmetric Conditional Probability) on the extraction of bi-grams out of the TREC-9 corpus for filtering, which consists of abstracts of publications from the medical domain. The gold standard was the MESH vocabulary. Table 1 and Figure 1 give the precision achieved when ordering bi-grams according to their score and considering the best scoring bi-grams. SRE clearly outperforms all other metrics. A t-test with a confidence level of 99% reveals that the precision achieved by SRE is significantly better than that of all other metrics.

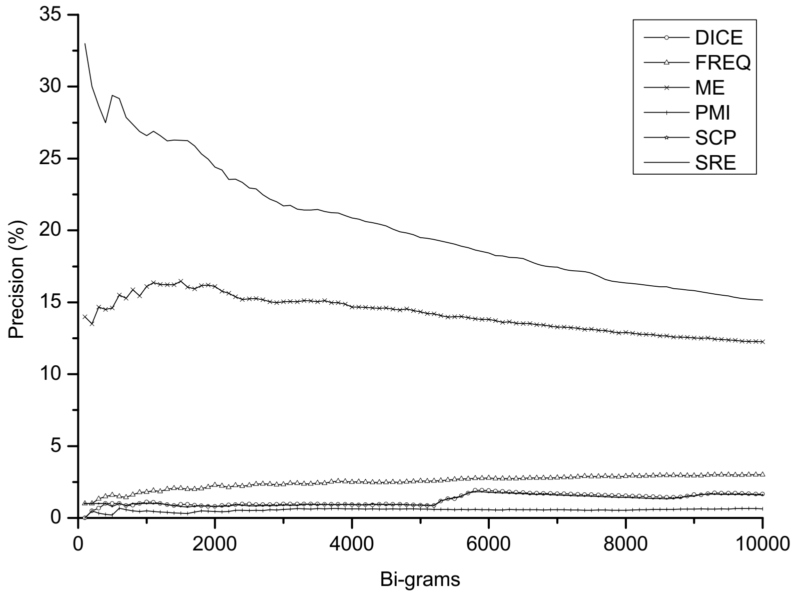


Fig. 1. Precision of six metrics

Table 1. Precision of six metrics for bi-gram extraction

Bi-grams	FR	DICE	SCP	PMI	ME	SRE
500	1.60	1.00	0.80	0.20	14.60	**29.40**
1000	1.80	1.10	1.00	0.50	16.10	**26.60**
1500	2.07	0.93	0.80	0.33	16.40	**26.26**
2000	2.30	0.80	0.80	0.45	16.10	**24.40**
2500	2.28	0.96	0.84	0.52	15.24	**22.96**
3000	2.33	0.97	0.91	0.65	15.03	**21.70**
3500	2.42	0.97	0.91	0.65	15.02	**21.45**
4000	2.50	0.93	0.95	0.63	14.68	**20.88**
4500	2.46	0.96	0.91	0.62	14.60	**20.31**
5000	2.56	0.88	0.90	0.62	14.34	**19.50**

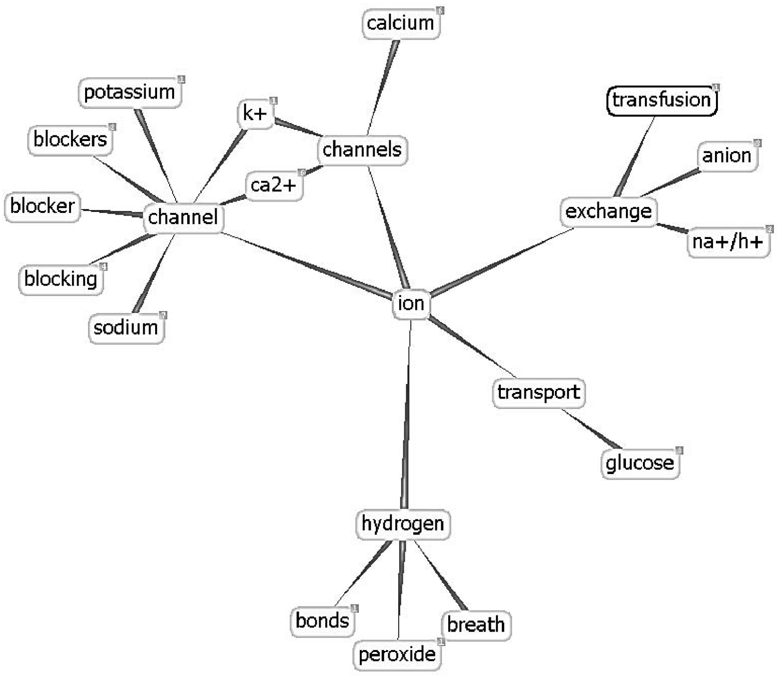


Fig. 2. Bi-gram graph for "ion"

Given any score function *score*() for word sequences, the results achieved by MWU extraction technique can be represented as weighted graphs $G = (V, E, \omega)$, with

– V being the vocabulary of the language,
– $E = \{(u, v) : score(uv) > 0\}$ and
– $\omega(uv) = \varphi(score(uv))$.

Figure 2 displays an example of such a graph. The score function was SRE, $\varphi = -1/log_{10}$.

4 SIGNUM

Collocation graphs display small-world characteristics [20], thus they have a high clustering coefficient. This property of collocation graphs makes them particularly suitable for graph clustering algorithms. Especially, the small mean path length between nodes allows the use of algorithms using exclusively local information for clustering, since the transfer of local information to all other nodes of the graph occurs considerably faster than in purely random graphs [13]. The main advantage of clustering approaches, which use local information lies at hand: they are computationally cheap and can thus deal with very large graphs, such as those usually generated during NLP. Although graphs extracted out of bi-grams (see Figure 2) are directed and thus not collocation graphs as such, they display similar topological characteristics (clustering coefficient, edge degree, etc.) and can thus be clustered using local information as well.

4.1 Basic Idea

SIGNUM was designed to achieve a binary clustering on weighted directed graphs. The basic idea behind SIGNUM originates from the spreading activation principle, which has been used in several areas such as neural networks and information retrieval [1]: the simultaneous propagation of information across edges. In the case of the basic version of SIGNUM, this information consists of the classification of the predecessors of each node in one of the two classes dubbed + and −. Each propagation step consists of simultaneously assigning the predominant class of its predecessors to each node. The processing of a graph using SIGNUM thus consists of three phases: the *initialization phase*, during which each node is assigned an initial class; the *propagation phase*, during which the classes are propagated along the edges until a termination condition is satisfied, leading to the *termination phase*. The resulting categorization is then given out.

4.2 Formal Specification

Phase I: Initialization. Directed weighted graph are triplets G = (V, E, ω) with $E \subseteq V \times V$ and $\omega : E \rightarrow \mathbb{R}$. Let

$$\sigma : V \rightarrow \{+, -\} \qquad (2)$$

be a function, which assign vertices a positive or negative signum. The goal of the initialization phase is the definition of the initial values of this function (i.e., the definition of the value it initially returns for each and every node in V). Depending on the field in which SIGNUM is used, this definition might differ. In the special case of terminology extraction, the information available about the edges is more suitable to determine the initial values of σ. Thus, let

$$\sigma_e : E \rightarrow \{+, -\} \tag{3}$$

be a function, which assigns a positive or negative signum to edges. The weight of the edge between two terms allows assumptions concerning the domain-specificity of the terms it connects. Let σ_e be fully known. Furthermore, let

$$\Sigma^+(v) = \{u : uv \in E \wedge \sigma_e(uv) = +\} \tag{4}$$

and

$$\Sigma^-(v) = \{u : uv \in E \wedge \sigma_e(uv) = -\}. \tag{5}$$

The initial values of σ are then be given by:

$$\sigma(v) = \begin{cases} + \text{ if } \sum_{u \in \Sigma^+(v)} \omega(uv) \ > \sum_{v \in \Sigma^+(v)} \omega(uv); \\ - \text{ else.} \end{cases} \tag{6}$$

This initialization prioritizes one class (in this case the $-$ class). In the case of lexicon extraction, this implies that a word is considered as initially not belonging to the lexicon when the evidence for its belonging equals the evidence for the opposite.

Phase II: Propagation. Each node is assigned the class of the majority of its predecessors. The class $-$ is assigned in case of a tie. Formally,

$$\sigma(v) = \begin{cases} + \text{ if } \sum_{\sigma(u)=+} \omega(uv) > \sum_{\sigma(u)=-} \omega(uv) \\ - \text{ else.} \end{cases} \tag{7}$$

Obviously, each edge is used exactly once during a propagation phase, making each step of SIGNUM linear in the number of edges. Furthermore, the reassignment of the classes to the node occurs simultaneously, making SIGNUM easy to implement in a parallel architecture.

Phase III: Termination. The algorithm terminates when the function σ remains constant. Obviously, several graph configurations exist, in which this propagation approach does not terminate. Fig. 3 displays an example of such a configuration. Every edge has a weight of 1. The nodes without relief are assigned to $+$, else to $-$. Yet such examples appear rarely in real life data, due to the fact that collocation graphs extracted from real world data are usually large and scale-free. For other categories of graphs, the simplest ways to ensure that the algorithm terminates is to set a threshold either for the number of iteration or for the number of changes.

4.3 Extensions

The SIGNUM concept can be extended is several ways, of which two are of particular interest for NLP. First, SIGNUM can be extended to be used on all other graph topologies (i.e., undirected and unweighted graphs): Undirected graphs can

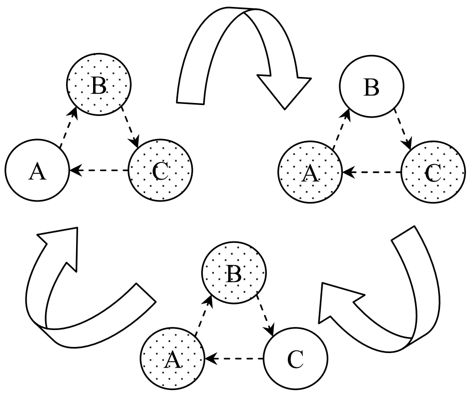


Fig. 3. Example of non termination of SIGNUM

be considered as directed graphs, with the particularity that each edge (u, v) has an equivalent edge (v, u) yielding the same weight. Furthermore unweighted graphs can be modeled as graphs with a constant edge weight function 1.

SIGNUM can also be used to cluster graphs with an unknown number of classes, for example to detect semantic classes. However, the initialization needs to be slightly modified, by assigning the same unique class label to each clique or almost-clique of the graph . An algorithm implementing such a clustering was presented in [14].

5 Using SIGNUM for Lexicon Extraction

For the practical application of lexicon extraction, we use the fact that termini from the same domain tend to appear in the same paradigmatic context, i.e., to collocate [12]. Thus, the predecessors and successors of domain-specific words can be seen as potentially belonging to the same lexicon. The initialization of the graph can thus be based on the information at hand, i.e., the degree to which words collocate. Assuming that we have an ordered list of ordered word pairs extracted from a domain specific corpus, a natural initialization of the graph would consist of selecting the upper half of the list as initially belonging to the same class (i.e., +) and the rest as belonging to the other one (i.e., −). The subsequent use of SIGNUM over the resulting graph to have the predecessors of the node would then confirm the hypothetic classes by their own classification.

5.1 Data Set

The underlying data set for the results presented below was extracted from the TREC-9 corpus [16]. This corpus is a test collection composed of abstracts of publications from the medical domain. The entries in the available test corpus included the abstract text of medical publications (marked in each entry with

Table 2. Topology of n-gram graphs

N-grams	Nodes	Edges	Components	Avg. N/C	Avg. E/C	Max N/C	Max E/C
10,000	11,106	9,969	2,606	4.26	3.83	4,854	6,282
20,000	23,733	19,939	7,415	3.20	2.69	7,136	10,688
50,000	47,905	49,685	14,579	3.29	3.41	13,895	30,204
100,000	79,658	98,893	21,454	3.71	4.61	25,315	65,811

a "`.W`") and further metadata such as the subject, type of publication, etc. The data extraction process consisted exclusively of the retrieval of all the text entries (i.e. those marked with "`.W`" in the TREC-9 corpus) and the deletion of punctuation. 233,445 abstracts (244 MB) were retrieved and utilized for the evaluation presented in this section. 355,616 word forms were extracted from the corpus with a mean frequency of 109.08. 6,096,183 different bi-grams were found, their mean frequency being 6.36. The mean occurrence of bi-grams in documents was 5.67 with a standard deviation of 137.27. Figure 2 displays an excerpt of the graph extracted from the data set. The length of the edges is inversely proportional to their weight.

5.2 Initialization

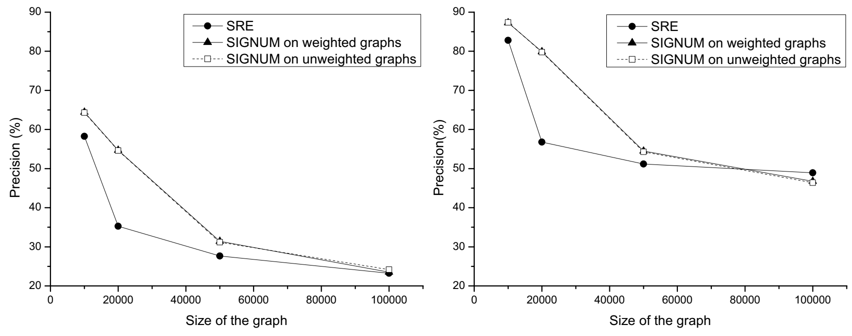
The scores computed using SRE bear small values for large corpora, ranging between 0 and 10^{-5} in the special case at hand. The weight $\omega(w_1 w_2)$ of the edge between two words w_1 and w_2 was thus set to

$$\omega(w_1 w_2) = \frac{-1}{log_{10}(SRE(w_1 w_2))}. \tag{8}$$

In order to compute σ_e, the sum of all scores was computed and halved. Subsequently, the bi-gram list was processed sequentially. Bi-gram were assigned positive signum values until the sum of their scores reached half of the total sum of scores. The residual edges were assigned negative signum values.

5.3 Results

SIGNUM was tested using the n best scoring bi-grams, with n taking values between 10,000 and 100,000 (see Table 2; N/C = nodes/component, E/C = edges/component). The resulting graphs presented a similar topology: they consisted of a large main component and a large number of small components. This topology is similar to that reported by other groups (see e.g., [7]). SIGNUM was tested on two graph configurations against the results of SRE: in the first configuration, the weights were not considered during the propagation phase (i.e., they were all set to 1). In the second configuration, the weights were considered. Two golden standards were used to measure the precision of the results achieved. The MESH vocabulary was selected because it was used to tag the TREC-9 data set. Due to the restrictiveness of the MESH vocabulary, the more complete UMLS



(a) Precision using MESH (b) Precision using UMLS

Fig. 4. Precision measured using the MESH and UMLS vocabularies

Table 3. Comparison of the precision of SRE and SIGNUM. The left column of each block displays the results of the precision using MESH, while the right one displays the same metric when using UMLS.

N-grams	SRE		Unweighted		Weighted	
10,000	58.26	82.81	64.29	**87.37**	**64.44**	87.31
20,000	35.27	56.79	54.63	79.77	**54.71**	**79.89**
50,000	27.67	51.18	31.18	54.29	**31.34**	**54.50**
100,000	23.23	**48,91**	**24.17**	46.40	23.52	48.91

vocabulary was also utilized for measuring the precision achieved by SRE and SIGNUM. The precisions achieved is displayed in Table 3. The left sub-column of each of the method columns displays the precision achieved using MESH as reference vocabulary. The left one display the same metric on the UMLS vocabulary. Both results are displayed graphically in figure4.

As shown by figure4(a) and 4(b), the weighting of the graphs does not significantly alter the performance of SIGNUM on the graph at hand. This hints toward the fact that the topology of the graph is the key influence for the performance of SIGNUM and not the weight distribution over the graph. A significantly high difference between the results of SRE and SIGNUM is observed when the graph is generated out of 20,000 bi-grams. However, the gain in precision then decreases with the size of the graph. This can be explained by the fact that larger graph include more functions words, which tend to collocate with terms from both classes and thus augment the total weight of the intra-cluster edges, leading to more errors as the class labels are transferred over the edges. This is especially clear, when the results achieved on the 100,000 bi-gram graphs are considered.

6 Conclusion and Outlook

We presented SIGNUM, a novel graph-based approach for the extraction of domain-specific terminology and showed that it improves the results achieved using state-of-the-art techniques in the task of extracting one-word dictionaries from word collocation graphs. As the technique for MWU extraction and SIGNUM are independent, our approach can be used for the improvement of any of the metrics for MWU extraction presented above. The results achieved using SIGNUM can be used to filter MWU results and improve the quality of automatically generated multi-word dictionaries. SIGNUM furthermore bears the advantage of using solely local information available to each node, making it computationally cheap. Therefore, it is able to handle very large graphs. Due to the simultaneous reclassification of nodes, SIGNUM can be easily implemented in a parallel architecture.

A category of graphs which was not considered in this work presented are link graphs, which can be used for disambiguation (see e.g., [18,7]) and thus for further improvement of the MWU extraction. This work is currently being undertaken.

References

1. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press, Addison-Wesley (1999)
2. Bourigault, D.: Lexter: A terminology extraction software for knowledge acquisition from texts. In: 9th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada (1995)
3. Church, K.W., Hanks, P.: Word association norms, mutual information, and lexicography. In: Proceedings of the 27th. Annual Meeting of the Association for Computational Linguistics, Vancouver, B.C, pp. 76–83. Association for Computational Linguistics (1989)
4. Dagan, I., Church, K.: Termight: identifying and translating technical terminology. In: Proceedings of the fourth conference on Applied natural language processing, pp. 34–40. Morgan Kaufmann, San Francisco (1994)
5. Dias, G.: Extraction Automatique dAssociations Lexicales partir de Corpora. PhD thesis, New University of Lisbon (Portugal) and LIFO University of Orléans (France), Lisbon, Portugal (2002)
6. Dice, L.R.: Measures of the amount of ecological association between species. Ecology 26, 297–302 (1945)
7. Dorow, B.: A Graph Model for Words and their Meanings. PhD thesis, University of Stuttgart, Stuttgart, Germany (2006)
8. da Silva, J.F., Lopes, G.P.: A local maxima method and a fair dispersion normalization for extracting multi-words units from corpora. In: Sixth Meeting on Mathematics of Language, Orlando, USA, pp. 369–381 (1999)
9. Giuliano, V.E.: The interpretation of word associations. In: Stevens, M.E., et al. (eds.) Proceedings of the Symposiums on Statistical Association Methods for Mechanical Documentation, Washington D.C., number 269, NBS (1964)
10. Hamming, R.: Error-detecting and error-correcting codes. Bell System Technical Journal 29(2), 147–160 (1950)
11. Justeson, J., Katz, S.: Co-occurrences of antonymous adjectives and their contexts. Computational Linguistics 17(1), 1–20 (1991)
12. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing, 1st edn. MIT Press, Cambridge (1999)

13. Milgram, S.: The small-world problem. Psychology Today 2, 60–67 (1967)
14. Ngonga Ngomo, A.-C.: CLIque-based clustering. In: Proceedings of Knowledge Sharing and Collaborative Engineering Conference, St. Thomas, VI, USA (November 2006)
15. Ngonga Ngomo, A.-C.: Knowledge-free discovery of domain-specific multi-word units. In: Proceedings of the 2008 ACM symposium on Applied computing, ACM, New York (to appear, 2008)
16. Robertson, S.E., Hull, D.: The TREC 2001 filtering track report. In: Text REtrieval Conference (2001)
17. Schone, P.: Toward Knowledge-Free Induction of Machine-Readable Dictionaries. PhD thesis, University of Colorado at Boulder, Boulder, USA (2001)
18. Schütze, H.: Automatic word sense discrimination. Computational Linguistics 24(1), 97–123 (1998)
19. Smadja, F.A.: Retrieving collocations from text: Xtract. Computational Linguistics 19(1), 143–177 (1993)
20. Steyvers, M., Tenenbaum, J.: The large-scale structure of semantic networks: Statistical analyses and a model of semantic growth. Cognitive Science: A Multidisciplinary Journal 29(1), 41–78 (2005)

Arabic Morphology Parsing Revisited

Suhel Jaber and Rodolfo Delmonte

University Ca' Foscari, Dept. Language Sciences, Laboratory Computational Linguistics, Ca' Bembo, Dorsoduro 1705, 30123 Venezia, Italy
`{jaber,delmont}@unive.it`

Abstract. In this paper we propose a new approach to the description of Arabic morphology using 2-tape finite state transducers, based on a particular and systematic use of the operation of composition in a way that allows for incremental substitutions of concatenated lexical morpheme specifications with their surface realization for non-concatenative processes (the case of Arabic templatic interdigitation and non-templatic circumfixation).

Keywords: Arabic, morphology, non-concatenative, finite state, composition.

1 Introduction

In this paper we propose a new approach to the description of Arabic morphology using 2-tape finite state transducers, based on a particular and systematic use of the operation of composition in a way that allows for incremental substitutions of concatenated lexical morpheme specifications with their surface realization for non-concatenative processes (the case of Arabic templatic interdigitation and non-templatic circumfixation). Then we compare it with what in our opinion represents the state-of-the-art among the 2-tape finite-state implementations, that of Xerox [1], which is mainly based on the operation of intersection. We intentionally limit ourselves to the evaluation of 2-tape strictly finite-state implementations for this paper, leaving out n-tape implementations such as [2] and [3], and those based on extended finite-state automata, such as [4]. In any case we believe that our approach could be trivially adapted to n-tape implementations as well.

In this paper we argue that:

1. the use of composition allows to overcome certain technical problems inherent to the use of intersection;
2. the method of incremental substitutions through compositions allows for an elegant description of all main morphological processes present in natural languages including non-concatenative ones in strict finite-state terms, without the need to resort to extensions of any sort;
3. our approach allows for the most logical encoding of every kind of dependency, including traditional long-distance ones (mutual exclusiveness), circumfixations and idiosyncratic root and pattern combinations;
4. a smart usage of composition such as ours allows for the creation of a same system that can be easily accomodated to fulfil the duties of both a stemmer (or lexicon development tool) and a full-fledged lexical transducer.

Here below is a short review of the Xerox implementation that we hold as our 'gold standard'.

2 Review of the 'Gold Standard'

The Xerox implementation follows from a late eighties commercial project at ALPNET which resulted in an Arabic morphological analyzer based on an enhanced Two-Level approach [5].

$$a{:}b\ . \tag{1}$$

$$a{:}\varepsilon\ a{:}b \cap a{:}\varepsilon\ a{:}b\ . \tag{2}$$

$$a{:}\varepsilon\ a{:}b\ . \tag{3}$$

$$a{:}\varepsilon\ a{:}b \cap a{:}b\ a{:}\varepsilon\ . \tag{4}$$

Two-Level morphology [6] differs from pure finite-state morphology [7] mainly for the way symbol pairs **(1)** are treated: as simple atomic symbols in Two-Level calculus and as relation between languages in the case of pure finite-state calculus. This fundamental distinction determines also difference in behaviour, such as the conception and closure of operations like intersection. Whereas in finite-state calculus intersection of regular language relations is not closed under the set of regular language relations itself, and therefore will not usually be computable, in Two-Level calculus this problem does not occur since intersection is intended not as the intersection of regular language relations but as the intersection of regular languages whose symbols just happen to be pairs. By means of example, **(2)** equals **(3)** in Two-Level calculus but is not computable in finite-state calculus, and **(4)** is still computable in Two-Level calculus but yields the empty set whereas its operands taken alone would encode exactly the same relation inside the framework of finite-state calculus.

The main insight leading the whole Xerox Arabic language analyzer development is that root and pattern (and even vocalism) morphemes could be intersected (more or less problematically) to form a proper Arabic word, an insight which dates back to the Two-Level implementation of [8]. With the licensing of the previous project material from ALPNET to Xerox, consequent moving from Two-Level calculus to finite-state calculus has meant leaving behind any hope of being able to intersect relations, having instead to intersect regular expressions which would be in turn mapped in a totally arbitrary way to any other regular expression by means of a crossproduct operator. There is nothing directly leading from the intersected representation to its 'decomposition' in morphemes.

Our implementation expressly tries to avoid this problem. Note that we do that by using 2 tapes only. We will cover these algorithms more in depth later, but for now let's get back to the kind of problems intersection generates, among which a special place is reserved to the fact that the operator of 'general intersection', meaning the operator having the usual set-theoretical semantics and implemented with the general case algorithms that we would always think of when talking of 'intersection', applied to certain morphemic systems such as that stemming from the linguistic analysis of

[9] (the one returned by the Xerox Arabic Morphological Analyzer demo at http://www.xrce.xerox.com/competencies/content-analysis/arabic/), yields very awkward results.

Here's an example: let | represent the union or disjunction operator, double quotes the escape character and `[C]` the language consisting of the union of every Arabic letter that might constitute a root morpheme, i.e. every consonant (the square brackets are just for grouping purposes).

Using the transliteration system of [10] (of which we present a small fragment in the appendix to this paper), in Xerox Finite State Tool (*xfst*) syntax we would notate that as:

```
define C [' | b | t | v | j | H | x | d | "*" | r | z |
s | "$" | S | D | T | Z | E | g | f | q | k | l | m | n
| h | w | y];
```

According to the analysis outlined in [9] (upon which we agree for purposes of fair evaluation in this review), the Arabic verb اِجْتَمَعَ is composed of a morphemic pattern اِـْتَـَـَ (where the *tatweel* symbol stands for any root consonant), a root morpheme ج م ع and a suffix َ .

Now let `&` be the intersection operator, `*` the 0 or more times iteration operator (commonly known as "Kleene star") and `?` a symbol representing any symbol (so-called wildcards).

In this case then, the following two regular expressions are equal:

```
read regex [[A i C o t a C a C & ?* j ?* m ?* E ?*] a];
read regex [A i j o t a m a E a];
```

Unfortunately though, in the case of the verb اِقْتَتَلَ, which is analyzed as composed by the same morphemic pattern as the previous verb plus the root morpheme ق ت ل, an analogous expression would not work. Indeed, the following two regular expressions are equal:

```
read regex [[A i C o t a C a C & ?* q ?* t ?* l ?*] a];
read regex [A i q o t a C a l a | A i q o t a l a C a];
```

This happens because the `[t]` in the root morpheme `[?* q ?* t ?* l ?*]` matches the `[t]` inside the pattern morpheme `[A i C o t a C a C]` leaving either the second or the third `[C]` in there unmatched. An initial workaround to this problem in [11] has been the choice to denote root consonants and template consonants differently, that's to say as in the following root consonant regular expression definition, where curly brackets are normal string symbols used to provide the wanted distinction from template consonants:

```
define C "{" [' | b | t | v | j | H | x | d | "*" | r |
z | s | "$" | S | D | T | Z | E | g | f | q | k | l | m
| n | h | w | y] "}";
```

Beware though, compiling the following consequent new relation results heavier on any machine:

```
read regex [[A i C o t a C a C & ?* "{" q "}" ?* "{" t
"}" ?* "{" l "}" ?*] a];
```

The solution to this commercially critical problem that has been thought of at Xerox is that of a new algorithm, called *merge* [12], whose operator `.m>.` takes as input strings of the kind of `[q t l]` and `{AiCotaCaC}`, as in the following example:

```
undefine C

list C ' b t v j H x d "*" r z s "$" S D T Z E g f q k
l m n h w y;

read regex [q t l] .m>. {AiCotaCaC};
```

In this case, the algorithm's behaviour can be correctly resumed as that of an operation which instantiates the actual value of 'class symbols' `C` in the template network from the value of the symbols in the string described by the filler network, in their correct order; both the template network's language class symbols and the filler network's language symbols must be in the same quantity for the operation to be successful. Note the lack of Kleene stars in the filler network. The same result, with a pure finite-state intersection operation would have been obtained only by compiling the following computationally more expensive expression, where `.o.` is the composition operator, `0` represents an ε-transition and `\"{"` is equivalent to `[? - "{"]` in Xerox syntax:

```
unlist C

define C "{" [' | b | t | v | j | H | x | d | "*" | r |
z | s | "$" | S | D | T | Z | E | g | f | q | k | l | m
| n | h | w | y] "}";

read regex [\"{" | 0:"{" ? 0:"}"]* .o. [[A i C o t a C
a C & ?* "{" q "}" ?* "{" t "}" ?* "{" l "}" ?*] a] .o.
[\"{" | "{":0 ? "}":0]*;
```

To complete the picture concerning the problems generated by the usage of an intersection approach, let's explain the solution adopted at Xerox to resolve the issue we have hinted at some lines ago, that's to say the one related to the impossibility to intersect relations in a finite-state framework. Well, as we have already said, the only possible way to tackle this problem without renouncing to an intersection-based approach is to intersect mere expressions and then turn them into relations by means of a crossproduct operation that maps strings together. [3] though made notice of the fact that in this way the intersection operator becomes a 'destroying' operator, meaning that after one has intersected all the morphemes there is no way to map each one of them to its correct lexical counterpart but instead all one can do is match full superficial (or intermediate, for all that counts) strings like "Aiqotatal" (representing what we may call a ‘stem’, i.e. a pattern still lacking the suffix but correctly fulfilled by the root) to another one (that we can hardly call ‘lexical’ from a linguistic point of view) such as “^[q t l .m>. {AiCotaCaC}^]”.

One might say that this sort of mapping kind of defeats the purpose of building a machine to run the analysis in the first place, if one has to produce the stem analysis by hand. This is not exactly the case though when using the Xerox tools, thank their compile-replace algorithm in [12], which lets the user specify a regular expression whose denoted string gets mapped to a string equal to the regular expression itself. This might sound complicated, but just think of a relation of strings with regular expression syntax, the surface of which will eventually get compiled and represents in fact the string the expression denotes.

Therefore, specifying a relation such as:

```
"^[" "q t l .m>. {AiCotaCaC}" "^]" .x. "^[" "q t l .m>.
{AiCotaCaC}" "^]"
```

(or even shorter, building an expression `"^[" "q t l .m>. {AiCotaCaC}" "^]"` that will be interpreted in proper context as the same identity relation) means in fact to account for the following relation:

```
"^[" "q t l .m>. {AiCotaCaC}" "^]" .x. [A i q o t a t a
l]
```

because one side of the relation gets doubly compiled at will. It becomes a matter of what we may call ‘meta-strings’ at the end (or ‘meta-expressions’, i.e. expressions *about* expressions). Note though that there are some usages of compile-replace different than this that allow for the resolution of problems which would normally exceed finite-state power, such as palindrome extraction, as explicitly stated in [12] itself.

Apart from all these solutions that might not sound very orthodox to formal language theorists and linguists alike, the last big problem with the Xerox implementation is that the lexical analysis in some cases appears to be wrong. We will not try to guess why this might have happened, it could have been the difficulty of encoding proper phonological rules or mere theoretical dissent or anything again, but as a matter of fact every Arabist would tell you that weak verbs of the kind of “qAla” have a lexical origin “qawala” and not “qawula”, even if we all wished it was “qawula” because mapping the superficial string “qulotu” to a lexical counterpart “qawulotu” instead of “qawalotu” would mean easier mapping rules and so on. Unfortunately, that’s not the case [13]. How can we be so sure of this? Well, that’s because of two reasons [14]:

1. verbs with pattern CaCuCa are always intransitive;
2. names deriving from verbs with pattern CaCuCa usually show the pattern CaCiyC (such is the case for instance of “Zariyf” from “Zarufa” and “$ariyf” from “$arufa”) whereas names deriving from verbs like “qAla” show the pattern CACiC (“qa}il”, with *hamza* instead of *waaw* because of other phonological rules)[1].

[1] In Arabic script ظَريـف from ظَرُفَ, شَريـف from شَرُفَ and قـا ئِل from قـا لَ.

Now we show how our implementation avoids many of the aforementioned problems and reaches descriptive elegance without resorting to extensions of any sort.

3 The "Incremental Substitutions" Compositional Approach

Let's have a look at a simplified version of a typical relation encoding in our implementation, using xfst syntax for purposes of consistency throughout our paper.

```
define C [' | b | t | v | j | H | x | d | "*" | r | z |
s | "$" | S | D | T | Z | E | g | f | q | k | l | m | n
| h | w | y];

read regex [[q t l| k t b| T r q] " Form_I_Impf_Act_u"]
.o. [C 0:o C 0:u C " Form_I_Impf_Act_u":0];
```

From an 'analytical' (as opposed to 'generative') point of view we can interpret this last regular relation as a two-phase mapping:

1. in `[C 0:o C 0:u C " Form_I_Impf_Act_u":0]` the vowels in the Verb Form I Imperfect Active pattern ـــْـــُـــ get 'filtered' in the passage from surface to lexical representation, 'erased' and 'substituted' by the agreeing tag which is in fact concatenated to the end of the remaining lexical material made up of those `[C]` roots which were allowed to 'pass through';
2. the resulting lexical string is 'passed' as an argument to a second regular expression `[[q t l | k t b | T r q] " Form_I_Impf_Act_u"]` by means of composition, which will operate on the remaining material if and only if the tags (in this case only 1) concatenated at the end of the regular expression correspond to those generated in or passed through the previous phase of analysis; in this case all it would do on the remaining material would be constraining its quality to that of the actual root morphemes which are allowed to combine with the pattern represented by the concatenated tag.

Notice that with this approach we don't need to previously define the `[C]` language, even if we did it in the previous example. Indeed the following regular expression denotes exactly the same relation as the previous one.

```
read regex [[q t l| k t b| T r q] " Form_I_Impf_Act_u"]
.o. [? 0:o ? 0:u ? " Form_I_Impf_Act_u":0];
```

The employment of ε-transitions is crucial here, because it's the erasure of already analyzed sectors of a string that allows for the smart resolution of the problem of the root and template consonants ambiguity without the need for any additional mechanisms such as those featured in the Xerox implementation, as one can easily ascertain by looking at the following regular expression construct:

```
read regex [[q t l | k t b] " Form_VIII_Impf_Act"] .o.
[? 0:o 0:t 0:a ? 0:i ? " Form_VIII_Impf_Act":0];
```

As stated at the beginning of this paper, we managed to encode a certain kind of dependency (idiosyncratic root and pattern combinations) without the need for unification-based grammars or formalisms and enhancements of any sort. Of course we could not enlist every root and pattern combination in this paper, but by the following expression we show how our implementation organizes more idiosyncratic combinations together in one compact structure:

```
read regex [
[[k t b | q t l] " Form_I_Perf_Act_a"] |
[[D r b | H s b] " Form_I_Perf_Act_i"] |
[["$" r f | H s n] " Form_I_Perf_Act_u"]
] .o. [
[? 0:a ? 0:a ? " Form_I_Perf_Act_a":0] |
[? 0:a ? 0:i ? " Form_I_Perf_Act_i":0] |
[? 0:a ? 0:u ? " Form_I_Perf_Act_u":0]
];
```

Let's now have a look at how circumfixation is handled in our implementation (again, for matters of simplicity and space not all the circumfixes will be enlisted, but of course in the actual implementation items are expanded to fully cover the language):

```
read regex
[[q t l] " Form_I_Impf_Act_u"
[" 1_Pers_Sing_Ind_a" | " 1_Pers_Plur_Ind_a"]] .o.
[? 0:o ? 0:u ? " Form_I_Impf_Act_u":0
[" 1_Pers_Sing_Ind_a" | " 1_Pers_Plur_Ind_a"]] .o.
[0:' 0:a ?* 0:u " 1_Pers_Sing_Ind_a":0 |
0:n 0:a ?* 0:u " 1_Pers_Plur_Ind_a":0];
```

Note that other implementations including the Xerox one in [15] usually deal with certain long-distance dependencies through the use of composition, but in a very different way:

1. all the prefixes, stems and suffixes are concatenated together to form every potential combination (even prohibited ones), and prefixes and suffixes are assigned each a distinctive tag;
2. through the use of composition, patterns featuring mutually exclusive tags are explicitly removed from the network.

Our method, on the other hand, just assigns one tag to each circumfix (for other purposes, moreover) and anyway the correct circumfixation is created in one single process instead of total prefixation plus total suffixation and subsequent pruning.

We're now ready to give an interpretation to our approach from a 'generative' point of view as that of an n-phase mapping:

1. in the first regular expression we enlist in a concatenative way all the morphemes (or rather, their lexical representations) which make up a word, in the order in which we should process their 'merging' with the string we obtain at each phase;

2. in the subsequent regular expressions we process their ‘merging’ with any intermediate string previously obtained, according to the order of the remaining tags at each point, ‘erasing’ one tag at a time after its surface counterpart has been created and merged to the rest.

In this way we were able to give a linear rendering of what globally assumes the entity of a hierarchical representation (cfn. ‘morphosyntax’) or incremental creation of bigger building blocks from already elaborated ones, i.e.:

ق ت ل + ـــْــُــ = قْـتُـل (5)

قْـتُـل + يَـــــُـ = يَـقْـتُـلُ (6)

To conclude this section, let us show how our initial commitment of implementation flexibility is honoured in the following regex:

```
read regex [
[? ? ? " Form_I_Perf_Act_a"] |
[? ? ? " Form_I_Perf_Act_i"] |
[? ? ? " Form_I_Perf_Act_u"]
] .o. [
[? 0:a ? 0:a ? " Form_I_Perf_Act_a":0] |
[? 0:a ? 0:i ? " Form_I_Perf_Act_i":0] |
[? 0:a ? 0:u ? " Form_I_Perf_Act_u":0]
];
```

This is one of the previously shown regexes, only without constraints on the allowed root morphemes for each pattern. By running this kind of machine on an Arabic text input we get an output of all the encountered root bundles classified by the patterns they were found in. This has helped us build the actual lexicon out of different sources.

4 Implementation Evaluation

For purposes of evaluation we have written a script composing more than 4700 root morphemes with the verbal patterns they can actually combine with extracted from several databases, including those in [10]. This grammar compiled in real time on an Intel Pentium M 730 1.60 GHz based Microsoft Windows XP system using the Xerox Finite-State Tool version 2.6.2.

5 Conclusions

In this paper we have explored the potential in a particular encoding of lexical transducers that we have labelled as the “incremental substitutions” compositional model as applied to the Arabic language.

We’ve shown how this new approach solves some technical problems that rise with the intersectional approach used by another 2-tape implementation, the ‘gold standard' one of [1]. We've also given hands-on details on our implementation, exemplifying how most morphological processes and descriptions are actually dealt with by going through some simplified snippets of code.

Moreover, we have designed more than one way our model could be put to practical usage (stemming, field research and lexicon developing, morphological analysis and generation).

Ultimately, we have shown that our model allows for a fair description of Arabic morphology in a strictly finite-state framework without the need to resort to enhancements or extensions of any sort.

References

1. Beesley, K.R.: Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001. In: Proceedings of the Workshop on Arabic Language Processing: Status and Prospects. 39th Annual Meeting of the Association for Computational Linguistics, pp. 1–8. Association for Computational Linguistics, Morristown, NJ, USA (2001)
2. Kay, M.: Nonconcatenative Finite-State Morphology. In: Proceedings of the Third Conference of the European Chapter of the Association for Computational Linguistics, pp. 2–10. Association for Computational Linguistics, Morristown, NJ, USA (1987)
3. Kiraz, G.A.: Multitiered Nonlinear Morphology Using Multitape Finite Automata: A Case Study on Syriac and Arabic. Computational Linguistics 26(1), 77–105 (2000)
4. Cohen-Sygal, Y., Wintner, S.: Finite-State Registered Automata for Non-Concatenative Morphology. Computational Linguistics 32(1), 49–82 (2006)
5. Beesley, K.R.: Computer Analysis of Arabic Morphology: A Two-Level Approach with Detours. In: Comrie, B., Eid, M. (eds.) Perspectives on Arabic Linguistics, III: Papers from the Third Annual Symposium on Arabic Linguistics, Benjamins, Amsterdam, pp. 155–172 (1991)
6. Koskenniemi, K.: Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production. Publication 11. University of Helsinki, Department of General Linguistics, Helsinki (1983)
7. Beesley, K.R., Karttunen, L.: Finite State Morphology. CSLI, Stanford (2003)
8. Kataja, L., Koskenniemi, K.: Finite-State Description of Semitic Morphology: A Case Study of Ancient Akkadian. In: COLING 1988. Proceedings of the 12th Conference on Computational Linguistics, pp. 313–315. Association for Computational Linguistics, Morristown, NJ, USA (1988)
9. Harris, Z.: Linguistic Structure of Hebrew. Journal of the American Oriental Society 62, 143–167 (1941)
10. Buckwalter, T.: Buckwalter Arabic Morphological Analyzer Version 1.0. LDC Catalog Number LDC2002L49. Linguistic Data Consortium (2002)
11. Beesley, K.R.: Arabic Stem Morphotactics via Finite-State Intersection. In: Benmamoun, E. (ed.) Perspectives on Arabic Linguistics, XII: Papers from the Twelfth Annual Symposium on Arabic Linguistics, Benjamins, Amsterdam, pp. 85–100 (1999)
12. Beesley, K.R., Karttunen, L.: Finite-State Non-concatenative Morphotactics. In: Proceedings of the Workshop on Finite-State Phonology. 38th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Morristown, NJ, USA (2000)

13. Wright, W.: A Grammar of the Arabic Language. Munshiram Manoharlal, New Delhi (2004)
14. Bohas, G., Guillaume, J.P.: Etude des Théories des Grammairiens Arabes. Institut Français de Damas, Damas (1984)
15. Beesley, K.R.: Constraining Separated Morphotactics Dependencies in Finite-State Grammars. In: Karttunen, L., Oflazer, K. (eds.) FSMNLP 1998. Proceedings of the International Workshop on Finite State Methods in Natural Language Processing, Bilkent University, Bilkent (1998)

Appendix: Buckwalter Transliteration System

In our code we use the 'Buckwalter transliteration system' presented in [10], of which, for purposes of clarity, we outline here just a small fragment including the letters most used in this paper whose character significantly differs from any of those corresponding to it within the plethora of other systems employed in academic publications.

Table 1. A partial transliteration of Arabic characters using the Buckwalter system

Arabic character	ئ	ا	ح	ش	ض	ط	ظ	ع	ـْ
Buckwalter transliteration	}	A	H	$	D	T	Z	E	o

A Probabilistic Model for Guessing Base Forms of New Words by Analogy

Krister Lindén

Department of General Linguistics, P.O. Box 9, FIN-00014 University of Helsinki
Krister.Linden@Helsinki.fi

Abstract. Language software applications encounter new words, e.g., acronyms, technical terminology, loan words, names or compounds of such words. Looking at English, one might assume that they appear in base form, i.e., the lexical look-up form. However, in more highly inflecting languages like Finnish or Swahili only 40-50 % of new words appear in base form. In order to index documents or discover translations for these languages, it would be useful to reduce new words to their base forms as well. We often have access to analyzes for more frequent words which shape our intuition for how new words will inflect. We formalize this into a probabilistic model for lemmatization of new words using analogy, i.e., guessing base forms, and test the model on English, Finnish, Swedish and Swahili demonstrating that we get a recall of 89-99 % with an average precision of 76-94 % depending on language and the amount of training material.

1 Introduction

New words and new usages of old words are constantly finding their way into daily language use. This is particularly prominent in quickly developing domains such as biomedicine and technology. Humans deal with new words based on previous experience: we treat them by analogy to known words. The new words are typically acronyms, technical terminology, loan words, names or compounds containing such words. They are likely to be unknown by most hand-made morphological analyzers. In some applications, hand-made guessers are used for covering this low-frequency vocabulary.

Unsupervised acquisition of morphologies from scratch has been studied as a general problem of morphology induction in order to automate the morphology building procedure. For overviews, see [8] and [2]. The problem is alleviated by the fact that there often are dictionaries available with common base forms or word roots for the most frequent words. If the inflectional patterns can be learned approximately from a corpus, the most common base forms can be checked against a dictionary in order to boost the performance of the methods. However, when we approach the other end of the spectrum, we have very rare words for which there are no ready base forms available in dictionaries and for heavily inflecting languages only 40-50 % of the words appear in base form in a corpus. When new words appear for the first time, we also do not have access to several forms of the same word in order to draw on paradigmatic information.

If we do not need a full analysis, but only wish to segment the words into morph-like units, we can use a segmentation method like Morfessor [1]. For a comparison of some recent successful segmentation methods, see the Morpho Challenge [4].

Unsupervised methods have advantages for less-studied languages, but for the well-established languages, we have access to fair amounts of training material in the form of analyzes for more frequent words. There are a host of large but shallow hand-made morphological descriptions available, e.g., the Ispell collection of dictionaries [3] for spell-checking purposes, and many well-documented morphological analyzers are commercially available.

One can also argue that humans do not learn words by only observing masses of inflected forms. We are raised in a world, where we refer to similar objects and events using similar sound patterns. Context-based clustering methods have been proposed for this, but in lieu of more advanced methods for indicating words with identical or similar referents, we will use base forms for this purpose. We propose a new method for automatically learning a lemmatizer for previously unseen words, i.e., a base form guesser. This essentially places us in a supervised framework, but the novelty of the method is that we assume no knowledge of the structure of the morphology, i.e., the words could inflect word-initially or word-finally. In Section 2, we describe the probabilistic methodology. In Section 3, we present the training and test data for four morphologically distinct languages: English, Finnish, Swedish and Swahili. In Section 4, we test the model and show that the results are statistically very highly significant for all four languages. In Section 5, we discuss the method and the test results and give a note on the implementation.

2 Methodology

Assuming that we have set of word and base form pairs and another set of new previously unseen words for which we wish to determine their base forms by analogy with the known words, we first describe the probabilistic framework for our analogical model in Section 2.1. We then describe the probabilistic model for morphology in Section 2.2.

2.1 Probabilistic Framework for Analogy

Assume that we have a set of words, $w \in W$, from a text corpus for which we have determined the base forms, $b(w) \in B \subset W$, i.e. the lexicon look-up form. In addition, we have another set of words, $o \notin W$, for which we would like to determine their most likely base form, $b(o) \notin B$. For this purpose, we use the analogy that w is to o as $b(w)$ is to $b(o)$. This relationship is illustrated in Figure 1.

$$\begin{array}{ccc} w & : & o \\ \updownarrow & & \updownarrow \\ b(w) & : & b(o) \end{array} \qquad\qquad \begin{array}{ccc} \textit{kokeella} & : & \textit{aikeella} \\ \updownarrow & & \updownarrow \\ \textit{koe} & : & ? \end{array}$$

Fig. 1. The analogy w is to o as $b(w)$ is to $b(o)$ illustrated by the Finnish words *kokeella (‘with the test’)* and *aikeella (‘with the intention’)*

We use the analogical relation for deriving transformations $w \rightarrow b(w)$ from the differences between the known word and base forms. The transformations can then be applied to a new word o in order to generate a base form that should be similar to an existing base form $b(w)$. Several transformations may apply to any particular o and we wish to determine the most likely $b(o)$ in light of the evidence, i.e. we wish to find the $b(o)$, which maximizes the probability $P(o, w \rightarrow b(w), b(w), b(o))$ for the new word o. By applying the chain rule to $P(o, w \rightarrow b(w), b(w), b(o))$, we get (1),

$$\underset{b(o)}{\operatorname{argmax}} \sum_{b(w),\, w \rightarrow b(w)} \begin{pmatrix} P(o) \times \\ P(w \rightarrow b(w) \mid o) \times \\ P(b(w) \mid w \rightarrow b(w), o) \times \\ P(b(o) \mid b(w), w \rightarrow b(w), o) \end{pmatrix} \tag{1}$$

As the probability of $b(w)$ and o is independent of the other terms and the probability of o is constant, we get (2),

$$\underset{b(o)}{\operatorname{argmax}} \sum_{b(w),\, w \rightarrow b(w)} \begin{pmatrix} P(w \rightarrow b(w) \mid o) \times \\ P(b(w)) \times \\ P(b(o) \mid b(w), w \rightarrow b(w), o) \end{pmatrix} \tag{2}$$

We can also assume that the probability for a candidate base form $b(o)$ to be similar to existing base forms $b(w)$, is independent of the particular transformation $w \rightarrow b(w)$ that produced the candidate as well as of the source o of the candidate. In addition, we assume no knowledge of the distribution of the analog base forms $b(w)$, i.e. we assume an even distribution for maximum entropy. This further simplifies the expression to (3),

$$\underset{b(o)}{\operatorname{argmax}} \sum_{b(w), w \rightarrow b(w)} P(w \rightarrow b(w) \mid o) P(b(o) \mid b(w)) \tag{3}$$

Finally, we make the Viterbi approximation and assume that the probabilities of the most likely transformations and the most similar base forms are good representatives of the sums of the probabilities over all transformations and base forms giving rise to the same candidate, which gives us the equation (4),

$$\underset{b(o)}{\operatorname{argmax}}\, P(w \rightarrow b(w) \mid o) P(b(o) \mid b(w)) \tag{4}$$

We have now arrived at an expression that models the analogy between a word and its candidate base form in light of existing words and their base forms. The next step is to model the morphology of the words.

2.2 Probabilistic Framework for Morphology

When we have a model for calculating the analogy, we also need a model for the words and the morphological transformations that we can learn from the exemplars in a training corpus. We decompose the word o into consecutive substrings α, μ, ω and the candidate base form $b(o)$ into corresponding consecutive substrings β, ν, ξ, such that $\alpha \to \beta$ is a prefix transformation and $\omega \to \xi$ is a suffix transformation, whose likelihoods have been estimated from a set of pairs of words w and base forms $b(w)$ in a training corpus. (Note that the terms prefix, stem and suffix mean any string beginning, middle, or ending, respectively, and is not limited to the linguistically motivated terms prefix, stem and suffix morphemes.) Since we deal with new roots, the stem transformation $\mu \to \nu$ cannot be estimated from the corpus and needs a separate model, which we return to below. We assume that the prefix, stem and suffix, transformations can be applied independently. For the first part $P(w \to b(w) \mid o)$ of the analogy model (4), we get (5),

$$P(\alpha \to \beta \mid \alpha\mu\omega)\, P(\mu \to \nu \mid \alpha\mu\omega)\, P(\omega \to \xi \mid \alpha\mu\omega) \tag{5}$$

Assume that we have a training corpus where the characters of the word and base form pairs are aligned. Estimating the probability of the prefix $P(\alpha \to \beta \mid \alpha\mu\omega)$ and suffix $P(\omega \to \xi \mid \alpha\mu\omega)$ transformations based on the aligned training data is straight forward. The conditional probability (6) of the prefix transformation is estimated directly from the counts $C(\alpha, \beta)$ of how often the prefixes α and β correspond in the aligned word and base form data compared to the total count $C(\alpha)$ of the prefix α in the word forms,

$$P(\alpha \to \beta \mid \alpha\mu\omega) = \frac{C(\alpha, \beta)}{C(\alpha)} \tag{6}$$

The conditional probability of the suffix transformations (7) for ω and ξ are estimated in the same way:

$$P(\omega \to \xi \mid \alpha\mu\omega) = \frac{C(\omega, \xi)}{C(\omega)} \tag{7}$$

In (5), μ is likely to be a previously unseen stem, as we aim at modeling the inflections of new words. This means that we cannot really estimate its likelihood nor its transformations from a corpus. We therefore roughly model the new stem by a flat distribution (8) for the characters of the alphabet $\mu_i \in \mathrm{A}$ assuming that each character independently transforms only into itself, $\mu_i = \nu_i$. This stem model essentially assigns higher probability to shorter fragments of unknown stems. As a side-effect, we favor transformations for longer prefixes and suffixes. The size of the alphabet is $|\mathrm{A}|$ and the length of the stem μ is m .

$$P(\mu \to \nu \mid \alpha\mu\omega) = (1/|\mathrm{A}|)^m \tag{8}$$

In order to model the likelihood of a new base form on its similarity to previously seen base forms, we compare the beginning and the end of a new base form to the base forms we have in the training material. Here β is a prefix and ξ is a suffix of a known base form and ν is a new stem. For the second part $P(b(o) \mid b(w))$ of the analogy model (4), we get (9),

$$P(\beta \mid \beta\nu\xi)\, P(\nu \mid \beta\nu\xi)\, P(\xi \mid \beta\nu\xi) \tag{9}$$

Previously seen prefixes and suffixes of base forms are modeled with the conditional probability 1 and unseen prefixes and suffixes get the conditional probability 0. We again model the new word stem fragment by a flat distribution (10) for the characters of the alphabet $\nu_i \in \mathrm{A}$ assuming independence for the characters. The size of the alphabet is $|\mathrm{A}|$ and the length of the stem ν is n.

$$P(\nu \mid \beta\nu\xi) = (1/|\mathrm{A}|)^n \tag{10}$$

We now have all the components for a simple probabilistic model of inflectional morphology of unknown words, whose affix transformation parameters can be estimated from a character aligned training corpus of word and base form pairs. For details on how to align characters using a generalized edit distance alignment model, see e.g. [5].

3 Data Sets

In order to test our model in a language-independent setting, we selected four languages with different characteristics: *English*–a Germanic isolating language; *Swedish*–an agglutinating Germanic language; *Finnish*–a suffixing highly-agglutinating Fenno-Ugric language; *Swahili*–a prefixing language with a fair amount of suffixes as well. In Section 3.1, we present the corpora, from which we draw the training material as shown in Section 3.2 and test data as shown in Section 3.3. We present the baseline, measures and significance test in Section 3.4.

3.1 Corpus Data

We used publicly available text collections for the four languages: English, Finnish, Swedish and Swahili. An overview of the corpus sizes are displayed in Table 1.

For English, we used part of *The Project Gutenberg* text collection, which consists of thousands of books. For this experiment we used the English texts released in the year 2000 [http://www.gutenberg.org/dirs/GUTINDEX.00]. The texts were morphologically analyzed into 26 million running text tokens and disambiguated by the Machinese Phrase tagger [www.connexor.fi]. The tokens consisted of 266 000 forms of 175 000 base forms.

For Finnish, we used the *Finnish Text Collection,* which is an electronic document collection of the Finnish language. It consisted of 180 million running text tokens, out of which 144 million were morphologically analyzed and disambiguated. The tokens were 4.8 million inflected forms of 1.8 million base forms. The corpus contains news texts from several current Finnish newspapers. It also contains extracts from a number of books containing prose text, including fiction, education and sciences. Gatherers are the Department of General Linguistics, University of Helsinki; The University of Joensuu; and CSC - Scientific Computing Ltd. The corpus is available through CSC [www.csc.fi].

For Swedish, we used the *Finnish-Swedish Text Collection*, which is an electronic document collection of the Swedish language of the Swedish speaking minority in Finland. It consisted of 35 million morphologically analyzed and disambiguated tokens. The tokens were 765 000 inflected forms of 445 000 base forms. The corpus contains news texts from several current Finnish-Swedish newspapers. It also contains extracts from a number of books containing fiction prose text. Gatherers are The Department of General Linguistics, University of Helsinki; CSC - Scientific Computing Ltd. The corpus is available through CSC [www.csc.fi].

For Swahili, we used *The Helsinki Corpus of Swahili* (HCS), which is an annotated corpus of Standard Swahili text. It consisted of 12 million morphologically analyzed and disambiguated running text tokens. The tokens were 268 000 inflected forms of 28 000 base forms. The corpus contains news texts from several current Swahili newspapers as well as from the news site of Deutsche Welle. It also contains extracts from a number of books containing prose text, including fiction, education and sciences. Gatherers are The Department of African and Asian Studies, University of Helsinki. The corpus is available through CSC [www.csc.fi].

Table 1. Corpus data sizes in tokens, types, inflected word forms and base forms (= lexicon look-up forms)

Language	*Tokens*	*Types*	*Inflected word forms*	*Base forms*	*Infl./ Base*	*Type/ Infl.*
Finnish	144 M	3 868 K	4 178 K	1 801 000	2.3	92.3
English	26 M	210 K	222 K	175 000	1.3	94.6
Swedish	35 M	645 K	655 K	445 000	1.5	98.5
Swahili	12 M	231 K	243 K	28 000	8.6	95.1

In Table 1, the difference between the figures in the columns *Types* and *Inflected word forms* means that some word forms have more than one base form, in which case we counted them as separate inflected word forms. This means ambiguity that can only be resolved in context. The ratio between the two gives us an upper-bound on how well any algorithm that only takes the words and not their contexts into account can perform if guessing only the single most likely base form for a new word of the language. By giving several suggestions, the recall can of course go above this figure while the precision goes down.

3.2 Training Data

We have corpora with pairs of word and base forms, for which the correct base form has been mechanically identified in context. We construct our training material by ordering the word and base form pairs according to decreasing frequency and divide the training material into four top frequency ranks as shown in Tables 2a-d.

3.3 Test Data

We draw 5000 word and base form pairs from the frequency rank 100 001-300 000 as test material. The test data frequency ranks can be seen in Table 3.

Table 2a. Top frequency ranks of inflected word and base form pairs for English

Frequency rank	*Number of inflected forms*	*Cum. number of inflected forms*	*Number of base forms*	*Cum. number of base forms*
1- 3 000	2 720	2 720	2 176	2 176
3 001- 10 000	6 143	8 863	4 566	6 742
10 001- 30 000	16 753	25 616	12 851	19 593
30 001-100 000	54 218	79 834	44 402	63 995

Table 2b. Top frequency ranks of inflected word and base form pairs for Finnish

Frequency rank	*Number of inflected forms*	*Cum. number of inflected forms*	*Number of base forms*	*Cum. number of base forms*
1- 3 000	2 781	2 781	1 587	1 587
3 001- 10 000	6 346	9 127	2 633	4 220
10 001- 30 000	17 757	26 884	6 601	10 821
30 001-100 000	61 139	88 023	21 301	32 122

Table 2c. Top frequency ranks of inflected word and base form pairs for Swedish

Frequency rank	*Number of inflected forms*	*Cum. number of inflected forms*	*Number of base forms*	*Cum. number of base forms*
1- 3 000	2 761	2 761	1 898	1 898
3 001- 10 000	6 351	9 112	3 764	5 662
10 001- 30 000	17 593	26 705	9 938	15 600
30 001-100 000	58 629	85 334	33 049	48 649

Table 2d. Top frequency ranks of inflected word and base form pairs for Swahili

Frequency rank	*Number of inflected forms*	*Cum. number of inflected forms*	*Number of base forms*	*Cum. number of base forms*
1- 3 000	2 619	2 619	2 012	2 012
3 001- 10 000	5 985	8 604	3 208	5 220
10 001- 30 000	17 116	25 720	4 708	9 928
30 001-100 000	60 512	86 232	7 485	17 413

Table 3. Test data frequency rank 100 001-300 000 of inflected word and base form pairs

Frequency rank 100 001-300 000	*Number of inflected forms*	*Number of base forms*
English	127 359	111 665
Finnish	170 725	57 306
Swedish	165 929	109 283
Swahili	144 964	10 497

3.4 Baseline and Significance Tests

We report our test results using recall and average precision at maximum recall. Recall means all the inflected word forms in the test data for which an accurate base form suggestion is produced. Average precision at maximum recall is an indicator of the amount of noise that precedes the intended base form suggestions, where *n* incorrect suggestions before the *m* correct ones give a precision of *1/(n+m)*, i.e., no noise before a single intended base form per word form gives 100 % precision on average, and no correct suggestion at maximum recall gives 0 % precision. All figures are reported with their 99 % confidence intervals. This means that corresponding test results with non-overlapping confidence intervals are statistically very significantly different.

The baseline assumption is that new words appear in their base form, i.e., we need not do anything. We tested the baseline hypothesis drawing 5000 word and base form pairs at random from the test data frequency rank in Table 3. Since we are only interested in words that we have not seen in the training material, we only count inflected forms of new base forms. As no more than one suggestion is available for each word form in our baseline test, the average baseline precision at maximum recall is identical to the recall in Table 4.

Table 4. Baseline precision and recall for 5000 words drawn from the test data frequency rank

Language	*New words (with unseen base form)*	*New words in base form*	*Precision & Recall in % ± confidence*
English	2912	2508	86.1 ±0.7
Finnish	2081	1051	50.5 ±1.6
Swedish	3395	2043	60.2 ±1.2
Swahili	384	159	41.4 ±3.7

As can be seen from the baseline experiment, around 86 % of the new words in English appear in their base form, whereas the corresponding figures for Swedish is around 60 %, for Finnish around 50 % and for Swahili around 40 %.

4 Experiments

We test how well the analogical guesser is able to predict base forms for new words using the test data for which we calculated the baseline in Section 3.3. The sensitivity of the model is tested using increasing amounts of training data. The model makes no particular assumptions about the language except that the inflections are encoded as prefixes and/or suffixes which may cover parts of the stem if the stem also changes. We test the model on the new words of the test data using various amounts of training material. The amounts of training data and the corresponding results can be seen in Tables 5a-d.

Table 5a. Recall and average precision in the test frequency rank 100 000-300 000 for English

Training data frequency ranks	*Found correct test base forms*	*Recall in % ± confidence*	*Avg. precision in % ± confidence*
1- 3 000	2734	93.8±0.3	74.9±0.7
1- 10 000	2764	94.8±0.3	78.2±0.6
1- 30 000	2781	95.5±0.2	81.3±0.5
1-100 000	2834	97.5±0.1	88.2±0.4

Table 5b. Recall and average precision in the test frequency rank 100 000-300 000 for Finnish

Training data frequency ranks	*Found correct test base forms*	*Recall in % ± confidence*	*Avg. precision in % ± confidence*
1- 3 000	1964	91.2±0.5	74.2±0.8
1- 10 000	2021	93.9±0.4	78.5±0.7
1- 30 000	2051	95.3±0.3	80.4±0.7
1-100 000	2033	94.4±0.3	79.1±0.7

Table 5c. Recall and average precision in the test frequency rank 100 000-300 000 for Swedish

Training data frequency ranks	*Found correct test base forms*	*Recall in % ± confidence*	*Avg. precision in % ± confidence*
1- 3 000	3294	97.5±0.1	86.2±0.4
1- 10 000	3341	98.9±0.1	88.9±0.3
1- 30 000	3358	99.5±0.05	91.8±0.2
1-100 000	3351	99.2±0.05	94.4±0.2

Table 5d. Recall and average precision in the test frequency rank 100 000-300 000 for Swahili

Training data frequency ranks	*Found correct test base forms*	*Recall in % ± confidence*	*Avg. precision in % ± confidence*
1- 3 000	291	76.0±2.8	69.4±2.8
1- 10 000	320	83.6±2.1	75.7±2.3
1- 30 000	339	89.7±1.4	79.7±1.9
1-100 000	338	89.4±1.5	76.3±2.1

Table 6. Relative improvement over the baseline precision and recall with the maximum amount of training data

Language	*Recall*	*Precision*
English	+13.2 %	+2,4 %
Finnish	+86.9 %	+ 56.6 %
Swedish	+64.8 %	+56.8 %
Swahili	+115,9 %	+84.3 %

4.1 Importance of the Results

The test results are statistically very highly significant and the test results also indicate that the relative improvements over the baseline are interesting in practice for all four languages as shown in Table 6.

5 Discussion

In this section, we discuss the test results and give some final notes on the nature of morphologies and the implementation of the model.

We used incremental amounts of training material, i.e. we successively added training material from a new range of ranks while testing on data from the frequency ranks 100 001-300 000. As might be expected, there were successive improvements with additional data. We note, however, that most of the improvements in recall were achieved already with as little training data as the 3 000 most common word and base form pairs of a language and after 10 000 small but significant improvements remained. After the 30 000 most frequent data pairs have been used as training material, the improvements in recall began leveling off. The precision continues to increase for two of the four languages with additional training material. From this, we conclude that using slightly more than the core set of word forms and their corresponding base forms is enough to automatically induce a reasonable guesser for a language. This observation is also true with some caution for human speakers of a language. As an aside, it should be noted that manually editing the most likely base form for each word form in a list of the 30 000 most frequent word forms is a tedious task, but it only takes a week or two for a native linguist.

For most languages, the inflections are affixed to the end or to the beginning of a word stem with some possible minor modification of the stem at the junction. Here Arabic is the most prominently used counter example, for which word inflections are indicated with stem internal vocalization patterns in addition to using affixes. However, the vocalizations are not customarily marked in text except in the Qur'an. In addition, stem changes derive new words, i.e. they are derivational processes of the language not inflectional, e.g., relating words like *book*, *read* and *reader*, and they therefore tend to be lexicalized. However, it remains to be seen whether the model applies as successfully to written Arabic as well.

The model for finding the most likely analog base form for a new word form was implemented with a cascade of weighted finite-state transducers–one for each part of the model. The cascade is composed with the word form at runtime. To extract the most likely base forms, we make a projection of the upper surface of the composed transducer and list the N-best unique base forms, i.e., the N base forms with the smallest total log-probability weights. The weighted transducers can be implemented in the tropical semiring, where finding the string with the highest probability coincides with the single source shortest distance algorithm. Open Source tools for weighted finite-state transducers have been implemented by, e.g., [6] and [7].

6 Conclusion

We have introduced a new probabilistic model for determining base forms for previously unseen words by analogy with a set word and base form pairs. The model makes no assumptions about whether the inflections are encoded word-initially or word-finally. We tested the model on four morphologically different languages: English, Finnish, Swedish and Swahili. Our model reached a recall of 89-99 % with an average precision of 76-94 % depending on language and the amount of training material. The model was statistically very highly significantly better than the baseline for all four languages and the relative improvement over the baseline was considerable both for recall and precision. From our experiments, it seems like using slightly more than the core set of word forms found in a corpus paired with their base forms would be enough to mechanically induce a reasonable base form guesser, i.e., a lemmatizer for new words of a language.

References

1. Creutz, M., Lagus, K., Lindén, K., Virpioja, S.: Morfessor and Hutmegs: Unsupervised Morpheme Segmentation for Highly-Inflecting and Compounding Languages. In: Proceedings of the Second Baltic Conference on Human Language Technologies, Tallinn, Estonia, April 4–8 (2005)
2. Goldsmith, J.: Morphological Analogy: Only a Beginning (2007), http://hum.uchicago.edu/~jagoldsm/Papers/analogy.pdf
3. Kuenning, G.: Dictionaries for International Ispell (2007), http://www.lasr.cs.ucla.edu/geoff/ispell-dictionaries.html
4. Kurimo, M., Creutz, M., Turunen, V.: Overview of Morpho Challenge in CLEF 200. In: Nardi, A., Peters, C. (eds.) Working Notes of the CLEF 2007 Workshop, Budapest, Hungary, September 19–21 (2007)
5. Lindén, K.: Multilingual Modeling of Cross-lingual Spelling Variants. Journal of Information Retrieval 9, 295–310 (2006)
6. Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., Mohri, M.: OpenFst: A General and Efficient Weighted Finite-State Transducer Library. LNCS (to appear)
7. Lombardy, S., Régis-Gianas, Y., Sakarovitch, J.: Introducing Vaucanson. Theoretical Computer Science 328, 77–96 (2004)
8. Wicentowski, R.: Modeling and Learning Multilingual Inflectional Morphology in a Minimally Supervised Framework. PhD Thesis. Baltimore, Maryland (2002)

Unsupervised and Knowledge-Free Learning of Compound Splits and Periphrases

Florian Holz and Chris Biemann

NLP Group, Department of Computer Science, University of Leipzig
{holz|biem}@informatik.uni-leipzig.de

Abstract. We present an approach for knowledge-free and unsupervised recognition of compound nouns for languages that use one-word-compounds such as Germanic and Scandinavian languages. Our approach works by creating a candidate list of compound splits based on the word list of a large corpus. Then, we filter this list using the following criteria:
(a) frequencies of compounds and parts,
(b) length of parts.

In a second step, we search the corpus for periphrases, that is a reformulation of the (single-word) compound using the parts and very high frequency words (which are usually prepositions or determiners). This step excludes spurious candidate splits at cost of recall. To increase recall again, we train a trie-based classifier that also allows splitting multi-part-compounds iteratively.

We evaluate our method for both steps and with various parameter settings for German against a manually created gold standard, showing promising results above 80% precision for the splits and about half of the compounds periphrased correctly. Our method is language independent to a large extent, since we use neither knowledge about the language nor other language-dependent preprocessing tools.

For compounding languages, this method can drastically alleviate the lexicon acquisition bottleneck, since even rare or yet unseen compounds can now be periphrased: the analysis then only needs to have the parts described in the lexicon, not the compound itself.

1 Introduction

A number of languages extensively use compounding as an instrument of combining several word stems into one (long) token, e.g. Germanic languages, Korean, Greek and Finnish. Compared to languages such as English, where (noun) compounds are expressed using several tokens, this leads to a tremendous increase in vocabulary size. In applications, this results in sparse data, challenging a number of NLP applications. For IR experiments with German, Braschler et al. report that decompounding results in higher text retrieval improvements than stemming [1].

As an example, consider the German compound "Prüfungsvorbereitungsstress" (stress occurring when preparing for an exam) - without an analysis, this word can neither be translated in an MT system nor found by a search query like "Stress AND Prüfung" (stress AND exam).

Several approaches have been used to alleviate this threat by analyzing and splitting compounds into their parts, which will be reviewed in the next section. Then, we describe our approach of finding not only the correct splits, but also periphrases (a sequence of tokens with the same semantic interpretation, usually a noun phrase). Especially for long compounds, these periphrases exist in large corpora - our example is more commonly expressed as "Stress bei der Prüfungsvorbereitung".

During the whole process, which his described in Sect. 2, we do neither assume the existence of language-specific knowledge nor language-specific preprocessing tools. We argue that before augmenting the process with additional sources of information that might blur evaluation of the basic method, we first want to provide a language-independent baseline.

Section 3 presents experimental results for German, Section 4 concludes.

1.1 Literature Review

Approaches to compound noun splitting can be divided into knowledge-intensive and knowledge-free approaches. In knowledge-intensive approaches, either supervised learning from a training set is used to create a splitter, or a set of handcrafted rules performs the task. Not surprisingly, most studies on compound noun splitting report experiments with German, which is the compounding language with most speakers. For German compounding rules, see [6].

Knowledge-free approaches are in principle independent of language-specific knowledge and try to induce a compound splitter by analyzing the word list (types) of a raw text corpus.

Perhaps the most straightforward knowledge-free approach is described in [8]: In the analysis of a compound candidate, all prefixes of the candidate are matched against the word list. Once a prefix is found, a split is performed and the remaining suffix is subject to further analysis. The authors report 60% precision and 50% recall evaluation on correct split positions for a set of around 700 complex nouns. The main problem of this approach is caused by too many splits due to short words in the word list (e.g. "Prüf" for the example) that also cause the subsequent splits to fail. What would be needed to repair spurious splits are more clues about the semantic composition of compounds. Nevertheless, significant improvements in a retrieval task were obtained.

Larson et al. train a letter-n-gram classifier on word boundaries in a large corpus and use it to insert compound part boundaries, successfully reducing the out-of-vocabulary rate in their speech recognition system, but not improving speech recognition accuracy [7]. Here, no evaluation on the compound splitting itself is given.

Our approach falls into the knowledge-free paradigm. In the remainder of this section, we discuss some knowledge-intensive methods, like handcrafted or trained morphological analyzers, for completeness.

E. g. Finkler et al. provide several splitting options and leave the choice to a post-processing component [4]. Comparable to approaches for the related task of Chinese word segmentation, Schiller uses weighed finite-state transducers, resulting in over 90% precision and almost perfect recall [10]. Sjöbergh et al. modified a spell checker to find and split compounds in a small Swedish text with very good results [11]. Yun et al. pursue a hybrid rule-based and statistical approach for Korean Compound noun splitting [14].

When translating between compounding and non-compounding languages in a Machine Translation system, productive compounds cannot be enumerated in the word list; moreover, they usually translate to two or more words or even phrases [5]. Parallel text can be employed to get clues about translations of compounds and to use the back-translation of parts for splitting, as in [2] and [5], who report over 99% accuracy when using POS-tagged parallel corpora. However, these methods need aligned parallel text, which might not be sufficiently available for all compounding languages in all relevant domains.

We know about one unsupervised approach to judge the suitability of descriptive patterns for given word pairs [12]. Here corpus-based frequencies of the word pairs and the patterns in which the word pairs appear in the corpus are used to rank the found or given patterns for a word pair. The patterns are taken to represent the relation between the to words. So for a word pair its inner relation can be identified using best fitting pattern or for a word pair another word pair out of a given set can be identified to resemble the same relation (SAT test). The evaluation of the first task is done with 600 manually labelled word pairs where for every pair the other 599 serve as training data to classify the choosen one. The other task is evaluated on 374 college-level multiple-choice word analogies. Both evaluations show results between 50% and 60% for precision, recall and the F1-measure.

1.2 Motivation for Our Method

Even if correct splits can be found automatically with high precision, the question arises how to interpret the results. Dependent on the application, extra information might be needed. Consider e.g. the two German compounds "Schweine-schnitzel" (pork cutlet) and "Kinder-schnitzel" (small cutlet for kids, literally kids cutlet). In e.g. semantic parsing, it might be advantageous to know that a "Kinderschnitzel" is made for kids and not out of kids, as in the case of pork. We therefore propose to find periphrases, e.g. "Schnitzel vom Schwein" and "Schnitzel für Kinder" and offer these to the interpreting engine. Periphrases do not only add more valuable information, they will also be employed to find the correct splits: If no periphrasis for a candidate split exists, it is more likely to be a spurious candidate. To the best of our knowledge, no previous approaches to automatically finding periphrases without using manual resources for compounds are reported in the literature.

2 Method

In this section we outline the steps of our method, which builds entirely on a word list with frequencies and a sentence list obtained from a massive raw text corpus. Parameters in the process are explained in the moment they arise first in our outline and summarized in Tab. 1. The whole workflow is shown in Fig. 1.

Table 1. Resources, parameters and abbreviations

Resources	corpus word list	CWL
	corpus word frequencies	CWF
	corpus sentence list	CSL
Parameters	minimum morpheme length	mM
	minimum word (compound) frequency	mFr
	minimum morpheme frequency	mTFr
	number of split parts	mA
	distance between parts in periphrases	d

Preprocessing. As preprocessing, we assume a word list (types) with frequencies from a massive monolingual raw corpus in lowered capitalization. Since we do not use parts-of-speech or other information, we always operate on the full list, where words with dashes and other special characters, as well as numbers, are removed beforehand.

Candidate Split Extraction. As a first step, we try to find candidate splits generating all possible splits and checking whether the resulting parts are contained in the word list. Candidate split determination is parameterized by a minimum length of parts (mM) and by a minimum frequency for the compound candidate (mFr), arguing that very rare words are more likely to be typing errors than valid compounds. This is compareable to the approach undertaken by [8].

Candidate Filtering. The list of candidates is then filtered according to various criteria. We applied a maximum number of parts (mA), a minimum frequency of parts (mTFr). If several splits for one compound candidate exist, we select the split with the highest geometric mean of part frequencies (quality measure taken from [5]).

After this step, we have a list of compounds with splits that can already be evaluated. Results characteristically show high precision but low recall – The filtering succeeds in finding good splits, but only for a small fraction of total types (cf. Tab. 6 and 7).

Generalized Splitter. To overcome the recall problem, we train two trie-based classifiers on the filtered candidate splits to generalize possible parts over the full word list. If we, for example, included the correct split "Kinder-schnitzel" but not "Schweine-schnitzel", the second split will be classified based on the assumption that splitting the suffix "schnitzel" is generally correct. Training is done for prefixes and suffixes separately and allows constructing a recursive splitter that can be applied to arbitrary word lists. Table 2 shows a small training sample for the prefix and suffix classifier. When classifying, the input is matched against the longest prefix (suffix) stored in the trie, and the corresponding class distribution is returned. Classes denote here the number of letters that should be separated from the beginning (ending). Note that the suffix classifier internally works on reversed strings. For more detailes on patricia-tree-based classifying, see [13].

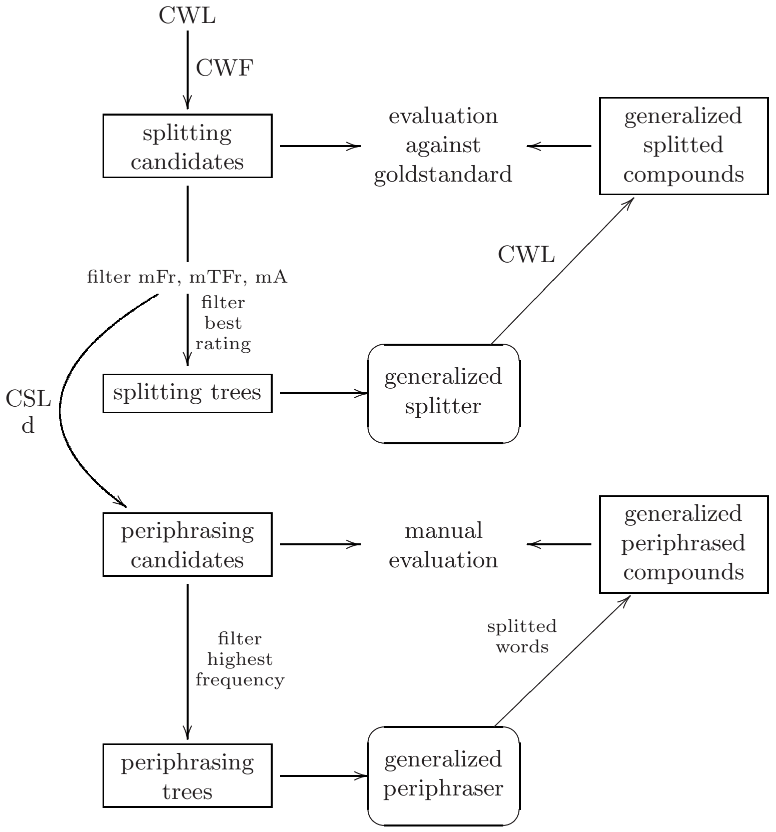


Fig. 1. Workflow: From the corpus word list (CWL), we build candidate splits and a generalized splitter; both are evaluated against the gold standard splits. From the candidate splits, we obtain periphrasis candidates and build a generalized periphraser; both are evaluated manually.

Table 2. Training set example for the prefix and suffix trie-based classifiers

Word	Split	Prefix String	Prefix Class	Suffix String	Suffix Class
Holzhaus	Holz-haus	Holzhaus	4	suahzloH	4
Berggipfel	Berg-gipfel	Berggipfel	4	lefpiggreB	6
Hintergedanke	Hinter-gedanke	Hintergedanke	6	eknadegretniH	7

Periphrase Detection. For a list of candidate splits, be it the list after candidate filtering or the list resulting from the application of the generalized splitter on the full word list, we set out to find periphrases in our corpus. Periphrases contain all parts, with at most d function words between them as intermediate fillers. For approximating function words, we allow the 200 most frequent words in this position.

For each candidate periphrasis per compound, we accept the periphrasis with the highest corpus frequency. Naturally, not all compounds are actually expressed as periphrases in the corpus. If strong evidence for a periphrasis is found, then the corresponding candidate split can be assumed to be valid with even higher confidence.

Generalized Periphrases. Based on the assumption that similar prefixes and suffixes are periphrased similarly, we train two other trie-based classifiers that learn the distribution of intermediate fillers per prefix and suffix. E.g. if we found the periphrasis "Hersteller von Aluminium" (manufacturer of aluminum) for the compound "Alminiumhersteller", then the prefix (suffix) classifier learns that "von" is likely to be used on the preceding (subsequent) position "Aluminium" ("hersteller") once.

Using this information, periphrases for arbitrary compound splits can be obtained by applying the classifiers on the parts. If intermediate filler information for both parts of a split is consistent, i.e. both classifiers overlap on the intermediate fillers, then the common filler is proposed which maximizes $\log(f_{\mathrm{p}1} + 1) * \log(f_{\mathrm{p}2} + 1)$ where $f_{\mathrm{p}i}$ is the number of occurrences of this filler with the part pi in the training data. In case of contradictory results, the most frequent filler is returned. Table 3 shows example periphrases and the training information for both prefix and suffix splits.

Table 3. Sample training set for the trie-based classifiers for intermediate fillers

Word	Periphrase	Right-position Trie	Right-position Class	Left-position Trie	Left-position Class
Aluminiumhersteller	Hersteller von Aluminium	Aluminium	von	Hersteller	von
Arbeitsaufwand	Aufwand bei der Arbeit	Arbeit	bei der	Aufwand	bei der

3 Experiments

3.1 Corpus Data

As corpus, we used the German corpus of Projekt Deutscher Wortschatz which comprises almost 1 billion tokens in about 50 million sentences [9]. The corpus was tokenized and preprocessed as described in the previous section.

3.2 Evaluation Data

Unfortunately, there is no publicly available standard dataset for German compound noun decomposition we are aware of. To judge the quality of our method, we created such a set, which will be available for download upon publication.

We report results on two evaluation sets: The first set is the CELEX database for German with a size of 90 077 splitted nouns [3]. Since here, all morphologically separable parts are marked as splits (also case endings etc.), we cannot hope for a high recall on this set. Nevertheless, we aim at high precision, since compound splits are marked (amongst many more splits). The second test set was created manually by the authors: it comprises 700 long German nouns of 14 different frequency bands from frequency = 1 up to frequency = 2^{13}. In this set, there are 13 words that are not compounds and should not be split, 640 items are compounds consisting of two pairs and 47 items consist of 3 parts - thus, we evaluate on 737 split positions.

Precision and Recall are defined as usual: Precision is obtained by dividing the number of correct splits by the number of splits taken by the method, recall is the number of correct splits divided by the total number of splits in the evaluation set. We further report results in the F1 measure, which is the harmonic mean of Precision and Recall.

3.3 Results

In our experiments we used parametrizations with the following values: mM = 4 (which has shown to be a good choice in premliminary experiments), mFr = $1, 2, 3$, mTFr = $1, 2, 5, \mathrm{Fr}$, and d = $1, 2, 3$. Here, mTFr = Fr means, that every identified part of a compound has to be at least as frequent as the whole compound candidate.

Splits. The counts of the splitting candidates are shown in Tab. 4 and 5. Table 4 shows the total numbers of splitting candidates after the search of the parts in the corpus and the number after filtering the best of different splits for the same compound by computing the geometric mean of the part frequencies (cf. [5]). Table 5 shows how many of these candidates could be found in the gold standard sets.

Tables 6 and 7 demonstrate that a higher threshold on minimum part frequency (mTFr) leads to higher Precision, since e.g. typing errors and spurious words from the word list are excluded. mTFr = Fr is not a viable option, because the most compound candidates are very low frequent so that the compound parts can also be very low frequent and thus spurious. However, this gain in Precision is traded of with a low Recall, as Tables 4 and 5 indicate.

Table 4. Total number of candidate splits

mM	mFr	mTFr							
		1		2		5		Fr	
		total	best	total	best	total	best	total	best
4	1	3114058	2490633	1554977	1405486	685604	653691	2051581	1710628
	2	1460443	1147076	719961	648802	309876	295624	397966	367071
	3	1013859	789987	496302	447016	211482	201983	174307	165285

Table 5. The number of candidate splits found in the gold standard sets

mM	mFr	mTFr							
		1		2		5		Fr	
		700	CELEX	700	CELEX	700	CELEX	700	CELEX
4	1	642	35948	362	19387	162	8094	244	11812
	2	474	28244	271	15252	129	6357	76	4108
	3	436	25230	249	13602	121	5625	54	2458

Table 6. Evaluation of candidate splits against the 700 manually splitted nouns

mM	mFr	mTFr											
		1			2			5			Fr		
		prec	rec	F1	prec	rec	F1	prec	rec	F1	prec	rec	F1
4	1	0.84	0.73	0.78	0.87	0.43	0.58	0.87	0.19	0.31	0.81	0.27	0.40
	2	0.85	0.54	0.66	0.86	0.31	0.46	0.85	0.15	0.25	0.82	0.08	0.15
	3	0.86	0.51	0.64	0.86	0.29	0.44	0.86	0.14	0.24	0.83	0.06	0.11

Table 7. Evaluation of candidate splits against CELEX

mM	mFr	mTFr											
		1			2			5			Fr		
		prec	rec	F1	prec	rec	F1	prec	rec	F1	prec	rec	F1
4	1	0.60	0.16	0.25	0.64	0.09	0.16	0.71	0.04	0.08	0.57	0.05	0.09
	2	0.63	0.13	0.21	0.66	0.07	0.13	0.73	0.03	0.06	0.69	0.02	0.04
	3	0.64	0.12	0.19	0.67	0.07	0.12	0.73	0.03	0.06	0.75	0.01	0.03

Table 8. Evaluation of generalized splittings against the 700 manually splitted nouns

mM	mFr	mTFr											
		1			2			5			Fr		
		prec	rec	F1	prec	rec	F1	prec	rec	F1	prec	rec	F1
4	1	0.54	0.59	0.57	0.52	0.49	0.5	0.44	0.36	0.39	0.53	0.54	0.54
	2	0.58	0.66	0.61	0.52	0.51	0.52	0.42	0.36	0.39	0.53	0.48	0.5
	3	0.56	0.68	0.62	0.5	0.52	0.51	0.41	0.37	0.39	0.5	0.43	0.46

Nor surprising, using the generalized splitter increases recall (compare figures in Tab. 6 with Tab. 8 and Tab. 7 with Tab. 9).

In summary, if one aims at high-quality splits but it is not required that all compounds get splitted, then a restrictive filter on candidate splits should be preferred. If the objective is to maximize F1 as Precision-Recall tradeoff, then the generalized splitter is the option to pursue.

Periphrases. To our knowledge, this is the first research aiming at automatically constructing periphrases for noun compounds from corpora. Thus, we have to manually evaluate our findings. A periphrasis was counted as correct if it

Table 9. Evaluation of generalized splittings against CELEX

mM	mFr	mTFr											
		1			2			5			Fr		
		prec	rec	F1	prec	rec	F1	prec	rec	F1	prec	rec	F1
4	1	0.50	0.21	0.29	0.47	0.17	0.25	0.44	0.13	0.20	0.47	0.17	0.25
	2	0.52	0.23	0.31	0.49	0.18	0.26	0.42	0.14	0.21	0.49	0.16	0.24
	3	0.52	0.23	0.32	0.48	0.18	0.26	0.41	0.14	0.20	0.46	0.15	0.23

was a grammatical noun phrase and its interpretation matches the default interpretation of the compound. E.g. ”Kameraleute” (camera crew) is correctly periphrased as ”Leute hinter der Kamera” (people behind the camera). ”Leute vor der Kamera” (people in front of the camera) would be semantically wrong, ”Leute zwischen Kamera” (people between camera) is not a grammatical noun phrase.

When taking the gold standard splits from our reference set of 700 words, our program gathered 216 periphrase candidates from the corpus, of which 160 were correct – a precision of 74%, recall at 23%, F1 at 35%. Using the generalized periphraser on the gold standard splits yielded 267 correct and 433 incorrect periphrases. We observed that some of the periphrases were correct as candidates and wrong for the generalized periphraser, so we propose the following setup:

1. If a candidate periphrase is found in the corpus, then accept it.
2. Otherwise, apply the generalized periphraser.

This experiment resulted in 336 correct and 364 incorrect periphrases (Precision = Recall = F1 = 336/700 = 48%).

4 Conclusions and Further Work

We discussed several ways to approach the problem of long one-word-compounds for some languages, which causes a high OOV rate in numerous NLP applications. First, we discussed how to split compounds into their respective parts: Similar to previous approaches like [8], we extract candidate splits by checking possible concatenations of short words against the long compounds, and rank several possible splits according to the geometric mean of the parts' frequencies as in [5] and propose to filter the candidate list according to criteria on frequency of parts and compounds. Since good precision values can only be obtained in hand with low recall, we propose to build a generalized splitter, taking the candidate splits as training. In this way, we increase overall F1. In a second experiment, we aim at periphrasing compounds to resolve their semantics. For this, we search our corpus for short snippets starting and ending with compound parts and having only few intermediate stopword fillers. Here, we are able to extract periphrases for our evaluation set with 74% precision and 23% recall (F1 = 35%). In a similar setup as in the splitting experiments, we train a generalized periphraser from all periphrases found in the corpus, again improving on F1 up to 48% by increasing recall and some loss in precision.

This is, to our knowledge, the first attempt to finding periphrases for compounds from corpora in a completely unsupervised and knowledge-free fashion. Our work can serve as a baseline for further experiments in this direction. For further work, we propose the following:

– Checking the splits according to existence of corpus periphrasis and using only splits that yielded a periphrasis for training of the generalized splitter. This could increase the quality of the generalized splits.
– Extenstion to *Fugenelemente*: Many compounds in Germanic languages contain the letter ”s” between parts for reasons of easier pronounciation, e.g. in ”Prüfung*s*vorbereitung*s*stress”. Until now, this language feature is ignored. A possibility would be to explicitely provide a list of these very few fillers; however, more interesting would be to find them automatically as well.
– Evaluation of different filters for candidate splits and different measures for periphrasis selection
– Experiments for other languages, incuding more compounding languages, but also non-compounding ones for sanity-check.

References

1. Braschler, M., Ripplinger, B.: Stemming and decompounding for german text retrieval. In: Sebastiani, F. (ed.) ECIR 2003. LNCS, vol. 2633, pp. 177–192. Springer, Heidelberg (2003)
2. Brown, R.D.: Corpus-driven splitting of compound words. In: Proceedings of the 9th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI) (2002)
3. Burnage, G., Harald Baayen, R., Piepenbrock, R., van Rijn, H.: CELEX: a guide for users. CELEX (1990)
4. Finkler, W., Neumann, G.: Morphix. a fast realization of a classification-based approach to morphology. In: 4. Österreichische Artificial-Intelligence-Tagung. Wiener Workshop - Wissensbasierte Sprachverarbeitung (1998)
5. Koehn, P., Knight, K.: Empirical methods for compound splitting. In: Proceedings of EACL, Budapest, Hungary, pp. 187–193 (2003)
6. Langer, S.: Zur Morphologie und Semantik von Nominalkomposita. In: Tagungsband der 4. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS) (1998)
7. Larson, M., Willett, D., Köhler, J., Rigoll, G.: Compound splitting and lexical unit recombination for improved performance of a speech recognition system for german parliamentary speeches. In: Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP) (2000)
8. Monz, C., de Rijke, M.: Shallow morphological analysis in monolingual information retrieval for dutch, german, and italian. In: Peters, C., Braschler, M., Gonzalo, J., Kluck, M. (eds.) CLEF 2001. LNCS, vol. 2406, pp. 262–277. Springer, Heidelberg (2002)
9. Quasthoff, U., Richter, M., Biemann, C.: Corpus portal for search in monolingual corpora. In: Proceedings of the LREC (2006)
10. Schiller, A.: German compound analysis with wfsc. In: Yli-Jyrä, A., Karttunen, L., Karhumäki, J. (eds.) FSMNLP 2005. LNCS (LNAI), vol. 4002, Springer, Heidelberg (2006)

11. Sjöbergh, J., Kann, V.: Finding the correct interpretation of swedish compounds – a statistical approach. In: Proceedings of LREC, Lisbon, Portugal (2004)
12. Turney, P.D.: Expressing implicit semantic relations without supervision. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (Coling/ACL-06), Sydney, Australia, pp. 313–320 (2006)
13. Witschel, F., Biemann, C.: Rigorous dimensionality reduction through linguistically motivated feature selection for text categorisation. In: Proceedings of NODALIDA (2005)
14. Yun, B.-H., Lee, H., Rim, H.-C.: Analysis of korean compound nouns using statistical information. In: Proceedings of the 22nd Korea Information Science Society Spring Conference (1994)

German Decompounding in a Difficult Corpus

Enrique Alfonseca, Slaven Bilac, and Stefan Pharies

Google, Inc.
{ealfonseca,slaven,stefanp}@google.com

Abstract. Splitting compound words has proved to be useful in areas such as Machine Translation, Speech Recognition or Information Retrieval (IR). In the case of IR systems, they usually have to cope with noisy data, as user queries are usually written quickly and submitted without review. This work attempts at improving the current approaches for German decompounding when applied to query keywords. The results show an increase of more than 10% in accuracy compared to other state-of-the-art methods.

1 Introduction

The so-called compounding languages, such as German, Dutch, Danish, Norwegian or Swedish, allow the generation of complex words by merging together simpler ones. So, for instance, the German word *Blumensträuße* (flower bouquet) is made up of *Blumen* (flower) and *sträuße* (bouquet). This allows speakers of these languages to easily create new words to refer to complex concepts by combining existing ones, whereas in non-compounding languages these complex concepts would normally be expressed using multiword syntactic constituents.

For many language processing tools that rely on lexicons or language models it is very useful to be able to decompose compounds to increase their coverage. In the case of German, the amount of compounds in medium-size corpora (tens of millions of words) is large enough that they deserve special handling: 5-7% of the tokens and 43-47% of the word forms in German newswire articles are compounds [1,2]. When decompounding tools are not available, language processing systems for compounding languages must use comparatively much larger lexicons [3]. German decompounders have been used successfully in Information Retrieval [4,5], Machine Translation [6,7,8], word prediction systems [1] and Speech Recognition [3,9].

When decompounding German words from well-formed documents that have undergone editorial review, one can assume that most of the words or compound parts can be found in dictionaries and thesauri and the number of misspellings is low, which greatly simplifies the problem of decompounding. For example, Marek [10] observed that only 2.8% of the compounds in texts from the German computer magazine called *c't* contain at least one unknown part.

On the other hand, when working with web data, which has not necessarily been reviewed for correctness, many of the words are more difficult to analyze. This includes words with spelling mistakes, and texts that, being mostly written

Table 1. Some examples of words found on German user queries that can be difficult to handle by a German decompounder

Word	**Probable interpretation**
achzigerjahre	achtzig+jahre (misspelled)
activelor	*unknown word*
adminslots	admin slots (English)
agilitydog	agility dog (English)
akkuwarner	*unknown word*
alabams	alabama (misspelled)
almyghtyzeus	almyghty zeus (English and misspelled)
amaryllo	amarillo (Spanish and misspelled)
ampihbien	amphibian (misspelled)

in German, contain small fragments in other languages. This problem exists to a larger degree when handling user queries: they are written quickly, not paying attention to mistakes, and the language used for querying can change during the same query session. Language identification of search queries is a difficult problem: does a user looking for *Windows* on a German search engine want English pages about windows, or German pages describing the operating system? Is the query *with or without you liedtexte* a German or an English query? Therefore we cannot rely too much on automatic language identification procedures to filter out all foreign words. Table 1 shows a few examples of unexpected words that can typically be found as search keywords in a German search engine. A system for automatically identifying and splitting German compounds ideally should be able to handle this kind of noise.

This paper presents a new method for German word decompounding able to handle noisy data like this. Its structure is as follows: Section 2 describes related work in German decompounding; Section 3 defines the evaluation settings, and Section 4 describes the approaches followed in this paper. Finally, Sections 5 and 6 discuss the results and conclusions.

2 Decompounding German Words

Compound words in German are formed by merging together nouns, verbs and adjectives, and the compounds themselves can be of any of these kinds. Some examples are *Kontrollfunktion* (control function), *Zivilbevölkerung* (civilian population), *hauchdünn* (very thin) or *kondolenzbesuchen* (to visit as an expression of sympathy with someone's grief). The last word inside a compound is commonly called the *head*, and the words before it are called its *modifiers*.

When a word acts as a modifier, in some cases an additional morpheme is added to its canonical form, called *fugenmorphem* (linking morpheme). Some of them are paradigmatic morphemes (e.g. plural or genitive inflections), e.g. *Staadtsfeind* (public enemy, lit. enemy of the state).

When analyzing German compound words, some German morphological analyzers just generate a list of all possibilities, leaving to the calling application the task of choosing the right one [11]. In order to generate all the possible decompositions, the following strategies can be followed:

– Using a German lexicon (possibly generated automatically as all word forms from a corpus), a word can be divided into every possible sequence of substrings that belong to the lexicon, allowing for linking morphemes in between [8]. For instance, let us consider the word *Tagesration* (recommended daily amount). If we assume that the lexicon contains the words *Tag* (day), *ration* (ration), *rat* (advice, suggestion) and *ion* (ion), and if *es* is a linking morpheme that can be added to *Tag*, then all the following decompositions would be possible:

  Tagesration
  Tag(es)+ration
  Tag(es)+rat+ion
– Using a lexicon of decompounded words, it is also possible to store separately which words have been observed functioning as a modifier or as a head inside a compound [2,10]. Then, a word can be divided into every possible sequence of known modifiers followed by a known head. For example, in the previous example, if *rat* has never been observed as a modifier inside a compound, the third candidate would not be produced. This approach is probably more precise, but it requires to have a large list of decompounded words beforehand.

In order to choose the best splitting, some earlier approaches to the problem used naive heuristics, e.g. always choosing the splitting with the least number of parts [12]. Monz and de Rijke [5] describe a recursive procedure that decompounds words from left to right, choosing at each step greedily the smallest substring of the word that belongs to a lexicon. This algorithm cannot handle properly words with several possible splittings. For example, when analyzing *Ration*, the smallest prefix from the word that appears in the lexicon is *rat*, and the proposed decomposition would always be *rat+ion*, regardless of the context. The authors report a precision and recall of compound parts of around 49% for compound words.

Larson et al. [3] start by calculating, for each possible prefix and suffix of a word w, how many words share those prefixes and suffixes, and the distribution of differences of those values between two adjacent positions. Next, there is a splitting juncture for w such that it is located at local maxima of the distributions for both prefixes and suffixes. Intuitively, if a word w can be split into w_1 and w_2, and there are many words that either start with w_1 or end in w_2, that is possibly a good decompounding. The authors do an extrinsic evaluation on the effect of applying this decompounder to speech recognition, but no evaluation of the decompounder itself is provided. A similar approach for general morphological analysis of German, based on frequencies of prefixes and suffixes, is described by [13].

Schiller [2] uses Finite-State models and estimate, from a corpus, the probability that each compound part is used as a modifier or as a compound. Next, all the decompounding candidates are weighted as the product of the probability of each of the candidate substrings. The results are a precision of 89-98%, and a recall of 98–99%. This approach is followed closely by Marek [10].

A different approach consists of exploiting English and German sentence-aligned corpora to decompound words for Machine Translation [7,8]. In both works, a translation lexicon is induced from the parallel corpora. The difference between these two approaches is that [7] uses information about English and German cognates and the maximum common substring between German and English words to find where to split the German compounds, whereas [8] uses co-occurrence information between the German split candidates and their English translations throughout the corpus. The first approach seems more restricted, as it forces the compound parts to have at least 6 characters, and requires the table of cognate letter correspondences. [7] reports an estimated error rate between 1% and 7.6%, but recall is not calculated, and [8] reports a precision of 93% and a recall of 90%.

3 Problem Definition and Evaluation Settings

When decompounding query keywords there is a large amount of noisy data. In order to be robust enough to handle this kind of corpora, we have chosen the following requisites for a German decompounder:

– It should be able to split correctly German compounds, and leave German non-compounds untouched.
– It should not decompound unknown words that cannot be interpreted as compounds. This includes non-German words and German non-compounds that are misspelled.
– It should decompound unknown compound words if there is a plausible interpretation of them, because the decomposition can provide useful information. For instance, the following words are interpreted easily *Zürichstadt* (Zurich city), *Turingmaschine* (Turing machine) or *Blumenstrause*, even though *Zürich* and *Turing* may not appear in dictionaries because they are proper nouns, and *strause* is misspelled. In these three cases the decompositions will allow us to improve on the analysis of the writer's intent. This is specially important if just one of the compound parts is misspelled: a decomposition in two parts will allow us to have at least one of them as a correct word.
– It should decompound words that are not really grammatical compounds, but instead they are due to the user forgetting to input the blankspace between the words, such as *Bismarckausseenpolitik* (which should be written as *Bismark Aussenpolitik*), or *desktopcomputer* (two English words with no space in between).

For evaluating the different approaches, we have built and manually annotated a training set from the fully anonymized search query logs. Because people do not use capitalization consistently when writing queries, all the query logs are lowercased. Just by sampling randomly search keywords we would get very few compounds (as their frequency is small compared to that of non-compounds), so we have proceeded in the following way to ensure that the gold-standard contains a substantial amount of compounds: we started by building a very naive decompounder that splits a word in two parts if they have been seen in the query logs with a certain frequency (allowing for linking morpheme between the words). Using this procedure, we obtain a random sample of 1,250 words that are considered non-compounds by the naive approach, and 1,250 words that are considered compounds. Next, we removed all the duplicates from the previous list, and annotated them manually as compounds or non-compounds, including the correct splittings. Each compound was annotated by two human judges. The percentage of agreement in classifying words as compounds or non-compounds is 92.79% (Kappa=0.86), which shows there was a high level of agreement in this task. The total agreement (the percentage of words in which the same decomposition is provided) is 88.26%, and the most common source of disagreement were long words that could be split into two or more parts. The final list contains 2,267 word forms, 814 of which are real compounds.

The different procedures are evaluated in terms of precision, recall and accuracy, defined in the following way:

– Correct splits: no. of words that should be split and are split correctly.
– Correct non-splits: no. of words that should not be split and are not.
– Wrong non-splits: no. of words that should be split and are not.
– Wrong faulty splits: no. of words that should be split and are, but incorrectly.
– Wrong splits: no. of words that should no be split but are.

$$Precision = \frac{\text{correct splits}}{\text{correct splits} + \text{wrong faulty splits} + \text{wrong splits}}$$

$$Recall = \frac{\text{correct splits}}{\text{correct splits} + \text{wrong faulty splits} + \text{wrong no splits}}$$

$$Accuracy = \frac{\text{correct splits}}{\text{correct splits} + \text{wrong splits}}$$

4 Approaches

Most approaches for German compound splitting can be considered as having this general structure: given a word w,

1. Calculate every possible way of splitting w in one or more parts.
2. Score those parts according to some weighting function.
3. Take the highest-scoring decomposition. If it contains just one part, it means that w is not a compound.

The first step is usually implemented as finding every possible way of splitting w in several parts, each of which is a known word. The list of known words can be obtained from a dictionary, but, for the sake of coverage, obtaining the list of known words from a corpus is more common. Also, it must be taken into account that modifiers sometimes need a linking morpheme (a *fugenmorphem*) to be part

Table 2. Linking morphemes allowed between compound parts in this work

Morpheme	**German Example**
∅	Halbblutprinz (halb Blut Prinz, half-blood prince)
-e	Wundfieber (Wunde Fieber, traumatic fever)
+s	Rechtsstellung (Recht Stellung, legal status)
+e	Mauseloch (Maus Loch, mousehole)
+en	Armaturenbrett (Armatur Brett, instrument panel, dashboard)
+nen	
+ens	Herzensangst (Herz Angst, heart fear)
+es	Tagesanbruch (Tag Anbruch, daybreak, dawn)
+ns	Willensbildung (Wille Buildung, decision-making)
+er	Bilderrahmen (Bild Rahmen, picture frame)

of the compound. For example, given the word *orangensaft* (orange juice), if we know that *orange* and *saft* are both German words, and that *n* is a typical linking morpheme, we can propose *orange(n)+saft* as a possible decomposition of that word. We believe that the broadness of the list of linking morphemes considered will not affect very much the results if the words are collected from a large enough corpus. This is due to the fact that many of them are paradigmatic morphemes, so the original word together with its linking morphemes should also appear stand-alone in the corpus. In the previous example, *orangen* is the plural of *orange* and is a perfectly valid word.

Table 2 lists the linking morphemes we have chosen for our system. They are a selection of the ones listed by Langer [14] and Marek [10] We have not taken into consideration vowel changes of modifier words (umlauts) because we assume that those word variations can be found in the corpora, as discussed above.

A common variation in step 1 is to allow only binary splits, to simplify the problem. In this case, if the word w is split in two, the procedure can be called recursively on each of the two parts in order to see if they can be further split. This has the advantage that, together with the decomposition, we also obtain the structural analysis of the compound.

Obviously, a very important decision over this algorithm is the choice of the weighting function in step 2. The following subsections describe the weighting schemes tested in this work.

4.1 Frequency-Based Methods

The back-off method described by Koehn and Knight [8] makes the assumption that the more frequent a word is, the more likely it is to appear as a part of a compound. Therefore, given a word w, we choose the split S such that the geometric mean of the frequencies of its parts p_i is highest:

$$argmax_S \left(\prod_{p_i \epsilon S} count(p_i) \right)^{\frac{1}{n}}$$

4.2 Probability-Based Methods

The method applied by Schiller [2] and Marek [10] defines the weight of a word w as

$$W(w) = -\log\left(\frac{count(w)}{N}\right)$$

where $count_i$ is the frequency of w in a corpus, and N is the corpus size. In other words, each word is weighted as the minus logarithm of the maximum likelihood estimate of its probability of appearance in a corpus. Using these weights, the total weight of a decomposition can be calculated as the sum of weights of its parts, and the decomposition with the smallest weight will be chosen:

$$argmin_S \sum_{p_i \epsilon S} W(p_i)$$

It is possible to extend this procedure to have different models to represent the weight of words acting as heads or as modifiers, and to have also weights for the linking morphemes, but in order to do this it is necessary to have a large corpus already decompounded by hand so we can learn the probabilities of those. An advantage of this approach is that it can be easily implemented as a highly efficient finite-state automata for decompounding words in real-time. On the other hand, this model highly favors the decompositions with the smallest number of parts. This is because, with large corpus sizes (of the order of 10^9 words) most probabilities are very small, and the addition of a new compound part greatly increases the total weight of the decomposition.

4.3 Mutual Information

The Mutual Information (M.I.) of two words measures their mutual dependence. If two words can create a compound word in a language, we can assume that there is some kind of semantic relationships between them and therefore we would expect to be able to find them near each other in other situations. For example, *Blumensträuße* consists of the two words *flower* and *bouquet*, and we could say that there is a meronymy relationship between these two words (flowers are part of bouquets). Therefore, we might expect to find these two words near each other, as separate words, in other contexts.

For calculating the M.I., we calculate the frequency of each word in search query logs, and the frequency of co-occurrence of each pair of words in the same query.

4.4 Appearance in Anchor Texts Pointing to the Same Document

Anchor texts are the visible texts that are associated to a hyperlink to a web document. These are usually very informative and concise descriptions of their target documents. Therefore, if two hyperlinks are pointing to the same document, it is rather safe to assume that the associated anchor texts contain similar information.

Following with the same idea as with the Mutual Information, if two words can be joined together to form a compound, probably there is some relation between them. If the compound parts also appear separately in other anchor text to the same document, we can assume that that is a good evidence that the two words are actually semantically related and they are the right constitutive parts of the compound.

To calculate a weight following this idea, we scan 900 million web documents and keep every word w that satisfies that

- w appears in an anchor text of a hyperlink to a document d.
- There exist two words w_1 and w_2 such that $w = w_1.w_2$ and they appear in the anchor text of a different hyperlink to d.

After counting the frequencies of the words and word parts that satisfy these requirements, we finally order them by log likelihood [15].

4.5 Ad-Hoc Filtering

As mentioned before, there are many terms that can appear in search queries but should not be decompounded. Some of these are words from different languages which were wrongly classified as belonging to the language of interest. Other example is that of names of locations or organizations: depending on the application, it may be useful to leave them unsplit:

- Many locations in German end with suffixes indicating which kind of location it is, e.g. *wald* (forest), *berg* (mountain) or *dorf* (village). So the word should only be split if the inner structure of the proper noun is useful for the particular application. We have opted to leave all locations unsplit, including also street names, which in German usually end in *strasse*.
- Some trademarks, e.g. *easyjet* or *SkyDome* are actually formed by joining existing words in a similar process as compounding, but given their new meaning as names of companies or facilities they usually should not be decompounded.

Therefore, it is possible to apply simple Named Entity taggers to identify locations and organizations in order not to decompose those. Given that search queries do not have context from which to guess which words are entities, and capitalization information is also not always present, we have opted for a simple approach to identify the Named Entities that takes into consideration only gazetteers, pre-compiled lists of entities, and common entity word endings. These are applied by enforcing that any term appearing in the list should not be decompounded.

4.6 Combining the Previous Features

The previous sections described four different ways of weighting the different decompounding candidates in order to find the best split for each compound word, and an ad-hoc filtering procedure. These schemes may be capturing different qualities of the words that, when combined to each other, might provide improved results. To this aim, we have trained a Support Vector Machine classifier for performing binary splits. For each word w in the gold-standard, we create a training instance for the unsplit word, and a training instance for each possible way of splitting it into two parts. Each instance has the following features:

– Whether it has been split or not.
– If it has just one part, its frequency in the logs. If it has two parts, the geometric mean of the frequencies of the parts. Other features indicate whether this split has the greatest value for this feature among all possible splits, and whether it is greater than the value for the non-split version.
– If it has just one part, its probability-based weight estimated from the logs. Otherwise, the probability-based weight for this split. As in the previous case, there are further features referring to the distribution of this weight among all the instances obtained from w.
– If it has just one part, its entropy as estimated from the logs. Otherwise, the mutual information of both parts.
– If it has two parts, the log-likelihood of this split as obtained from the study of the anchor texts.
– Additional boolean features indicating whether the word contains one of the suffixes which usually appear in location names.
– Additional boolean features indicating whether the word appears in any of the dictionaries, lists of trademark terms or location gazetteers.
– The class of the feature, which is positive for the instances representing the correct way of splitting (or leaving as a single word) each term in the gold-standard.

When decompounding a new word, the features for the whole word and for any possible binary splits are generated, and the SVM classifies all the possibilities. The one that is classified as correct with the highest confidence is the one chosen.

5 Evaluation

Table 3 shows the results for each of the ranking functions, with or without the ad-hoc gazetteers and lists of entities that should not be split. From the results, we can draw the following conclusions:

– The worst result is obtained with the method based on the geometric mean of frequencies, specially in terms of precision. This is not surprising, considering that it is one of the simplest methods, and it does not take into consideration how often the compound parts co-occur. An example of a mistake is*Bauhofer* (a person from Bauhof), which is split into *Bau* (construction) and *Hofer* (a person from Hof). This method greatly favors very short substrings that appear with high frequencies on the web, resulting in splittings such as *Compostella* in *com+post+ella*. This is specially problematic for misspelled words, non-German words and Named Entities.

Table 3. Results of the several configurations. Each possible way of weighting the splits has been tested with or without the ad-hoc filtering.

Method	Filtering	Precision	Recall	Accuracy
Never split	-	-	0.00%	64.09%
Geometric mean of frequencies	no	39.16%	54.06%	65.04%
	yes	39.77%	54.06%	65.58%
Compound probability	no	53.99%	**81.27%**	70.06%
	yes	60.41%	80.68%	76.23%
Mutual Information	no	77.16%	50.53%	79.87%
	yes	82.00%	48.29%	80.52%
Presence in anchor texts	no	53.48%	25.32%	66.88%
	yes	71.10%	25.21%	71.07%
Support-Vector Machine	no	83.19%	79.15%	**90.55%**
	yes	**83.56%**	79.48%	87.21%

– Concerning the method based on estimating the probability of the compound parts, it performs much better than the method based on frequencies. It is relevant to note the high recall obtained by this method, as 81% of the compounds in the gold standard were properly identified and split. Precision, on the other hand, ranges around 60%, so it is rather low still. As the previous method, this one tends to over-generate compounds and it is still susceptible to words that have a very high-frequency on the web (e.g. *com* or *org*), but precision is much better.
– The method based on Mutual Information shows a different picture: here, recall is much lower than in the previous method, but precision reaches 82%. This is mainly due to a data sparseness problem, given that, in order to be able to split a compound in two parts, we need the probability of co-occurrence of those substrings in the same query. For many infrequent compound words, the compound parts do not appear in that situation, and their Mutual Information is zero. On the other hand, if the two parts of the compound appear together often in queries that seems to be a good indication that their meanings are related and they are parts of the same splitting. The precision of this method is the highest of the four stand-alone ranking functions tested.
– Concerning the method based on anchor texts, here the sparse data problem is even bigger, as it is much more difficult to have German anchor texts pointing to the same documents from which to extract these data than collecting a large amount of German query keywords. Therefore, recall is the lowest of all methods. The best precision obtained is 71%, placing this as the second best of the stand-alone ranking functions.
– Finally, as expected, combining the previous four methods into a single supervised system produces the best splittings in terms of precision and accuracy. The results shown in the table correspond to a 10-fold cross validation on the gold standard, using a polynomial kernel of degree 5, configuration

Table 4. Results dropping one of the information sources

Features	Precision	Recall	Accuracy
All features	83.19%	**79.15%**	90.55%
No frequencies	81.44%	74.88%	88.37%
No probabilities	82.30%	73.82%	88.23%
No Mutual Information	**83.54%**	78.79%	90.45%
No anchor texts data	83.50%	**79.15%**	**90.63%**

with which we have got the best results. It is important to note that the accuracy of this method is higher than the inter-judge agreement, so we can say that it is performing as good as possible given how the task has been defined.

It is interesting to note that the quality of the data, which includes many misspellings, foreign words and neologisms, makes this problem much harder than with revised and edited texts. Working with the Europarl corpus, [8] attained 57.4% precision and 86.6% recall, and [10] reports 99.4% precision and 99.4% recall on journal texts using the probability-based method.

Finally, Table 4 shows the results if we remove one of the sets of features from the model. The results agree with what intuitively we could expect: the weighting functions with the highest recall (the ones based on frequencies and probabilities) produce the biggest drop in recall when not used. On the other hand, if we remove the features about the Mutual Information and the co-occurrence in anchor texts, the overall impact is much smaller, so we could remove one of these features without really affecting much the overall results.

6 Conclusions

This paper describes a new procedure for splitting German compounds taken from query keywords, which constitute a very noisy corpus. The contributions of this paper are several: firstly, we have show, using a gold-standard created by sampling the query logs, that traditional procedures based on word frequencies and probabilities do not work well on this noisy data. Secondly, we have studied the application to this problem of other metrics, such as the Mutual Information and the co-occurrence in anchor texts. Thirdly, we have shown that the application of a simple Named Entity recognizer based on lists, gazetteers and word suffixes can substantially improve the precision of the previous methods. And, finally, we have provided an ensemble method that combines all the ranking functions using a Support Vector Machine, which attains a 39% error reduction over the best system in terms of accuracy. Future work include the application of this procedure to other compounding languages, and exploring the addition of new features to the model in order to obtain better results.

References

1. Baroni, M., Matiasek, J., Trost, H.: Predicting the Components of German Nominal Compounds. In: Proceedings of ECAI (2002)
2. Schiller, A.: German compound analysis with wfsc. In: Proceedings of Finite State Methods and Natural Language Processing 2005, Helsinki (2005)
3. Larson, M., Willett, D., Köhler, J., Rigoll, G.: Compound splitting and lexical unit recombination for improved performance of a speech recognition system for German parliamentary speeches. In: Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP) (2000)
4. Braschler, M., Göhring, A., Schäuble, P.: Eurospider at CLEF 2002. In: Peters, C., Braschler, M., Gonzalo, J. (eds.) CLEF 2002. LNCS, vol. 2785, pp. 127–132. Springer, Heidelberg (2003)
5. Monz, C., de Rijke, M.: Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In: Peters, C., Braschler, M., Gonzalo, J., Kluck, M. (eds.) CLEF 2001. LNCS, vol. 2406, pp. 262–277. Springer, Heidelberg (2002)
6. Brown, R.: Adding Linguistic Knowledge to a Lexical Example-Based Translation System. In: Proceedings of the Eighth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 1999), pp. 22–32 (1999)
7. Brown, R.: Corpus-driven splitting of compound words. In: Proceedings of the Ninth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-2002) (2002)
8. Koehn, P., Knight, K.: Empirical methods for compound splitting. In: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics, vol. 1, pp. 187–193 (2003)
9. Adda-Decker, M., Adda, G., Lamel, L.: Investigating text normalization and pronunciation variants for German broadcast transcription. In: Proceedings of ICSLP, pp. 66–269 (2000)
10. Marek, T.: Analysis of german compounds using weighted finite state transducers. Technical report, BA Thesis, Universität Tbingen (2006)
11. Finkler, W., Neumann, G.: Morphix. A fast realization of a classification-based approach to morphology. In: 4. Osterreichische Artificial-Intelligence-Tagung, Wiener Workshop-Wissensbasierte Sprachverarbeitung (1998)
12. Rackow, U., Dagan, I., Schwall, U.: Automatic translation of noun compounds. In: Proceedings of COLING-1992 (1992)
13. Demberg, V.: A language-independent unsupervised model for morphological segmentation. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007), Prague, Czech Republic (2007)
14. Langer, S.: Zur Morphologie und Semantik von Nominalkomposita. In: Tagungsband der 4. Konferenz zur Verarbeitung naturlicher Sprache (KONVENS) (1998)
15. Dunning, T.: Accurate Methods for the Statistics of Surprise and Coincidence. Computational Linguistics 19(1), 61–74 (1994)

Clause Boundary Identification Using Conditional Random Fields

R. Vijay Sundar Ram and Sobha Lalitha Devi

AU-KBC Research Centre,
MIT Campus Anna University,
Chromepet, Chennai -44,
India
(Sobha,sundar)@au-kbc.org

Abstract. This paper discusses about the detection of clause boundaries using a hybrid approach. The Conditional Random fields (CRFs), which have linguistic rules as features, identifies the boundaries initially. The boundary marked is checked for false boundary marking using Error Pattern Analyser. The false boundary markings are re-analysed using linguistic rules. The experiments done with our approach shows encouraging results and are comparable with the other approaches

1 Introduction

The clause identification is one of the shallow parsing tasks, which is important in various NLP applications such as translation, parallel corpora alignment, information extraction, machine translation and text-to-speech. The clause identification task aims at identifying the start and end boundaries of the clauses in a sentence, where clauses are word sequences which contain a subject and a predicate. The subject can be explicit or implied. For the clause identification task we have come up with a hybrid approach, where conditional random fields (CRFs), a machine learning technique and rule-based technique are used. The CRFs module with linguistic rules as features identifies the clause boundaries initially. The erroneous clause boundary detections are identified using an error analyzer and those sentences are processed using the rule-based module.

The clause identification was a shared task in CoNLL 2001. The task of identifying the clause boundaries is non-trivial. More research has been done in this task. The initial approaches to this task were using rule-based technique, which was followed by machine learning and hybrid techniques.

Early experiments in the clause boundary detection are Eva Ejeuhed's basic clause identification system for improving AT&T text to speech system [7], Papergeorgiou's rule based clause boundary system as preprocessing tool for bilingual alignment parallel text [15]. Leffa's rule based system reduces clauses to noun, adjective or an adverb, which was used in English/Portuguese machine translation system [10].

Since the rule based system requires more human work, machine learning systems were used for this task using manually annotated texts.

Orasan came up with a hybrid system using memory based learning and post processed using a rule based system [13]. Six different machine learning systems came out of CONLL (2001) shared task Ada boost algorithm for boosting the performance of decisions graph by Patrick and Goyal [14]. Symbolic learning ALLis by Dejean [6], specialized HMM based by Molina and Pla [11], Hammerton used long short-tem memory, a recurrent neural network architecture[8], Tjong Kim Sang used memory based learning approach [18] and Carreras's approach of decomposing clause into combination of binary decisions using Ada boost learning algorithm [1]. A partial parsing of sentence done by Carreras makes a global inference on a local classifier and used a dynamic programming for choosing the best decomposition of sentence to clauses [3]. Molina (2002) built a specialized HMM system to solve different shallow parsing task. Carreras (2003) did a phrase recognition using perceptrons and an online learning algorithm [4]. A multilingual clause splitting experiment was done by Georgiana, where he used a machine learning technique and indicators of co-ordination and subordination with verb information [16].

In this paper we used conditional random fields for clause boundary detection. CRFs is an undirected graphical model, where the conditional probability of the output are maximized for a given input sequences. [9]. This technique is proved successful for most of the sequence labeling tasks, such as shallow parsing by Sha and Pereira [17], named entity recognition task by Mc Callum and Li [5]. Recently CRFs was used for clause splitting task by Vinh Van Ngugen, where they have also used linguistic information and a bottom-up dynamic algorithm for decoding to split a sentence into clauses [20].

The rest of the paper is organized as follows, section 2 describes on our approach towards the clause boundary detection task. The different experiments conducted and the results are discussed in section 3. The paper concludes with a discussion on future works.

2 Our Approach

We have come up with a hybrid approach for clause boundary detection. The hybrid system contains conditional random fields (CRFs) and a rule-based approach. The clause detection system contains three main components, first the CRFs, second is the Error Analyser and third the linguistic rules.

The input sentence which is enriched with part-of-speech and chunking information is given to the Conditional Random Fields module. The incorrect detection of clause boundaries is detected using an error analyser module. The sentences with incorrect clause boundaries are checked for error patterns and corrected using linguistic rules. The different components of the system are explained below.

2.1 Conditional Random Fields for Clause Boundary Detection

Conditional Random Fields (CRFs)

CRFs is undirected graphical model where the conditional probabilities of the output are maximized for a given input sequence [8]. CRFs make a first-order markov independence assumption and thus it is a conditionally-trained finite state machine

(FSMs) [4]. CRFs have all the advantages of Maximum Entropy Markov model (MEMMs) but solves label bias problem which is the disadvantage of MEMMs.

Now let $o = (o_1, \dots, o_T)$ be some observed input data sequence, such as a sequence of words in a text document, (the values on T input nodes of the graphical model). Let S be a set of FSM states, each of which is associated with a label, $l \in L$, (such as PERSON). Let $s = (s_1, \dots, s_T)$ be some sequence of states, (the values on T output nodes).

Linear-chain CRFs thus define the conditional probability of a state sequence given as follows

$$P_\Lambda(\mathrm{s}|\mathrm{o}) = \frac{1}{Z_\mathrm{o}} \exp\left(\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(s_{t-1}, s_t, \mathrm{o}, t)\right),$$

where Z_0 a normalization factor over all state sequences, $f_k(s_{t-1}, s_t, o, t)$ is an arbitrary feature function over its arguments, and λ_k (ranging from $-\infty$ to ∞) is a learned weight for each feature function. A feature function may, for example, be defined to have value 0 in most cases, and have value 1 if and only if s_{t-1} is state #1 (which may have label OTHER), and s_t is state #2 (which may have START or END label), and the observation at position t in o is a relative pronoun or a conditional marker. Higher λ weights make their corresponding FSM transitions more likely, so the weight λ_k in the above example should be positive since the word appearing is any clause marker (such as conditional or relative clause marker) and it is likely to be the starting of a clause boundary.

Inference in CRFs

As in forward-backward for hidden Markov models (HMMs), inference can be performed efficiently by dynamic programming. We define slightly modified "forward values", $\sum_s \alpha_T(s_i)$, to be the probability of arriving in state si given the observations $(o_1, \dots, o_T)$. We set $\alpha_0(s)$ equal to the probability of starting in each state s, and recurse:

$$\alpha_{t+1}(s) = \sum_{s'} \alpha_t(s') \exp\left(\sum_{k=1}^{K} \lambda_k f_k(s', s, \mathrm{o}, t)\right).$$

The backward procedure and the remaining details of Baum-Welch are defined similarly. Z_0 is then $\sum_s \alpha_T(s_i)$. The Viterbi algorithm for finding the most likely state sequence given the observation sequence can be correspondingly modified from its HMM form.

Training of CRFs

The weights of a CRF, $\Lambda = \{\lambda, ...\}$, are set to maximize the conditional log-likelihood of labeled sequences in some training set,

$$\mathcal{D} = \{\langle o, l\rangle^{(1)}, ...\langle o, l\rangle^{(j)}, ...\langle o, l\rangle^{(N)}\},$$

$$L_\Lambda = \sum_{j=1}^{N} \log \left(P_\Lambda(\mathbf{l}^{(j)} | \mathbf{o}^{(j)}) \right) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2},$$

where the second sum is a Gaussian prior over parameters (with variance σ^2) that provides smoothing to help cope with sparsity in the training data. When the training labels make the state sequence unambiguous (as they often do in practice), the likelihood function in exponential models such as CRFs is convex, so there are no local maxima, and thus finding the global optimum is guaranteed.

Parameter estimation in CRFs requires an iterative procedure, and some methods require less iteration than others. Iterative scaling is the traditional method of training these maximum-entropy models (Darroch et al., 1980; Della Pietra et al., 1997), however it has recently been shown that quasi-Newton methods, such as L-BFGS, are significantly more efficient (Sha & Pereira, 2003).

L-BFGS can simply be treated as a black-box optimization procedure, requiring only that one provide the value and first-derivative of the function to be optimized. Assuming that the training labels on instance j make its state path unambiguous, let $s^{(j)}$ denote that path, and then the first-derivative of the log-likelihood is

$$\begin{aligned}\frac{\delta L}{\delta \lambda_k} &= \left(\sum_{j=1}^{N} C_k(\mathbf{s}^{(j)}, \mathbf{o}^{(j)}) \right) - \\ &\left(\sum_{j=1}^{N} \sum_{\mathbf{s}} P_\Lambda(\mathbf{s}|\mathbf{o}^{(j)}) C_k(\mathbf{s}, \mathbf{o}^{(j)}) \right) - \frac{\lambda_k}{\sigma^2}\end{aligned}$$

where $C_k(s, o)$ is the "count" for feature k given s and o. The first two terms correspond to the difference between the empirical expected value of feature f_k. The last term is the derivative of the Gaussian prior.

Features

The feature set used in our CRFs module is as follows. We used a word level feature by defining a window of size five. We have also added linguistic rules as features. The input to the CRFs module has four columns, the first column contains the list of words, the second column contains the part-of-speech (POS) information of the list of the words, the third column contains the chunking information, and the fourth column has a boolean entry based on whether the current word with its context obey the linguistics rules or not. Figure 1 shows an example of the input to the CRFs module.

In the above example the boolen entry in the last column on line 4 is set to 1 because the current word and its context obey one of the linguistic rules, such as the noun phrase between a verb phrase and an infinitive verb phrase has high probability

1	He	PRP	B-NP	0
2	would	MD	B-VP	0
3	allow	VB	I-VP	0
4	the	DT	B-NP	1
5	bill	NN	I-NP	0
6	to	TO	B-VP	0
7	become	VB	I-VP	0
8	law	NN	B-NP	0
9	without	IN	B-PP	0
10	his	PRP$	B-NP	0
11	signature	NN	I-NP	0
12	.	.	O	0

Fig. 1. Sample input text

of being the starting of the infinitive clause. The chunk boundaries are more important as a feature since most of the start and end boundaries of the clause matches with that of the chunk boundaries.

The word level features are represented using the following template

1. present word, its POS information and chunk information
2. present word, its POS information, chunk information and next words POS information.
3. previous words and its POS information and the present word.
4. previous word, present word and its POS information.
5. previous word's chunk information and the present word's chunk information.

The linguistic rules are represented in the fourth column which is added to the template.

2.2 Error Analyzer

The error analyzer module is used for detecting the erroneous clause boundary markings done by CRFs module. For detecting the errors, the training data (gold standard data) itself is given as test data to CRFs system. On analyzing the results, it was observed that the CRFs system was not able to handle the long distance dependencies between the start of the clause boundary and the end of the clause boundary. Similarly CRFs does not maintain a balance between the number of clause start and end boundary markings. These errors in the clause boundary marking are considered as patterns and we term it as 'error patterns'. Here we are giving a sample pattern.

Here in the above example though the clause ending is already marked, CRFs is marking a clause ending at the end of the sentence. This is due to the training corpus having many sentences where the ending of the sentence is marked as clause ending.

The error patterns derived by processing the gold standard data are compared with the output of the CRFs module to detect the incorrect clause boundaries marked by the CRFs module. Those sentences which match with error patterns are filtered out for further processing. Here we also check for more than one verb chunk after a

1. The	DT	B-NP	START
2. boys	NNS	I-NP	o
3. were	VBD	B-VP	o
4. playing	VBG	I-VP	END
5. COMMA	,	O	o
6. she	PRP	B-NP	o
7. said	VBD	B-VP	o
8. to	TO	B-PP	o
9. Mary	NNP	B-NP	END
10..	.	O	END

Fig. 2. Sample Error Pattern

clause start boundary marking and within a clause (ie before getting a clause end boundary marker). These sentences are also filtered out for further processing using the rule-based module.

2.3 Linguistic Rules

The erroneous clause boundary marked sentences which are filtered out by the error analyzer module are further analysed using linguistic rules. This consists of a set of linguistic and heuristic rules. The heuristic rules are used to treat the erroneous clause boundary markings done in the CRFs module. The linguistic rules are used to identify the unidentified clause boundaries in the given sentences. By using the linguistic rules the long dependencies between the start and the end of the clause are handled.

From the error analyzer we find that major errors are occurring in complementizer, infinitive and relative clauses. Hence we crafted linguistic rules to handle them. There are 8 linguistic rules and some of the rules are explained below.

Rule 1: This is more for complementizer clause
-1, VP = 1
0 <NP>
1 ,
2 say (past/present)
3 <NP>
0 should be marked with ending of clause boundary.

If the current word is ending with a noun phrase chunk (NP) and there occurs a verb phrase (VP) prior to the NP and the current word is followed by the verb ‘say’ (past/present) along with an NP then the current word should be marked with clause end marker.

Rule 2: This is more for infinitive clause
-1 <VP>
0 <NP>
1 <VP infinitive>
0 should be marked with starting of clause boundary.

If the current word is a starting of an NP and the NP occurs between a VP and an infinitive VP, the current word is marked with starting of a clause marker.

Rule 3: This rule is used for conditional and relative clauses
-2 <start>
-1 <VP>
0 <NP>
0 should be marked with ending of clause boundary.

If the current word is the ending of the NP and prior to the NP there is a VP and the VP is preceded by the starting of a clause marker, then the current word is marked with clause ending marker.

The various rules are as follows

The implementation of the linguistic rules are explained in the following steps

```
1, Reads the list of the rules from the file.
2 Checks for the inner most starting of clause boundary
marking
3 Checks if the clause at that start position obeys the
rule
          Then
             The clause ending boundary is marked.
          Else
             Check with the next rule.
4 the above steps are done recursively to handle the
different clauses and the clauses in a sentence.
```

3 Experiments

We did our experiments and evaluated our results using the data, which was used for CoNLL 2001 shared task. The data is from WSJ corpus for both training and testing. The training data has 8936 sentences, development data has 2012 setences and the testing data has 1671 sentences. The CoNLL data was enriched with the part-of-speech information, chunking information and clause boundary information. The data is in column format, where the words with their punctuation marks are listed in the first column, the second column is the part-of-speech information of the listed words, and the third column is the chunking information. In the training data of CoNLL the fourth column had the information about the clause boundaries. But we inserted a new fourth column pushing the clause boundary information to the fifth column. Our fourth column had a Boolean entries based on the current word and its context obeying any of the set of linguistic rule. By adding this information we were able to include linguistic features in the CRFs module. In our system, we used CRF++ [19] to implement the CRFs module.

We measure our systems performance in terms of precision and recall, where precision is the number of correctly recognized clauses to the number of clauses marked in the output and recall is defined as the number of correctly recognized clauses to the number of clauses. The precision of the system is found to be 92.06% and the recall of the system is 87.89%. The performance of the system using the CRFs and linguistic rules and CRfs alone is tabulated in Table (1).

Table 1. Perfomance of the system

S.No	System	Precision (%)	Recall (%)	Fmeasure (%)
1	CRFs	83.68%	78.65%	81.08%
2	**CRFs with linguistic Rules**	**92.06%**	**87.89%**	**89.04%**

Significance of linguistic Rules with CRFs

The performance improved when linguistics rules were applied to the CRFs output. The clause boundaries which were ambiguous could not be handled by CRFs and these could be fixed using linguistics rules and this in turn helped in improving the precision and the recall of the system.

We have also evaluated the performance of the system in terms of the start and end of clause boundaries. We have tabulated the results from the development and the test data. The results on start and end of the clause boundaries are given on Table 2 and 3 respectively. The results on how each linguistic rule performs are given on Table 4.

Table 2. Start of Clause boundary

Category	Precision (%)	Recall (%)	F measure (%)
Development data	97.08	93.56	95.29
Test Data	95.89	92.28	94.05

Table 3. End of Clause boundary

Category	Precision (%)	Recall (%)	F measure (%)
Development Data	92.87	89.78	91.29
Test Data	91.68	88.84	90.24

The start of the clause boundaries are more fixed than the end and the machine learning system learns them with a high precision, but when there are grammatical changes in the sentence the machine learning systems fails and by adding the linguistic rules as a feature, we are able to handle the identification of the start of the clause boundaries with very high precision.

Since the CRFs module fails to handle the long distance dependencies between the start and end of the clause boundaries, we use linguistic rules to handle it. This has significantly improved the clause end identification as well. The rule looks for one verb chunk per clause and this rule improved the overall clause identification. The use of error-analyser module has filtered out the erroneous clause marked sentences The erroneous clause marked sentences which are filtered by the error analyzer had 647 wrong start tags and 1277 wrong end tags. These clauses are processed with the linguistic and heuristic rules. There are 8 rules and the performance of each rule is given below.

Table 4. Performance of the various linguisitc rules

Rules	Number of Start marked	Number of End marked
1	84	37
2	-	174
3	23	38
4	-	74
5	-	466
6	64	-
7	21	-
8	-	177
	192	966

Table 5. Comparison of different systems

S. No	References	Techniques	Precision	Recall	F1 mesaure
1	**Our method**	**CRFs + linguistic Rules**	**92.06%**	**87.89%**	**89.04%**
2	Carreras et al. 05	FR-Perceptron	88.17%	82.10%	85.03%
3	Vinh Van Nyugenet al 07	CRFs	**90.01%**	**78.98%**	**84.09%**
4	Carreras et al. 02	AdaBoost class	90.18%	78.11%	83.71%
5	Carreras et al. 01	AdaBoost class	84.82%	78.85%	81.73%
6	Monila and Pla 01	HMM	70.85%	70.51%	70.68%

The result of our system is very much comparable in performance with other existing systems. We have shown the comparison results in Table 5.

4 Conclusion

Thus in this paper we presented a hybrid system using conditional random fields and rule-based approach for the task of detecting the clause boundaries. We tried a different approach of adding linguistic rules as one of the features of the CRFs. We are planning to improve our system by incorporating more linguistic rules and to train the CRFs with phrases instead of words.

References

1. Carreras, X., Màrquez, L.: Boosting Trees for Clause Splitting. In: Daelemans., W., Zajac, R. (eds.) Proceedings of CoNLL 2001, Toulouse France, pp. 73–75 (2001)
2. Carreras, X., Màrquez, L.: Phrase Recognition by Filtering and Ranking with Percep-trons. In: Proceedings of RANLP-2003, Borovets Bulgaria, pp. 205–216 (2003)
3. Carreras, X., et al.: Learning and Inference for Clause Identification. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) ECML 2002. LNCS (LNAI), vol. 2430, pp. 35–47. Springer, Heidelberg (2002)
4. Carreras, X., Màrquez, L., Castro, J.: Filtering-ranking Perceptron Learning for Partial Parsing. Machine Learning 60(1), 41–71 (2005)
5. McCallum, A., Li, W.: Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web Enhanced Lexicons. In: Proceedings of CoNLL-2003, Edmonton Canada, pp. 188–191 (2003)
6. Déjean, H.: Using Allis for Clausing. In: Daelemans, W., Zajac, R. (eds.) Proceedings of CoNLL-2001, Toulouse France, pp. 64–66 (2001)
7. Ejerhed, E.: Finding Clauses in Unrestricted Text by Finitary and Stochastic Methods. In: Proceedings of the 2nd Conference on Applied Natural Language Processing, Austin Texas, pp. 219–227 (1988)
8. Hammerton, J.: Clause Identification with Long Short-term Memory. In: Daele-mans, W., Zajac, R. (eds.) Proceedings of CoNLL 2001, Toulouse France, pp. 61–63 (2001)
9. Lafferty., J., McCallum., A., Pereira, F.: Conditional Random Fields: Prob-abilistic Models for Segmenting and Labeling Sequence Data. In: Proc. 18th International Conference on Machine Learning, pp. 282–289. Morgan Kaufmann, San Francisco (2001)
10. Vilson, J.L.: Clause Processing in Complex Sentences. In: Proceedings of the First International Conference on Language Resource & Evaluation, vol. 1, pp. 937–943 (1998)
11. Molina., A., Pla, F.: Clause Detection Using HMM. In: Daelemans., W., Zajac., R. (eds.) Proceedings of CoNLL-2001, Toulouse, France, pp. 70–72 (2001)
12. Molina., A., Pla, F.: Shallow Parsing Using Specialized HMMs. Journal of Ma-chine Learning Research 2, 595–613 (2002)
13. Orasan, C.: A Hybrid Method for Clause Splitting in Unrestricted English Text. In: Proceedings of ACIDCA 2000 Corpora Processing, Monastir Tunisia, pp. 129–134 (2000)
14. Jon, D.P., Goyal, I.: Boosted Decision Graphs for NLP Learning Tasks. In: Daelemans., W., Zajac, R. (eds.) Proceedings of CoNLL-2001, Toulouse France, pp. 58–60 (2001)

15. Harris, V.P.: Clause Recognition in the Framework of Alignment. In: Mitkov, R., Nicolov, N. (eds.) Recent Advances in Natural Language Processing, pp. 417–425. John Benjamins Publishing Company, Amsterdam/Philadelphia (1997)
16. Puscasu, G.: A Multilingual Method for Clause Splitting. In: Proceedings of the 7th Annual Colloquium for the UK Special Interest Group for Computational Linguistics, Birmingham UK (2004)
17. Sha, F., Pereira, F.: Shallow Parsing with Conditional Random Fields. In: Proceedings of HLT-NAACL03, pp. 213–220 (2003)
18. Erik, F.T.K.S., Déjean, H.: Introduction to the CoNLL-2001 shared task: Clause Identification. In: Daelemans, W., Zajac, R. (eds.) Proceedings of CoNLL-2001, Toulouse France, pp. 53–57 (2001)
19. Kudo, T.: CRF++, an Open Source Toolkit for CRF (2005), `http://crfpp.sourceforge.net`
20. Van Nguyen, V.: Using Conditional Random Fields for Clause Splitting. In: Proceedings of The Pacific Association for Computational Linguistics, University of Melbourne Australia (2007)

Natural Language as the Basis for Meaning Representation and Inference

Ido Dagan[1], Roy Bar-Haim[1], Idan Szpektor[1], Iddo Greental[2], and Eyal Shnarch[1]

[1] Bar Ilan University, Ramat Gan 52900, Israel
`dagan@cs.biu.ac.il`
[2] Tel Aviv University, Tel Aviv 69978, Israel

Abstract. Semantic inference is an important component in many natural language understanding applications. Classical approaches to semantic inference rely on logical representations for meaning, which may be viewed as being "external" to the natural language itself. However, practical applications usually adopt shallower lexical or lexical-syntactic representations, which correspond closely to language structure. In many cases, such approaches lack a principled meaning representation and inference framework. We describe a generic semantic inference framework that operates directly on language-based structures, particularly syntactic trees. New trees are inferred by applying entailment rules, which provide a unified representation for varying types of inferences. Rules were generated by manual and automatic methods, covering generic linguistic structures as well as specific lexical-based inferences. Initial empirical evaluation in a Relation Extraction setting supports the validity and potential of our approach. Additionally, such inference is shown to improve the critical step of unsupervised learning of entailment rules, which in turn enhances the scope of the inference system.

This paper corresponds to the invited talk of the first author at CICLING 2008.

1 Introduction

It has been a common assumption that the structure of natural language is not suitable to formally represent meanings and to conduct inferences over them. Indeed, according to the traditional formal semantics approach inference is conducted at a logical level. Texts are first translated, or *interpreted*, into some logical form and then new propositions are inferred from interpreted texts by a logical theorem prover. Meaning and inference are thus captured by representations that are "external" to the language itself, and are typically independent of the structure of any particular natural language.

However, practical text understanding systems usually employ shallower lexical and lexical-syntactic representations, which clearly correspond to the structure of the particular natural language being processed. Such representations are sometimes augmented with partial semantic annotations like word senses, named-entity classes and semantic roles. This state of affairs was clearly demonstrated in the recent PASCAL Recognizing Textual Entailment (RTE) Challenges [1, 2, 3], a popular framework for evaluating application-independent semantic inference, where only a few systems applied logical inference [4, 5, 6].

While practical semantic inference is mostly performed over linguistic rather than logical representations, such practices are typically partial and quite ad-hoc, and lack a clear formalism that specifies how inference knowledge should be represented and applied. This paper describes a step towards filling this gap, by defining a principled semantic inference mechanism over natural language based representations, particularly parse-based structures (originally presented in [7]). In fact, the formulation of the textual entailment task as a touchstone for semantic processing, which is not bound to any particular extra-linguistic representation, encourages the development of semantic formalisms like ours which operate directly over relatively immediate natural language structures.

Within the textual entailment setting a semantic inference system is required to recognize whether a hypothesized statement h can be inferred from an asserted text t. Overall, the task consists of two different types of inference. Some inferences can be based on available knowledge, such as information about synonyms, paraphrases, world knowledge relationships etc. In the general case, however, some knowledge gaps arise and it is not possible to derive a complete "proof" based on available inference knowledge. Such situations are typically handled through approximate matching methods.

This paper focuses on the first type of knowledge-based inference (see [8] for an initial integration of approximate matching methods within our framework, applied to the RTE-3 benchmark). We define a proof system that operates over syntactic parse trees, which are the basis for representing both sentence meaning and inference knowledge. New trees are derived using *entailment rules*, which provide a principled and uniform mechanism for incorporating a wide variety of inference knowledge types. Notably, this approach allows easy incorporation of rules learned by unsupervised methods, which seems essential for scaling inference systems. Interpretation into stipulated semantic representations, which is often difficult and is inherently a supervised semantic task for learning, is circumvented altogether.

Our overall research goal is to explore how far we can get with such an inference approach, and identify the practical scope within which semantic interpretation is not needed. This goal is somewhat analogous to that of the Natural Logic paradigm [9, 10], which attempted to model monotonicity-based inference using natural language rather than logical representation. While similar modeling of such phenomena may be embedded in our framework as well, we are mostly interested in modeling a broader range of phenomena which are most relevant for applied semantic inference.

The first sections of this paper present our formulation and implementation of the inference framework, and its evaluation on an inference task. In addition, we show (in Section 6) how the inference mechanism can be utilized also to improve unsupervised learning of entailment rules, which in turn enhances the potential scope of the inference system itself (originally presented in [11]; see this paper and [7] for additional detail about our and related works).

2 Inference Framework

Given two syntactically parsed text fragments, termed *text* (t) and *hypothesis* (h), the goal of the inference system (or *prover*) is to determine whether t entails h. The prover tries to generate h from t by applying *entailment rules* that aim to transform t into h, through a sequence of intermediate parse trees. If such a proof is found, the prover concludes that entailment holds.

Like logic-based systems, our inference framework is composed of *propositions* and *inference rules*. The propositions include t (the assumption), h (the goal), and intermediate premises inferred during the proof. The inference (entailment) rules define how new propositions are derived from previously established ones.

2.1 Propositions

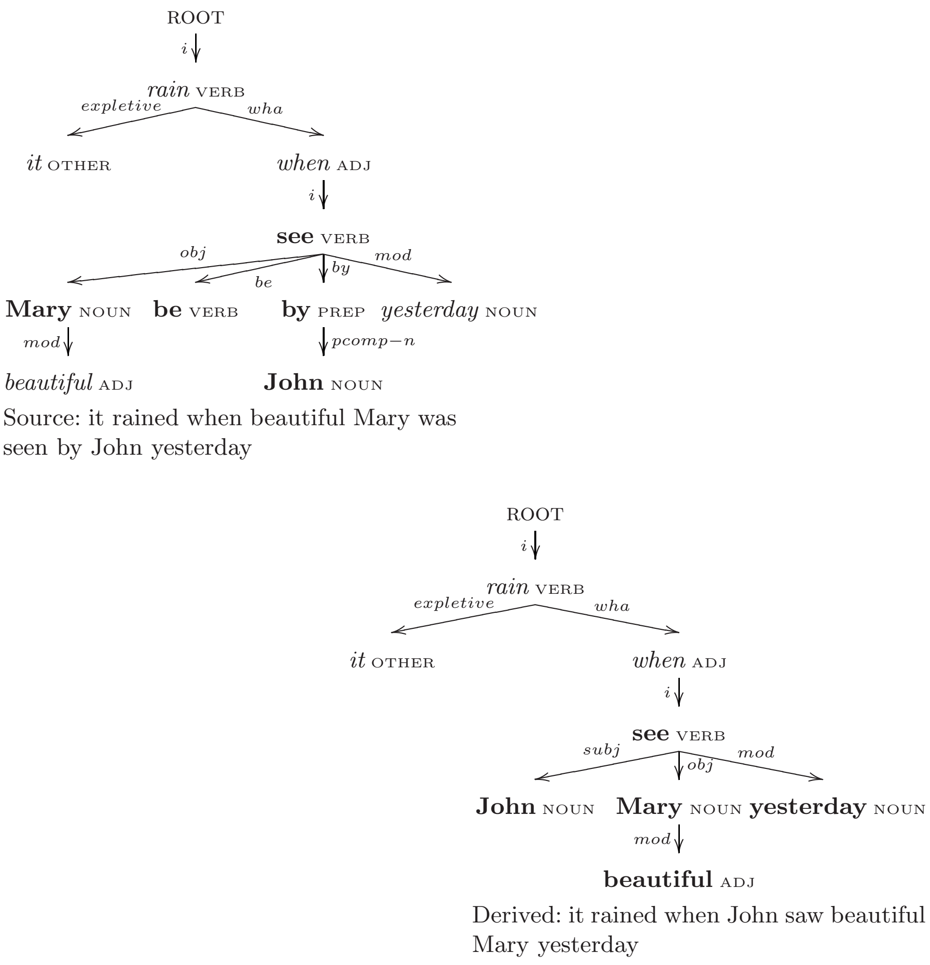
The general inference framework assumes that propositions are represented by some form of parse trees. In this paper we focus on dependency tree representation, which is often preferred to capture directly predicate-argument relations (Figure 1(a)). Nodes represent words and hold a set of features and their values. These features include the word lemma and part-of-speech, and additional features that may be added during the proof process. Edges are annotated with dependency relations.

2.2 Entailment Rules

At each step of the proof an entailment rule generates a *derived* tree d from a *source* tree s. A rule '$L \rightarrow R$' is primarily composed of two templates, termed *left-hand-side* (L), and *right-hand-side* (R). *Templates* are dependency subtrees which may contain *variables*. Figure 1(b) shows an entailment rule, where V, $N1$ and $N2$ are common variables shared by L and R. L specifies the subtree of s to be modified, and R specifies the new generated subtree. Rule application consists of the following steps:

L matching. The prover first tries to match L in s. L is *matched* in s if there exists a one-to-one node mapping function f from L to s, such that: (i) For each node u in L, $f(u)$ has the same features and feature values as u. Variables match any lemma value in $f(u)$. (ii) For each edge $u \rightarrow v$ in L, there is an edge $f(u) \rightarrow f(v)$ in s, with the same dependency relation. If matching fails, the rule is not applicable to s. Otherwise, successful matching induces *variable binding* $b(X)$, for each variable X in L, defined as the full subtree rooted in $f(X)$ if X is a leaf, and $f(X)$ alone otherwise. We denote by l the subtree in s to which L was mapped (as illustrated in bold in the upper-left part of Figure 1(a)).

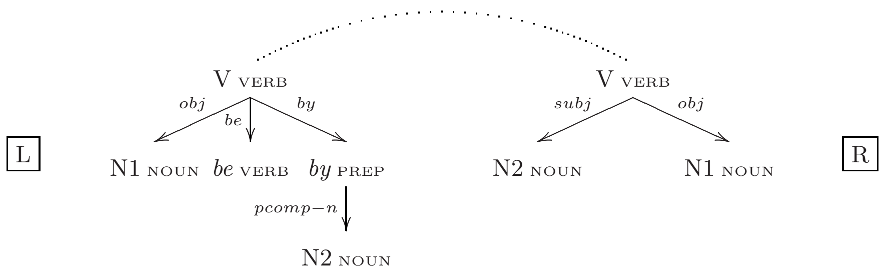
R instantiation. An instantiation of R, which we denote r, is generated in two steps: (i) creating a copy of R; (ii) replacing each variable X with a copy of



Source: it rained when beautiful Mary was seen by John yesterday

Derived: it rained when John saw beautiful Mary yesterday

(a) Passive-to-active tree transformation



(b) Passive to active substitution rule. The dotted arc represents alignment.

Fig. 1. Application of an inference rule. POS and relation labels are based on Minipar [12]

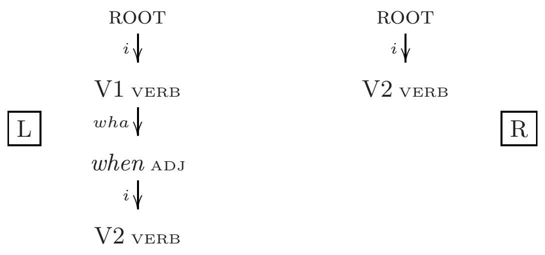


Fig. 2. Temporal clausal modifier extraction (introduction rule)

its binding $b(X)$ (as set during L matching). In our example this results in the subtree *John saw beautiful Mary*.

Alignment copying. Part of the rule definition is an *alignment* relation between pairs of nodes in L and R that specifies which modifiers in l that are not part of the rule structure need to be copied to the generated r. Formally, for any two nodes u in l and v in r whose matching nodes in L and R are aligned, we copy the daughter subtrees of u in s, which are not already part of l, to become daughter subtrees of v in r. The bold nodes in the lower-right part of Figure 1(a) correspond to r after alignment copying. *yesterday* was copied to r due to the alignment of its parent verb node.

Derived tree generation by rule type. Our formalism has two methods for generating the derived tree: *substitution* and *introduction*, as specified by the rule type. With *substitution* rules, the derived tree d is obtained by making a local modification to the source tree s. Except for this modification s and d are identical (a typical example is a lexical rule, such as *buy* → *purchase*). For this type, d is formed by copying s while replacing l (and the descendants of l's nodes) with r. This is the case for the passive rule. The lower-right part of Figure 1(a) shows the derived tree for the passive rule application. By contrast, *introduction* rules are used to make inferences from a subtree of s, while the other parts of s are ignored and do not effect d. A typical example is inference of a proposition embedded as a relative clause in s. In this case the derived tree d is simply taken to be r. Figure 2 presents such a rule which enables to derive propositions that are embedded within temporal modifiers. Note that the derived tree does not depend on the main clause. Applying this rule to the lower-right part of Figure 1(a) yields the proposition *John saw beautiful Mary yesterday*.

2.3 Annotation Rules

Annotation rules add features to parse tree nodes, and are used in our system to annotate negation and modality. Annotation rules do not have an R, but rather each node of L may contain annotation features. If L is matched in a tree then the annotations are copied to the matched nodes. Annotation rules are applied to the original text t, and to each inferred premise, prior to any entailment rule application. Since the annotated features would be checked during subsequent L

matching of rules, these additional features may block inappropriate subsequent rule applications, such as for negated predicates.

2.4 Template Hypotheses

For many applications it is useful to allow the hypothesis h to be a template rather than a proposition, that is, to contain variables. The variables in this case are existentially quantified: t entails h if there exists a proposition h', obtained from h by variable instantiation, so that t entails h'. The obtained variable instantiations may stand, for example, for sought answers in questions or slots to be filled in relation extraction applications. For example, applying this framework in a question-answering setting, the question *Who killed Kennedy?* may be translated into the template hypothesis *X killed Kennedy*. A successful proof of h from the sentence *"The assassination of Kennedy by Oswald shook the nation"* would instantiate X with *Oswald*.

3 Rules for Generic Linguistic Structures

Based on the above framework for entailment rules we have manually created a rule base for generic linguistic phenomena. The current rule base was developed under the assumption that the hypothesis h has a relatively simple structure and is positive (non-negated) and non-modal, which is often the case in applications such as question answering and information extraction. Accordingly, the rules aim to simplify and decompose the source proposition, and to block inference from negated and modal predicates. The various rule categories we developed are summarized in Table 1 and explained below.

3.1 Syntactic-Based Rules

These rules capture entailment inferences associated with common syntactic structures. The rules have three major functions: (1) simplification and canonization of the source tree (categories 6 and 7 in Table 1); (2) extracting embedded propositions (categories 1, 2, 3); (3) inferring propositions from non-propositional subtrees of the source tree (category 4).

3.2 Polarity-Based Rules

Consider the following two examples:

John *knows* that Mary is here $\Rightarrow$ Mary is here.
John *believes* that Mary is here $\nRightarrow$ Mary is here.

Valid inference of propositions embedded as verb complements depends on the verb properties, and the polarity of the context in which the verb appears (positive, negative, or unknown) [13]. We extracted from the polarity lexicon of Nairn

Table 1. Summary of rule base for generic linguistic structures, and examples of their application

#	Category	Example: source	Example: derived
1	Conjunctions	Helena's very experienced and has played a long time on the tour.	⇒ Helena has played a long time on the tour.
2	Clausal modifiers	But celebrations were muted as many Iranians observed a Shi'ite mourning month.	⇒ Many Iranians observed a Shi'ite mourning month.
3	Relative clauses	The assailants fired six bullets at the car, which carried Vladimir Skobtsov.	⇒ The car carried Vladimir Skobtsov.
4	Appositives	Frank Robinson, a one-time manager of the Indians, has the distinction for the NL.	⇒ Frank Robinson is a one-time manager of the Indians.
5	Determiners	The plaintiffs filed **their** lawsuit last year in U.S. District Court in Miami.	⇒ The plaintiffs filed **a** lawsuit last year in U.S. District Court in Miami.
6	Passive	We have been approached by the investment banker.	⇒ The investment banker approached us.
7	Genitive modifier	Malaysia's crude palm oil output is estimated to have risen by up to six percent.	⇒ The crude palm oil output of Malasia is estimated to have risen by up to six percent.
8	Polarity	Yadav was forced to resign.	⇒ Yadav resigned.
9	Negation, modality	What we've **never** seen is actual costs come down.	What we've never $\overline{\text{seen}}$ is actual costs come down. (⇏ What we've seen is actual costs come down.)

et al. (see Acknowledgments) a list of verbs for which inference is allowed in positive polarity context, and generated entailment rules for these verbs (category 8 in Table 1). The list was complemented with a few reporting verbs, such as *say* and *announce*, since information in the news domain is often given in reported speech, while the speaker is usually considered reliable.

3.3 Negation and Modality Annotation Rules

We use annotation rules to mark negation and modality of predicates (mainly verbs), based on their descendent modifiers. Since annotation rules may capture subtrees of any size, we can use them to identify negation and modality phenomena in complex subtrees where the source of the phenomenon is not in the immediate daughter node of the predicate. Negation rules identify full and contracted verbal negation, as well as negation implied by certain determiners and nouns. Modality rules identify modality expressed by the use of modal verbs such as *should*, as well as conditional sentences and modal adverbials. Category

9 in Table 1 illustrates a negation rule, annotating the verb *seen* for negation due to the presence of *never*.

3.4 Generic Default Rules

Generic default rules are used to define default behavior, in situations where no case-by-case rules are available. We used one default rule that allows removal of any modifiers from nodes. Desirably, specific rules should be specified in future work to capture more precisely many cases that are currently handled by this default rule.

4 Lexical-Syntactic Rules

Lexical-Syntactic entailment rules include open-class lexical components within varying syntactic structures. Accordingly these rules are numerous compared to the generic rules of the previous section, and have been acquired either from available large-scale lexicographic resources or automatically (e.g. paraphrases). We incorporated several sources of such rules, as described below (see [8] for our use of WordNet as an additional resource for lexical entailment rules).

4.1 Nominalization Rules

Entailment rules such as 'X's acquisition of $Y \rightarrow X$ acquired Y' capture the relations between verbs and their nominalizations. These rules were derived automatically [14] from Nomlex, a hand-coded database of English nominalizations [15], and from WordNet.

4.2 Automatically Learned Rules

DIRT [16] and TEASE [17] are two state-of-the-art unsupervised algorithms that learn lexical-syntactic inference rules.[1] Some of the learned rules are linguistic paraphrases, e.g. 'X confirm $Y \rightarrow X$ approve Y', while others capture world knowledge, e.g. 'X file lawsuit against $Y \rightarrow X$ accuse Y'. These algorithms do not learn the entailment direction, which reduces their accuracy when applied in any given direction. For each system, we considered the top 15 bi-directional rules learned for each template.

5 Evaluation

As the current work is concerned with performing exact proofs, we should evaluate its precision over text-hypothesis pairs for which a complete proof chain is found, using the available rules. We note that the PASCAL RTE datasets are

[1] Their output is publicly available at the ACLWiki Textual Entailment Resources Pool.

not suitable for this purpose. These rather small datasets include many pairs for which entailment recognition requires approximate matching, as currently it is not realistic to assume sufficient knowledge that will enable a complete exact proof. As an alternative we chose a Relation Extraction (RE) setting, for which complete proofs can be achieved for a large number of corpus sentences. In this RE setting, the system needs to identify in sentences pairs of arguments for a target semantic relation (e.g. *X buy Y*).

5.1 Evaluation Process

We use a sample of test template hypotheses that correspond to typical RE relations, such as *X approve Y*. We then identify in a large test corpus sentences from which an instantiation of the test hypothesis is proved. For example, the sentence *The law was confirmed by the parliament* is found to prove the instantiated hypothesis *parliament approve law*. Finally, a sample of such sentence-hypothesis pairs are judged manually for true entailment. The process was repeated to compare different system configurations.

We tested hypotheses that are covered by all our lexical-syntactic resources. Since the publicly available output of TEASE is much smaller than the other resources, we selected from this resource 9 transitive verbs that may correspond to typical RE predicates,[2] forming test templates by adding subject and object variable nodes.

For each test template h we need to identify in the corpus sentences from which it is proved. To find efficiently proof chains that generate h from corpus sentences we combined forward and backward (Breadth-First) search over the available rules. First, backward search is used over the lexical-syntactic rules, starting with rules whose right-hand-side is identical to the test template hypothesis. While backward chaining the DIRT/TEASE and nominalization rules, this process generates a set of intermediate templates t_i, all of them proving (deriving) h. For example, for the hypothesis *X approve Y* we may generate the template *X confirm Y*, through backward application of the DIRT/TEASE rule 'X confirm $Y \rightarrow X$ approve Y', and then further generate the template *confirmation of Y by X*, through the corresponding nominalization rule. Since the templates t_i are generated by lexical-syntactic rules, which modify open-class lexical items, they may be considered as "lexical expansions" of h.

Next, for each specific t_i we generate a search engine query composed of the open-class words in t_i. This query fetches from the corpus candidate sentences, from which t_i might be proven using the generic linguistic rules (recall that the generic rules do not modify open-class words). To that end we apply a forward search that applies the generic rules, starting from a candidate sentence s and trying to derive (prove) t_i by a sequence of rule applications. If successful, this process instantiates the variables in t_i with the appropriate variable bindings to elements in s. Consequently, we know that, under the same variable instantiations, h can be proved from s (since s derives t_i which in turn derives h).

[2] The verbs are *approach, approve, consult, lead, observe, play, seek, sign, strike.*

Table 2. Empirical evaluation - results

#	Configuration	Precision	Yield
1	Baseline	67.0%	2,414
2	Proof	78.5%	1,426
3	+Gen	74.8%	2,967
4	+Gen+LexSyn	23.6%	18,809

The above search for sentences that prove each test template was performed over the Reuters RCV1 corpus, CD#2, applying Minipar [12] for parsing. Through random sampling we obtained 30 sentences that prove each of the 9 test templates, yielding a total of 270 pairs of a sentence and an instantiated hypothesis for each of the four tested configurations (1080 pairs overall). These pairs were split for entailment judgment between two human annotators. The annotators achieved, on a sample of 100 shared examples, agreement of 87%, and a Kappa value of 0.71 (corresponding to "substantial agreement").

5.2 Results

We tested 4 configurations of the proof system:

1. Baseline The baseline configuration follows the prominent approach in graph-based entailment systems (see next section): the system simply tries to embed the given hypothesis anywhere in the text tree, while only modality or negation (detected by the annotation rules) may block embedding.
2. Proof: The basic configuration of our prover, where h has to be strictly generated from t rather than embedded in t. The only inference rule available in the basic Proof configuration is the default rule for removing modifiers (annotation rules are active as in Baseline).
3. +Gen: As Proof, plus generic linguistic rules.
4. +Gen+LexSyn: As +Gen, plus lexical-syntactic rules.

For each system configuration we measure *precision*, the percentage of examples judged as correct (entailing), and average *extrapolated yield*, which is the expected number of truly entailing sentences in the corpus that would be proved as entailing by the system.[3] We note that, similar to IR evaluations, it is not possible to compute true recall in our setting since the total number of entailing sentences in the corpus is not known (recall would be equal to the yield divided by this total). However, it is straightforward to measure *relative* recall differences among different configurations based on the yield. Thus, using these two measures estimated from a large corpus it is possible to conduct robust comparison between different configurations, and reliably estimate the impact of different rule types. Such analysis is not possible with the RTE datasets, which are rather

[3] The extrapolated yield for a specific template is calculated as the number of sample sentences judged as entailing, multiplied by the sampling proportion. The average is calculated over all test templates.

Table 3. Examples of learned rules that differ only in their morpho-syntactic structure

Morpho-Syntactic Variations	
X compose $Y \rightarrow X$ write Y	X is composed by $Y \rightarrow X$ write Y
X accuse $Y \leftrightarrow X$ blame Y	X's accusation of $Y \leftrightarrow X$ blame Y
X acquire $Y \rightarrow X$ obtain Y	acquisition of Y by $X \rightarrow Y$ is obtained by X

small, and their hand-picked examples do not represent the actual distribution of linguistic phenomena in any corpus.

The results are reported in Table 2. First, it is observed that the requirement for exact proof rather than embedding improves the precision considerably over the baseline (by 11.5%), while reducing the yield by nearly 40%. Remarkably, using the generic inference rules, our system is able to gain back the lost yield in Proof and further surpass the yield of the baseline configuration. In addition, a higher precision than the baseline is obtained (a 7.8% difference), which is significant at a $p < 0.05$ level, using z test for proportions. This demonstrates that our principled proof approach appears to be superior to the more heuristic baseline embedding approach, and exemplifies the contribution of our generic rule base. Overall, generic rules were used in 46% of the proofs.

Adding the lexical-syntactic rules the prover was able to increase the yield by a factor of six(!). This shows the importance of acquiring lexical-syntactic variability patterns. However, the precision of DIRT and TEASE is currently quite low, causing overall low precision. Manual filtering of rules learned by these systems is currently required in order to obtain reasonable precision .

Error analysis revealed that for the third configuration (+GEN), a significant 65% of the errors are due to parsing errors, most notably incorrect dependency relation assignment, incorrect POS assignment, incorrect argument selection, incorrect analysis of complex verbs (e.g. *play down* in the text vs. *play* in the hypothesis) and ungrammatical sentence fragments. Another 30% of the errors represent conditionals, negation and modality phenomena, most of which could be handled by additional rules, some making use of more elaborate syntactic information such as verb tense. The remaining, and rather small, 5% of the errors represent truly ambiguous sentences which would require considerable world knowledge for successful analysis.

6 Applying Entailment Inference During Rule Learning

This section describes how certain forms of entailment inferences described earlier can be utilized also to improve unsupervised learning of entailment rules (see [11] for further detail). In this setting, entailment inference yields "canonical forms" of various template variations, thus enhancing the statistical evidence that underlies learning and removing redundancies amongst the learned rules.

A noticeable phenomenon of lexical-syntactic templates is that they have many morpho-syntactic variations, which (largely) represent the same predicate and are semantically equivalent. For example, 'X compose Y' can be expressed also by 'Y is composed by X' or 'X's composition of Y'. Current learning algorithms ignore this morpho-syntactic variability. They treat these variations as semantically different and learn rules for each variation separately. This leads to several undesired consequences. First, statistics for a particular semantic predicate are scattered among different templates. This may result in insufficient statistics for learning a rule in any of its variations. Second, though rules may be learned in several variations (see Table 3), in most cases only a small part of the morpho-syntactic variations are learned. Thus, an inference system that uses only these learned rules would miss recognizing a substantial number of variations of the sought predicate.

Consequently, we propose using an entailment module that recognizes generic morphological and syntactic regularities (morpho-syntactic entailments) during learning time, and learn only canonical forms of templates and rules. Then, applying morpho-syntactic entailments also at inference time, in conjunction with the learned lexical-based canonical rules (as described earlier in this paper), guarantees the coverage of all morpho-syntactic variations of a given canonical rule.

Our proposed approach poses two advantages. First, the statistics from the different morpho-syntactic variations accumulate for a single (canoncial) template form. The improved statistics may result, for example, in learning more rules. Second, the learning output does not contain redundancies due to variations of the same predicate. Additionally, the evaluation of learning algorithms becomes more accurate, since rules learned for different variations of the same predicate are not counted seperately.

For the purpose of generating canonical templates during learning we implemented a morpho-syntactic entailment module that applies some of the inferences of the complete entailment system. These include syntactic rules for major syntactic phenomena (like passive and conjunctions) and morphological rules that address nominalizations. In the remainder of this section we provide first some necessary background about entailment rule learning; then we describe how learning is enhanced with the canonization module and evaluate its impact.

6.1 Background – Entailment Rule Learning

Many algorithms for automatically learning entailment rules and paraphrases (which can be viewed as bidirectional entailment rules) were proposed in

Table 4. Examples for features of the anchor set and single-feature approaches for two related templates

Template	**Single-feature Approach (DIRT)**		**Anchor-Set Approach**
	X-vector Features	**Y-vector Features**	**Common Features**
X compose Y	Bach, Beethoven Mozart, he	symphony, music sonata, opera	{X='Mozart'; Y='Jupiter symphony'}, {X='Bach'; Y='Sonata Abassoonata'}
X write Y	Tolstoy, Bach, author, Mozart, he	symphony, anthem, sonata, book, novel	

recent years. These methods recognize templates in texts and identify entailment relations between them based on shared features.

These algorithms may be divided into two types. The prominent approach identify an entailment relation between two templates by finding variable instantiation tuples, termed here *anchor-sets*, that are common to both templates [18,19,20,21, 17,22]. Anchor-sets are complex features, consisting of several terms, labelled by their corresponding variables. Table 4 (right column) presents common anchor-sets for the related templates 'X compose Y' and 'X write Y'. Typically, only few common anchor-sets are identified for each entailment relation.

A different *single-feature* approach is proposed by the DIRT algorithm [16]. It uses simple, less informative but more frequent features. It constructs a feature vector for each variable of a given template, representing the context words that fill the variable in the different occurrences of the template in the corpus. Two templates are identified as semantically related if they have similar vectors. Table 4 shows examples for features of this type. DIRT parses a whole corpus and limits the allowed structures of templates only to paths in the parse graphs, connecting nouns at their ends.

In this paper we report experiments with the TEASE algorithm [17]. It is an unsupervised algorithm that acquires entailment relations from the Web for given input templates using the anchor-set approach, where at least two common anchor-sets were required for learning a relation. Sentences were parsed using the Minipar dependency parser [12].

For a given input template I, the algorithm can be viewed as learning a list of output templates $\{O_j\}_1^{n_I}$, where n_I is the number of templates learned for I. Each output template is suggested as holding an entailment relation with the input template, but most current algorithms do not specify the entailment direction(s). Thus, each pair $\{I, O_j\}$ induces two candidate directional entailment rules: '$I \rightarrow O_j$' and '$O_j \rightarrow I$'.

The learned entailment rules and paraphrases can be used at *inference time* in applications such as IE [19,23,24] and QA [16,18,25], where matched rules deduce new target predicate instances from texts (like the 'compose → write' example in the beginning of Section 6). As shown in previous evaluations the precision of algorithms like DIRT and TEASE is still limited [16,20,17,26]. Currently, utilizing their output in applications may require manual filtering of the learned rules, and the algorithms' utility is reflected mainly by the amount of correct rules they learn.

Current methods for learning lexical-syntactic rules do not address the morpho-syntactic variability at learning time. Thus, they learn rules separately for each template variation. This results in either learning redundant rules (see Table 3) or missing some of the relevant rules that occur in a corpus. Moreover, some rules might not be learned in any variation. For example, if for each of the rules 'X acquire Y → X own Y', 'Y is acquired by X → X own Y' and 'X's acquisition of Y → X own Y' there is just anecdotal, but insufficient, statistical evidence then none of them will be learned. On the other hand, accumulating the statistics for all rules together may suffice to enable their learning.

Table 5. Some of the syntactic rules used in our implementation, together with usage examples (the application of the second rule and the third rule is demonstrated in Figure 3)

Rule	Original Template	Simplified Template
passive to active	$X \xleftarrow{pcomp-n} by \xleftarrow{by-subj} find \xrightarrow{obj} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$
conjunction	$X \xleftarrow{subj} find \xrightarrow{obj} gold \xrightarrow{conj} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$
apposition	$X \xleftarrow{subj} find \xrightarrow{obj} protein \xrightarrow{appo} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$
abbreviation	$X \xleftarrow{subj} find \xrightarrow{obj} NDA \xrightarrow{spellout} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$

6.2 The Morpho-syntactic Canonization Module

In our scheme, we use a morpho-syntactic entailment module to transform lexical-syntactic template variations that occur in a text into their *canonical form.* This form, which we chose to be the active verb form with direct modifiers, is entailed by other template variations.

We implemented the canonization module based on a set of *canonization rules*, which are highly accurate morpho-syntactic entailment rules. Each rule represents one morpho-syntactic regularity that is eliminated when the rule is applied to a given template (see examples in Table 5 and Figure 3).

The current canonization rule collection consists of two types of rules: (a) syntactic-based rules; (b) morpho-syntactic nominalization rules. We next describe each rule type. As we use the Minipar parser, all rules are adapted to Minipar's output format.

Syntactic-based Rules. These rules capture entailment patterns associated with common syntactic structures. Their function is to simplify and generalize the syntactic structure of a template.

In the current implementation we manually created the following simplification rules: (a) passive forms into active forms; (b) removal of conjunctions; (c) removal of appositions; (d) removal of abbreviations; (e) removal of set description by the 'such as' preposition. Table 5 presents some of the rules we created together with examples of their effect.

Nominalization Rules. Entailment rules such as 'acquisition of Y by $X \rightarrow X$ acquire Y' and 'Y's acquisition by $X \rightarrow X$ acquire Y' capture the relations between verbs and their nominalizations. We automatically derived these rules from Nomlex, a hand-coded database of about 1000 English nominalizations [15], as described in [14]. These rules transform any nominal template in Nomlex into its related verbal form, preserving the semantics of the original template predicate. We chose the verbal form as the canonical form since for every predicate

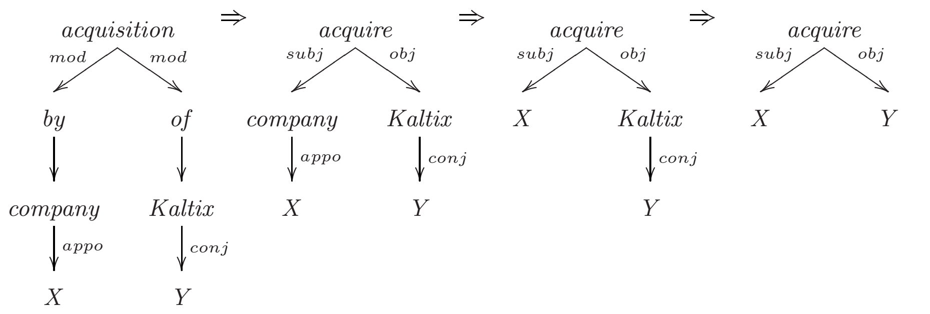


Fig. 3. Chaining of canonization rules that transforms the path template between the arguments {X=‘Google’;Y=‘Sprinks’}, which occurs in the sentence “*We witnessed the acquisition of Kaltix and Sprinks by another growing company, Google*”, into a canonized template form. The first rule applied is a nominalization rule, followed by removal of apposition and removal of conjunction (as described in Table 5). As can be seen, applying the rules in any order will result in the same final canonical form.

with specific semantic modifiers there is only one verbal active form in Nomlex, but typically several equivalent nominal forms.

Chaining of Canonization Rules. Each of the syntactic rules we implemented decreases the size of a template. In addition, nominalization rules can be applied only once for a given template, since no rule in our rule-set transforms a verbal template into one of its nominal forms. Thus, applying rules until no rule can apply is a finite process. In addition, each of our rules is independent of the others, operating on a different set of dependency relations. Consequently, chaining of canonization rules is a well-defined procedure, since applying any sequence of rules until no other rule can apply will result in the same final canonical template form. Figure 3 illustrates an example for rule chaining.

6.3 Applying the Canonization Module

When morpho-syntactic entailment rules are utilized at inference time (e.g. [23]), they recognize a closure of morpho-syntactic variations for a lexical-syntactic template. Accordingly, acquisition algorithms may learn just a single morpho-syntactic variation of an entailment rule.

With this modular scheme in mind, we propose to solve the learning problems discussed in Subsection 6.1 by utilizing the morpho-syntactic entailment module at learning time as well. We incorporate the module in the learning algorithm by converting each template variation occurrence in the learning corpus into an occurrence of a canonical template. Consequently, the learning algorithms operate only on canonical forms.

As discussed earlier, when canonization is used no morpho-syntactically redundant rules are learned, with respect to the variations that are recognized by the module. This makes the output more compact, both for storage and for use. In addition, the statistical reliability of learned rules may be improved. For example, rules that could not be learned for any particular variation may be learned now for the canonical form.

Methodologically, previous evaluations of learning algorithms reported accuracy relative to the redundant list of rules, which creates a bias for templates with many frequent variations. When this bias is removed and only truly different lexical-syntactic rules are assessed, evaluation is more efficient and accurate.

6.4 Evaluation

Human Judgements. We have selected 20 different verbs and verbal phrases[4] as input templates for TEASE (see [11] for an additional evaluation based on the DIRT algorithm). We executed its baseline version (without canonization), denoted by $TEASE_b$, as well as the version with the canonization module, denoted by $TEASE_c$. The results of these two executions constitute our test-set rules. To enable the comparison between the outputs of the two version we converted the output templates of $TEASE_b$ into their canonical forms (that is, this canonization is not considered as part of the $TEASE_b$ algorithm, and was not exploited during learning).

As discussed above, TEASE does not learn the direction(s) of an entailment relation between an input template I and a learned output template O. Thus, we evaluated both candidate directional rules, '$I \rightarrow O$' and '$O \rightarrow I$'.

Rule Evaluation. The prominent *rule based* approach for evaluating entailment rules is to present them to human judges, who assess whether each rule is correct or not. Generally, a rule is considered correct if the judge could think of reasonable contexts under which it holds. However, it is difficult to define explicitly when a learned rule should be considered correct under this methodology.

Instead, we follow the *instance based* evaluation methodology presented in [26]. By this methodology, each rule '$L \rightarrow R$' is evaluated by presenting the judges not only with the rule itself but rather with a sample of sentences that match its left hand side L. The judges then assess whether the rule is valid under each specific example sentence. The precision of a rule is computed by the percentage of examples for which entailment holds out of all "relevant" examples in the judged sample. The rule is then considered correct if its precision is higher than 0.8 (see [26] for details). This instance-based approach for human judgment was shown to be more reliable than the rule-based approach.

Evaluation Setup. We separated the templates that were learned by $TEASE_c$ into two lists: (a) a *baseline-templates* list containing templates learned also by $TEASE_b$; (b) a *new-templates* list containing templates that were not learned by $TEASE_b$, but learned by $TEASE_c$ thanks to the improved accumulated

[4] The verbs are: accuse, approve, calculate, change, demand, establish, finish, hit, invent, kill, know, leave, merge with, name as, quote, recover, reflect, tell, worsen, write.

Table 6. Average Precision and Yield of the output lists

Template List	Avg. Precision	Avg. Yield
$TEASE_b$	30.1%	49.8
$TEASE_c$	28.7%	55.6

statistics for canonical forms. In total, 3871 (unique) canonical templates were learned: 3309 in the baseline-templates list and 562 in the new-templates list. Inherently, every output template learned by $TEASE_b$ is also learned in its canonical form by $TEASE_c$, since its supporting statistics may only increase.

We randomly sampled 100 templates from each list and evaluated their correctness according to the human judgment methodology. To that end we retrieved 10 example sentences for each rule from the first CD of Reuters RCV1. Two judges, fluent English speakers, evaluated the examples. We randomly split the rules between the judges with 100 rules (942 examples) cross annotated for agreement measurement.

Results. First, we measured the redundancy in the rules learned by $TEASE_b$, that is, the percentage of output templates that could be removed due to redundancy of morpho-syntactic variations, to be 6.2% per input template on average. This redundancy was eliminated using the canonization module.

Next, we evaluated the quality of the sampled rules using two scores: (1) micro average ***Precision***, the percentage of correct templates out of all learned templates, and (2) average ***Yield***, the average expected number of correct templates learned for each input template, as extrapolated based on the sampling proportion. The results are presented in Table 6. The agreement between the judges was measured by the Kappa value [27], obtaining a 0.67 score (corresponding to substantial agreement).

We expect $TEASE_c$ to learn new rules using the canonization module. In our experiment, 5.8 more correct templates were learned on average per input template by $TEASE_c$. This corresponds to an increase of 11.6% in average Yield (see Table 6). Examples of new correctly learned templates are shown in Table 7.

There is a slight decrease in precision when using $TEASE_c$. One possible reason is that the new templates are usually learned from very few occurrences of different variations, accumulated for the canonical templates. Thus, they may have a somewhat lower precision in general. Overall, the significant increase in Yield is much more important, especially if the learned rules are later filtered manually.

6.5 Analysis

Parser errors are one of the main reasons that variations are sometimes not transformed into their canonical form. These errors result in different parse trees for the same syntactic construct. Thus, several parser-dependent rules may be needed to capture the same syntactic structure. Moreover, it is difficult to design

Table 7. Examples for correct templates that TEASE learned only after using canonization rules

Input Template	Learned Template
X accuse Y	X blame Y
X approve Y	X take action on Y
X demand Y	X call for Y, X in demand for Y
X establish Y	X open Y
X hit Y	X slap Y
X invent Y	grant X patent on Y, X is co-inventor of Y
X kill Y	X hang Y, charge X in death of Y
X named as Y	hire X as Y, select X as Y
X quote Y	X cite Y
X tell Y	X persuade Y, X say to Y
X worsen Y	X impair Y

canonization rules for some parsing errors, since the resulting parse trees consist of structures that are common to other irrelevant templates. For example, when Minipar chooses the head of the conjunct 'Y' in "*The interaction between X and Y will not hold for long*" to be '*interaction*' rather than 'X', the appropriate canonization rule cannot be applied. These errors affect both the learning phase, where statistics are not accumulated for the appropriate canonical form, and the inference phase, where variations of a canonical rule template are not recognized.

Finally, we note that the reported results correspond only to the phenomena captured by our currently implemented canonization rules. Adding rules that cover more morpho-syntactic phenomena is expected to increase the benefit of the canonization scheme. For example, there are many nominalizations that are not specified in the current Nomlex version, but can be found in other resources, such as WordNet [28].

7 Conclusions

In this paper we presented a framework for semantic inference at the lexical-syntactic level, suggesting that natural language based structures may be suitable for meaning representation and inference. Our formalism was found suitable for describing a wide spectrum of inference knowledge, in the form of entailment rules, both automatically derived and manually created. We also presented a much-needed evaluation methodology for individual components in knowledge-based inference systems. Finally, we showed that the inference module can be utilized also for improving unsupervised acquisition of entailment rules through canonization, which in turn enhances the scope of the inference system.

In future work we plan to enhance the framework to allow inference from multiple sentences. We will also investigate integration of the proof system with different methods for approximate matching, which would enable its application in additional settings, as was demonstrated initially in [8] for the RTE-3 benchmark.

Acknowledgements

We are grateful to Cleo Condoravdi for making the polarity lexicon developed at PARC available for this research. We also thank Dan Roth for helpful discussions, and Chen Erez for her help in the experiments. This work was partially supported by ISF grant 1095/05, the Israel Internet Association (ISOC-IL) grant 9022, the Israeli Ministry of Industry and Trade under the NEGEV Consortium (www.negev-initiative.org) and the IST Programme of the European Community under the PASCAL Network of Excellence IST-2002-506778.

References

1. Dagan, I., Glickman, O., Magnini, B.: The pascal recognising textual entailment challenge. In: Quiñonero-Candela, J., Dagan, I., Magnini, B., d'Alché-Buc, F. (eds.) MLCW 2005. LNCS (LNAI), vol. 3944, pp. 177–190. Springer, Heidelberg (2006)
2. Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., Szpektor, I.: The second pascal recognising textual entailment challenge. In: Second PASCAL Challenge Workshop for Recognizing Textual Entailment (2006)
3. Giampiccolo, D., Magnini, B., Dagan, I., Dolan, B.: The third pascal recognizing textual entailment challenge. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing (2007)
4. Raina, R., Ng, A.Y., Manning, C.D.: Robust textual inference via learning and abductive reasoning. In: Proceedings of AAAI (2005)
5. Tatu, M., Moldovan, D.: A logic-based semantic approach to recognizing textual entailment. In: Proceedings of COLING-ACL (2006)
6. Bos, J., Markert, K.: When logical inference helps determining textual entailment (and when it doesn't). In: Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment (2006)
7. Bar-Haim, R., Dagan, I., Greental, I., Shnarch, E.: Semantic inference at the lexical-syntactic level. In: Proceedings of AAAI (2007)
8. Bar-Haim, R., Dagan, I., Greental, I., Szpektor, I., Friedman, M.: Semantic inference at the lexical-syntactic level for textual entailment recognition. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing (2007)
9. Valencia, V.S.: Parsing-driven inference: natural logic. Linguistic Analysis 25, 258–285 (1995)
10. MacCartney, B., Manning, C.D.: Natural logic for textual inference. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague, pp. 193–200, Association for Computational Linguistics (2007)
11. Szpektor, I., Dagan, I.: Learning canonical forms of entailment rules. In: Proceedings of RANLP (2007)
12. Lin, D.: Dependency-based evaluation of minipar. In: Proceedings of the Workshop on Evaluation of Parsing Systems at LREC (1998)
13. Nairn, R., Condoravdi, C., Karttunen., L.: Computing relative polarity for textual inference. In: Proceedings of ICoS-5 (2006)

14. Ron, T.: Generating entailment rules using online lexical resources. Masterś thesis, Computer Science Department, BarIlan University (2006)
15. Macleod, C., Grishman, R., Meyers, A., Barrett, L., Reeves, R.: Nomlex: A lexicon of nominalizations. In: EURALEX (1998)
16. Lin, D., Pantel, P.: Discovery of inference rules for question answering. Natural Language Engineering 7(4), 343–360 (2001)
17. Szpektor, I., Tanev, H., Dagan, I., Coppola, B.: Scaling web-based acquisition of entailment relations. In: Proceedings of EMNLP (2004)
18. Ravichandran, D., Hovy, E.: Learning surface text patterns for a question answering system. In: Proceedings of ACL (2002)
19. Shinyama, Y., Sekine, S., Kiyoshi, S., Grishman, R.: Automatic paraphrase acquisition from news articles. In: Proceedings of HLT (2002)
20. Barzilay, R., Lee, L.: Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In: Proceedings of HLT-NAACL (2003)
21. Quirk, C., Brockett, C., Dolan, W.: Monolingual machine translation for paraphrase generation. In: Proceedings of EMNLP (2004)
22. Sekine, S.: Automatic paraphrase discovery based on context and keywords between ne pairs. In: Proceedings of IWP (2005)
23. Romano, L., Kouylekov, M., Szpektor, I., Dagan, I., Lavelli, A.: Investigating a generic paraphrase-based approach for relation extraction. In: Proceedings of EACL (2006)
24. Sekine, S.: On-demand information extraction. In: Proceedings of the COLING/ACL Main Conference Poster Sessions (2006)
25. Harabagiu, S., Hickl, A.: Methods for using textual entailment in open-domain question answering. In: Proceedings of ACL (2006)
26. Szpektor, I., Shnarch, E., Dagan, I.: Instance-based evaluation of entailment rule acquisition. In: Proceedings of ACL (2007)
27. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20, 37–46 (1960)
28. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. Language, Speech and Communication. MIT Press, Cambridge (1998)

Layer Structures and Conceptual Hierarchies in Semantic Representations for NLP

Hermann Helbig, Ingo Glöckner, and Rainer Osswald

Intelligent Information and Communication Systems
FernUniversität in Hagen, Germany

Abstract. Knowledge representation systems aiming at full natural language understanding need to cover a wide range of semantic phenomena including lexical ambiguities, coreference, modalities, counterfactuals, and generic sentences. In order to achieve this goal, we argue for a multidimensional view on the representation of natural language semantics. The proposed approach, which has been successfully applied to various NLP tasks including text retrieval and question answering, tries to keep the balance between expressiveness and manageability by introducing separate semantic layers for capturing dimensions such as facticity, degree of generalization, and determination of reference. Layer specifications are also used to express the distinction between categorical and situational knowledge and the encapsulation of knowledge needed e.g. for a proper modeling of propositional attitudes. The paper describes the role of these classificational means for natural language understanding, knowledge representation, and reasoning, and exemplifies their use in NLP applications.

1 Introduction

Although envisaged since the early days of automated natural language processing (NLP), there are currently only a few implemented systems that aim at a full semantic analysis of unrestricted natural language. One of the reasons for this situation may be the diversification in formal semantics with its highly elaborate but specialized theories focusing on specific semantic phenomena such as presuppositions, generics, pluralities or modalities (see e.g. [1]), whereas building language understanding systems calls for a uniform and concise formalism covering all aspects of natural language semantics up to a certain degree of granularity. A second reason may be the current dominance of shallow NLP even in areas where traditionally deep semantic analysis has been taken as a *sine qua non*. For instance, many approaches to open-domain textual question answering rest mainly on text retrieval techniques with no or little analysis of the underlying documents. However, this approach will fail if the relevant information is mentioned only once in the text collection, has no lexical overlap with the question and uses other syntactic constructions, or is distributed over several sentences linked by anaphora.

In this paper, we argue for a concept-centered representation of natural language semantics that employs a moderate amount of reification and represents semantic aspects like plurality or facticity by ontological features whenever appropriate. The proposed approach is explicated by a slightly simplified version of the *MultiNet* knowledge representation formalism, which is described in detail in [2]. MultiNet has been designed for representing the semantics of unregimented text (*universality criterion*) while being applicable at all stages of linguistic analysis, knowledge representation, inference processes, and natural language generation (*interoperability criterion*).

The MultiNet formalism has been employed for building a large semantically-based computational lexicon [3] as well as a syntactico-semantic parser [4] that generates MultiNet representations of natural language input. The lexicon and the parser are the core component of several NLP systems, ranging from question answering systems [5,6] over natural language interfaces [7] to systems for checking text readability [8].

2 Sorts, Relations, and Conceptual Hierarchies

This section briefly introduces part of the sortal and relational inventory of the MultiNet formalism in order to provide a basis for explaining the use of layer structures in Sections 3 and 4.

2.1 Ontological Sorts and Semantic Relations

The definition of a formalism for knowledge representation and natural language semantics, usable in all parts of an NLP system, requires a clear characterization of its basic categories. On the one hand, one needs a classification of the concepts to be dealt with (yielding the sorts of conceptual entities shortly described in this subsection). On the other hand, one has to define the fundamental relations and functions holding between these entities, using these sorts for domain and range specifications. The most basic means for conceptual classification used in MultiNet is a tree shaped hierarchy of *ontological sorts*, which can be regarded as the backbone of an upper ontology. Part of the MultiNet sort hierarchy is shown in Fig. 1 (see [2, Chap. 17] for the full hierarchy). The use of these sorts for specifying the signatures of relations shown in Table 1 explains some of the proposed distinctions. Consider, for example, the discernment of ordinary properties (*p*) vs. relational qualities (*rq*). Concepts of sort *rq*, like *equivalent*, are different from regular properties in that they can only qualify collections. Thus ordinary properties will be expressed by PROP and relational qualities by a relation PROPR governed by different axioms (see Table 2). The refinement of a sort into subsorts sometimes reflects differences in the inheritance patterns. As shown by axiom (b) in Table 2, gradable properties (sort gq) and total properties (sort tq) are distinguished for that reason. In contrast to upper domain ontologies like DOLCE [9], the hierachy of ontological sorts proposed here does not cover concepts like *sets*, *collections* or *aggregates*, because building pluralities and collections is possible across different sorts and should thus be seen as orthogonal to the basic sortal hierarchy. Therefore, plurality is expressed by an *ontological feature* as a separate classificational dimension (see Sect. 3.3).

Based on general requirements like universality, homogeneity and granularity (see [2, Chapt. 1]), to be fulfilled by a formalism for natural language semantics, MultiNet comes up with a system of more than hundred relations and functions for modeling relationships between conceptual entities [2]. These comprise basic structuring relations (for subordination and meronymy), argument roles (for describing participants of a situation), temporal and local relations (for describing spatio-temporal aspects) and finally

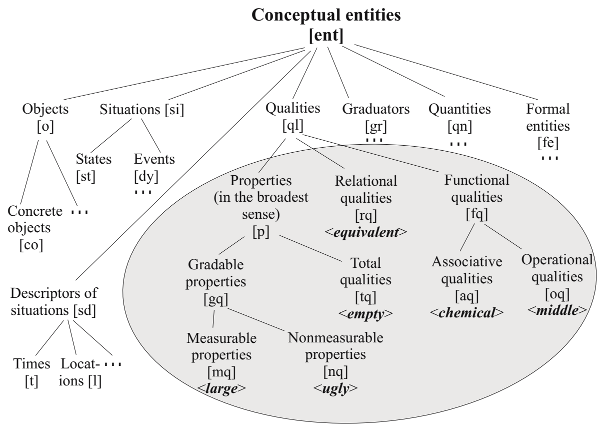


Fig. 1. Part of the MultiNet sort hierarchy (with emphasis on qualities)

intersituational relations (for describing causes, circumstantials and modal embeddings), see Table 1.[1] The commitment to a fixed catalogue of relations and functions fosters the long-term development of linguistic resources and knowledge bases (e.g. of the computational lexicon HaGenLex [3]), as well as the interoperability of applications [7].

Fig. 2 illustrates the use of these relations for describing the meaning of an example sentence, automatically generated by the WOCADI parser [4].[2] In particular, the example shows that all nodes which represent instances (i.e. $c1, c2, \dots$) are described in terms of the pre-defined relations and the concept constants (like *analyst*, *meeting*...) defined in the lexicon. This warrants a tight coupling between knowledge representation and the computational lexicon, on the one hand (homogeneity property), and a smooth interfacing between syntactic semantic analysis and following inference processes in a question-answering system on the other hand (interoperability property). The figure also displays the ontological features FACT, GENER, ETYPE of nodes and the characterization of edges explained in Sections 3 and 4.

2.2 Conceptual Hierarchies

There is a general consensus that the *subordination* of concepts is the most important relation governing ontologies and hierarchies of concepts. In Description Logics [10], this relation is extensionally interpreted as an inclusion of extensions and called subsumption. In our formalism, we assume a single subordination relation SUB for conceptual

[1] Some of the signature definitions given in the table make use of ontological features, which is indicated by the following notational conventions: The notation $\ddot{\sigma}$ restricts an argument to collections of sort σ. It is needed to describe the combinatorics of PROPR. The specification $\overline{\sigma}$ indicates generic, i.e. non-instantiated concepts of sort σ. It is used in the signature of SUB and SUBS to ensure that the second argument is a generic concept (see Sect. 3.2).

[2] Notice that some of the relationships, e.g. SUB$(c_1, \mathit{analyst})$, are not displayed as directed edges but rather listed below the node name.

Table 1. Simplified description of relations used in this paper

Relation	Signature	Short Characteristics
AGT	$si \times co$	Agent
ATTR	$[o \cup l \cup t] \times at$	Specification of an attribute
CIRC	$si \times si$	Relation between situation and circumstance
COND	$si \times si$	Conditional relation
ELMT	$ent \times \ddot{ent}$	Membership (of extensions)
EXP	$si \times o$	Experiencer
HSIT	$si \times si$	Composition of situations
LOC	$co \times l$	Location of an object
MCONT	$si \times [si \cup o]$	Mental or informational content
MEXP	$st \times co$	Mental experiencer
ORIGM	$co \times co$	Relation of material origin
PARS	$co \times co$	Part-whole relation for concrete objects
PROP	$o \times p$	Relation between object and property
PROPR	$\ddot{o} \times qr$	Relation between plurality and relational property
SUB	$o \times \overline{o}$	Relation of conceptual subordination (for objects)
SUBM	$\ddot{ent} \times \ddot{ent}$	Subsumption (inclusion of extensions)
SUBS	$si \times \overline{si}$	Relation of conceptual subordination (for situations)
TEMP	$si \times [si \cup t]$	Relation of temporal containment
VAL	$at \times [qn \cup p \cup fe]$	Relation between attribute and value

Table 2. Selected axioms related to subordination

(a)	$\text{SUB}(o_1, o_2) \wedge \text{SUB}(o_2, o_3) \rightarrow \text{SUB}(o_1, o_3)$
(b)*	$\text{SUB}(o_1, o_2) \wedge \text{PROP}(o_2, p) \wedge p \in \text{tq} \rightarrow \text{PROP}(o_1, p)$
(c)	$\text{SUB}(o_1, o_2) \wedge \text{ATTR}(o_2, a) \rightarrow \exists a'[\text{SUB}(a', a) \wedge \text{ATTR}(o_1, a')]$
(d)*	$\text{SUB}(o_1, o_2) \wedge \text{ORIGM}(o_2, s_2) \rightarrow \exists s_1[\text{SUB}(s_1, s_2) \wedge \text{ORIGM}(o_1, s_1)]$

objects, which we characterize axiomatically in order to avoid this assumption of extensional interpretation. We are well aware of the necessity to separate instantiation and specialization of concepts as pointed out by Brachman [11]. In our system, instantiation corresponds to a subordination relationship $\text{SUB}(o_1, o_2)$ where o_1 is a specific entity and o_2 is generic. Specialization, by contrast, is modelled by a subordination relationship $\text{SUB}(o_1, o_2)$ where both participants are generics. In this way, we cleanly distinguish the two flavours of subordination, while avoiding a duplication of axioms valid in both cases. Table 2 lists basic axioms related to subordination. (Axioms marked by an asterisk have default status only). Axiom (a) expresses the transitivity of SUB. Inheritance of properties is described by (b). The term $p \in \text{tq}$ restricts applicability to total properties. This prevents a wrong conclusion that an object which is both a small animal and an ant, can be described as a small ant. Axiom (c) formalizes inheritance of attributes. For example, the attribute of eye color is inherited by all instances of *human*. Finally, (d) expresses inheritance of material origin: if ORIGM(*window*, *glass*), i.e. windows are made of glass, then a concrete window is also made of glass. From the SUB relation we distinguish another subordination relation (in MultiNet called SUBS) which carries

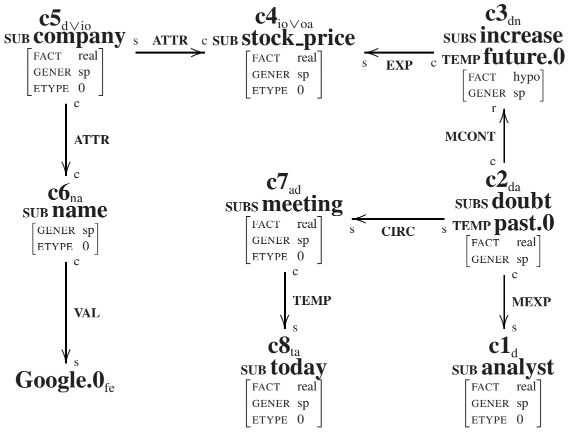


Fig. 2. MultiNet representation of the sentence "*On today's meeting, the analyst doubted that Google's stock price would increase*"

Table 3. Selected axioms related to meronymy relations

(a)*	PARS$(k_1, k_2) \wedge$ PARS$(k_2, k_3) \rightarrow$ PARS(k_1, k_3)
(b)	SUB$(d_1, d_2) \wedge$ PARS$(d_3, d_2) \rightarrow \exists d_4$[SUB$(d_4, d_3) \wedge$ PARS(d_4, d_1)]
(c)*	PARS$(k_1, k_2) \wedge$ LOC$(k_2, \ell) \rightarrow$ LOC(k_1, ℓ)
(d)*	PARS$(k_1, k_2) \wedge$ ORIGM$(k_2, s) \rightarrow$ ORIGM(k_1, s)
(e)	ELMT$(k_1, k_2) \wedge$ ELMT$(k_2, k_3) \rightarrow \neg$ ELMT(k_1, k_3)
(f)	HSIT$(k_1, k_2) \wedge$ HSIT$(k_2, k_3) \rightarrow$ HSIT(k_1, k_3)
(g)*	HSIT$(s, p) \wedge$ LOC$(s, \ell) \rightarrow$ LOC(p, ℓ)
(h)	HSIT$(s, p) \wedge$ COND$(c, s) \rightarrow$ COND(c, p)
(i)	HSIT$(s, p) \wedge$ CIRC$(e, p) \rightarrow$ CIRC(e, s)

the conceptual hierarchy for situational entities. This relation, for instance, connects the activities go and move (oneself). The main characteristics of SUBS is the *inheritance of argument structure* of superconcepts by subconcepts.

Besides subsumption or conceptual subordination, *part-whole relations* traditionally play an important role in knowledge representation. The proper way of reasoning along part-whole hierarchies has been discussed extensively in the literature [12]. In particular, it has been observed that transitivity violations are often due to mixing up different types of part-whole relations. We thus discern several mereological relations which capture the different ways in which a part can be related to a whole, viz PARS (physical parts of concrete objects), ELMT (membership of an element in a collection), HSIT (relationship between a complex situation and its parts), and the loosely related SUBM (containment of a collection in another collection) and ORIGM (which relates an object and the material it is made of). See Table 3 for basic characteristics of these relations. Axiom (a) asserts the limited transitivity of PARS. Due to the default status of the axiom, transitive chains of PARS relationships can only be followed if no predefined epistemic or functional borderlines are transgressed. Axiom (b) describes the inheritance of parts in a SUB-hierarchy, axiom (c) asserts that a part is normally co-located with the whole. It is also plausible to assume that the part consist of the same material as the whole, see axiom (d). Axiom (e) asserts the intransitivity of membership. Axioms (f) through (j) describe the properties of situational composition HSIT(s_1, s_2), expressing the embedding of a subsituation s_2 in an enclosing situation s_1. Axiom (f) states that HSIT be transitive (i.e. scoring a goal in the final match of a soccer championship is also part of the championship), axiom (g) asserts that if the soccer world championship takes place in Germany, then the final of this championship also takes place in Germany, and axiom (h) states that if c is a sufficient condition for s to take place (expressed by the relation COND), then c is also sufficient for situational parts of s to take place. Finally, axiom (i) describes a widening of circumstantials for a given situation: If the goalkeeper becomes injured in the final match of the 2006 world championship (subsituation), then he becomes injured in the 2006 world championship (enclosing situation).

3 Layer Structures and Their Utility to NLP

The classification of concepts in terms of sorts, subordination, and meronymy is too weak to cope with complex linguistic phenomena like quantification, scope, modal embeddings and propositional attitudes, presuppositions etc. Here we focus on the aspects of facticity (real vs. nonreal or hypothetical entities), genericity (concrete instances vs. generic abstractions) and extensionality or plurality (i.e. individuals vs. collections). Due to the independence of these aspects from the hierarchy-forming relations of subordination and meronymy, we treat these aspects as *ontological features* defined for all conceptual entities of suitable sort. This annotation effects a stratification of the conceptual entities into *layers* of those entities which share the same value of a given layer feature. The feature-based representation lends itself to the compositional construction of a semantic representation by means of unification and is thus suited for automated semantic analysis. Moreover these representations are simple and robust enough to provide an immediate benefit for NLP applications.

3.1 Facticity

One important aspect of NL meaning which is not accounted for by the relational representation is concerned with the *facticity* of the entities mentioned in the discourse. Within natural language we can discuss the existence or non-existence of objects of a certain kind – like ghosts, aliens or black holes. It is also possible to talk about situations or events regardless of facticity. Consequently, Hobbs [13] emphasizes the importance of modeling non-existing objects for knowledge representation. In predicate logic, however, there is an inherent existence assumption for all individuals (i.e. individual constants) because the model-theoretic semantics demands that all constants denote in the universe. The ontological feature FACT therefore expresses the facticity of a considered entity. The FACT feature discerns three cases: real entities [FACT *real*], non-existing entities [FACT *nonreal*] and hypothetical entities [FACT *hypo*]. The third value *hypo* is needed e.g. to describe states of affairs in conditionals (relation COND), e.g. "*If*

Table 4. Selected axioms related to facticity

(a)	$R(x, x') \wedge \mathrm{FACT}(x) = \mathit{real} \rightarrow \mathrm{FACT}(x') = \mathit{real}$ where $R \in \{\mathrm{AGT}, \mathrm{EXP}, \mathrm{MEXP}, \ldots\}$ (most argument roles)
(b)*	$\mathrm{MCONT}(x, x') \rightarrow \mathrm{FACT}(x') = \mathit{hypo}$
(c)	$\mathrm{SUBS}(s, \mathit{know}) \wedge \mathrm{MCONT}(s, s') \wedge \mathrm{FACT}(s) = \mathit{real} \rightarrow \mathrm{FACT}(s') = \mathit{real}$
(d)	$R(x, x') \wedge \mathrm{FACT}(x) = \mathit{real} \rightarrow \mathrm{FACT}(x') = \mathit{real}$ where $R \in \{\mathrm{PARS}, \mathrm{ELMT}, \mathrm{HSIT}\}$ (meronymy relations)
(e)*	$\mathrm{COND}(s, s') \rightarrow \mathrm{FACT}(s) = \mathit{hypo} \wedge \mathrm{FACT}(s') = \mathit{hypo}$

John returns late, he will be tired." In this case, there is obviously no commitment as to the facticity of premise and conclusion, which are thus classified *hypo*.[3]

Table 4 presents some basic axioms related to facticity. Axiom (a) states that the facticity of a situational concept demands the facticity of its argument roles. For example, Maria can only believe in something if she is a real person. The same can also be said about LOC and TEMP, i.e. a real situation takes place at a real location and has a temporal extent in the real world. An exception to axiom (a) is the MCONT role which expresses mental content or modal embedding. Here, Maria's belief in the existence of the Yeti gives no indication as to the actual existence of the Yeti, which is classified *hypo* by axiom (b). However, as shown by (c), there are exceptional cases (e.g. the mental state of knowing something) which justify this conclusion. Axiom (d) states that a real aggregate can only be composed of real parts. Finally, axiom (e) expresses the above observation that if-then-relationships involve hypothetical situations.

Utilizing Facticity Information for NLP. There is growing interest in supporting question answering and, more generally, systems for recognizing textual entailment by an explicit modeling of facticity and facticity-related inferences. Successful examples of integrating facticity-related information are de Marneffe et al [14], whose system checks if an embedded proposition has a non-factive verb as a parent, and Hickl and Bensley [15], whose discourse commitment-based approach considers presuppositions including conversational implicatures. Bobrow et al [16] propose a logic for textual inference which uses 'instantiability conditions' to express facticity. Nairn et al [17] present a typology of factive and implicative English verbs which was formalized in that logic.

Klutsch [18] presents a typology of factive and implicational verbs which is a refinement of the proposal of Nairn et al. The typology was used for a classification of the German verbs represented in the computational lexicon HaGenLex [3]. Moreover a system of facticity propagation rules was developed which makes it possible to derive unknown facticity values in complex sentences from known FACT values of other entities, from facticity projection properties of the involved participant roles, and from the presence of implicative or factive verbs of a certain type.

This facticity annotation based on the FACT feature is used in the MAVE answer validator, which is part of the IRSAW question answering system [6]. Suppose the system

[3] Based on the hypothetical/real distinction, one can also approach the modelling of counterfactuals, see [2, Chap. 12.3].

Table 5. Selected axioms related to genericity

(a)	$R(x, x') \rightarrow \text{GENER}(x) = \text{GENER}(x')$ where $R \in \{\text{AGT}, \text{EXP}, \text{MEXP}, \dots\}$ (most argument roles)
(b)	$\text{MCONT}(s, s') \wedge \text{GENER}(s') = \textit{sp} \rightarrow \text{GENER}(s) = \textit{sp}$
(c)	$R(s, d) \wedge \text{GENER}(s) = \textit{sp} \rightarrow \text{GENER}(d) = \textit{sp}$, $R \in \{\text{LOC}, \text{TEMP}\}$
(d)	$R(x, x') \rightarrow \text{GENER}(x) = \text{GENER}(x')$ where $R \in \{\text{PARS}, \text{ELMT}, \text{HSIT}\}$ (meronymy relations)
(e)	$R(x, x') \rightarrow \text{GENER}(x') = \textit{ge}$, for $R \in \{\text{SUB}, \text{SUBS}\}$

is asked *"Who escaped Mozambique jail?"* and suppose that the answer candidate *"the Cardoso killer"* was found in a text passage *"The Cardoso killer again tried to escape Mozambique jail."* Then the answer is not validated by the text since the question relates to situations of successful escaping marked by [FACT *real*]. There is no evidence for this facticity value in the text (assuming conventional use of "*try*", it was probably not successful). MAVE accounts for that by decreasing the rank of an answer if it depends on hypothetical information or information about non-existing entities [6].

3.2 Degree of Generality

Another representational challenge to be mastered is concerned with the genericity (degree of generality) of a conceptual entity. A possible starting point for discussing genericity is the T-Box/A-Box distinction known from Description Logics [10], which essentially separates assertions about individual instances (in the A-Box) from knowledge which is universally valid for arbitrary instances of a concept (terminological knowledge in the T-Box). However, consider the example "*Quicksilver was known to the ancient Chinese*". The sentence does not express general knowledge about instances of *quicksilver* but rather about an abstraction which represents quicksilver in a more general sense. We thus consider generic entities in addition to the familiar specific entities. By introducing the genericity attribute GENER which divides the world of conceptual entities into generics [GENER *ge*] and specific individuals [GENER *sp*], assertions about the generic concept and about instances of that concept can be cleanly separated. Generic entities also serve to model prototypical knowledge. Consider the sentence "*The squirrel likes eating nuts*". Universal quantification over all instances of squirrels would be inadequate because the sentence expresses default knowledge only. In our formalism, however, one can use a generic entity to model a squirrel which acts as a prototype. Table 5 lists axioms related to genericity. Axiom (a) asserts that a situation and its agent or (mental) experiencer must have the same genericity type. This property of co-genericity is shown by most argument roles of situations, but MCONT (mental content) is an exception. The example "*Peter does not know that the squirrel likes nuts*" shows that a specific propositional attitude (Peters knowledge about squirrels eating nuts) can embed a generic sentence "*the squirrel likes eating nuts*". Still, MCONT satisfies axiom (b), i.e. if a mental content is specific, then the embedding mental situation is also specific. The converse pattern (c) is shown by LOC and TEMP. In fact, generic sentences like "*The polar bear lives in the arctic*" or "*The dinosaur died out at the end*

Table 6. Selected axioms related to the extensionality type

(a)	ELMT$(e, c) \rightarrow$ ETYPE$(c) =$ ETYPE$(e) + 1$
(b)	SUBM$(c, c') \rightarrow$ ETYPE$(c) > 0$
(c)	SUBM$(c, c') \rightarrow$ ETYPE$(c) =$ ETYPE(c')
(d)	PROPR$(c, p) \rightarrow$ ETYPE$(c) > 0$
(e)	SUBS$(s, \langle$*die out*$\rangle) \wedge$ EXP$(s, e) \rightarrow$ ETYPE$(e) > 0 \vee$ GENER$(e) =$ *ge*

of the mesozoikum" refer to specific regions in time or space. However, a concrete situation cannot have a generic location or temporal extension. Axiom (d) summarizes that the meronymic relations (PARS, ELMT, HSIT) entail co-genericity. In general, if there is any interaction or physical coupling between two participants of a situation, then these participants must be co-generic. Still, there are a few more relations which may cross genericity levels, including the prominent cases SUB and SUBS. Axiom (e) rephrases the signature constraint expressed by the notation $\overline{o}$ and $\overline{si}$ for generics in Table 1.

Utilizing the Degree of Generality for NLP. The GENER feature has proven useful in NLP applications like question answering and automated knowledge acquisition: (a) Knowledge integration or 'assimilation' [19], in particular coreference resolution [4], can profit from the GENER markup, since co-referring mentions are always co-generic. This information eliminates ambiguities concerning the proper antecedent. (b) The MAVE answer validator uses GENER for avoiding wrong results of the textual entailment check. Basically, when the question requests information about a concrete object or event, as expressed by the constraint [GENER *sp*] on the queried variable, then the variable can only be bound to an entity in the supporting background knowledge which is also specific. In this way GENER improves validation results by eliminating a source of false positives.

In perspective, the GENER annotation makes it possible to automatically extract general knowledge from analyzed text, since it clearly separates specific and generic fractions of the represented document content. Rules for projecting GENER assignments, like those sketched in Table 5, are very important in this respect since they can infer unknown GENER values from GENER values of other conceptual entities and the information provided by the semantic representation of the surrounding text.

3.3 Extensionality Type

The MultiNet formalism allows a simplified extensional characterization of specific concepts with [GENER *sp*] by means of the ontological feature ETYPE (extensionality type), which serves to distinguish individuals from pluralities. For example, ⟨*the best player of the soccer world championship 2006*⟩ denotes an individual [ETYPE 0], ⟨*the German soccer team*⟩ denotes a collection of individuals [ETYPE 1], and ⟨*the teams at the soccer world championship 2006*⟩ denotes a group of groups [ETYPE 2]. Table 6 lists basic axioms related to the ETYPE. Axioms (b) and (d) rephrase the $\ddot{\sigma}$-notation for collections used in Table 1, while (a) and (c) specify the admissible arguments of ELMT and SUBM with more detail compared to the signature table. Thus, c can only be a subset of c' if both collections share the same ETYPE. Axiom (e) applies the ETYPE for describing the combinatory potential of situational concepts: The proposition "*x has died out*" can only be asserted of generic entities or collections.

Applications of the Extensionality Type. As shown by the latter example, the ETYPE can be used for a differentiated description of lexical entries, which in turn reduces ambiguities in syntactic and semantic analysis. In particular, knowing the ETYPE is important for PP interpretation. An example for that is given by the preposition "*in*", which can mean local containment ("*the ball in the bowl*") or membership in a collection ("*the politician in the party*"). The WOCADI parser which automatically generates MultiNet representations from German text uses corresponding PP interpretation rules involving ETYPE constraints [4]. The ETYPE is also used in disambiguation rules for coreference resolution and parse refinement rules which elaborate the result of syntactic-semantic analysis after finishing the main parse. Apart from these uses during parsing, the ETYPE is also available in the axiom syntax. As shown by Table 6, this is useful for integrating set-theoretic notions (like membership) into the KR formalism. Moreover the ETYPE can be used to restrict application of an axiom to individuals or to collections.

4 Classifying Relational Knowledge

So far we considered the classification of conceptual entities by sorts, relations, and ontological features. However, consider the sentences (a) "*The old man is wise*" and (b) "*The wise man is old.*" In both cases, the man of interest (say m) is described by the same elementary relationships SUB(m, *man*), PROP(m, *old*) and PROP(m, *wise*). To capture the difference between the sentences, we analyse the relationships as to their role for describing the involved arguments. A relationship is considered defining or 'categorical' for an argument node if it must necessarily be known in order to understand what the node stands for. In sentence (a), m is characterized as an old man, i.e. SUB(m, *man*) and PROP(m, *old*) are considered defining, while m being wise, which merely adds further knowledge about m, is said to be assertional or 'situational' knowledge. In (b), by contrast, m is described as a wise man, i.e. SUB(m, *man*) and PROP(m, *wise*) is defining while PROP(m, *old*) is assertional. The annotation by knowledge types 'c' (categorical) and 's' (situational) thus captures the difference between (a) and (b). The role of these KTYPEs for logical answer finding is explained in more detail in [20].

The KTYPE assignment is also needed for a proper treatment of propositional attitudes. In our example of an analyst's expectations on Google's stock price (see Fig. 2), the analyst's doubt becomes a modal operator which embeds the future increase in Google's stock price represented by node c_3. We may then ask which *proposition* the modal operator applies to. The point is that the sort hierarchy knows situational concepts, but does not avail us with (reifications of) propositions. The solution rests on the KTYPE annotation: A modally restricting edge like MCONT (labelled by 'r' in the figure) will always pick up the proposition formed by the literals in the *capsule* of the restricted node, which comprises all literals marked as categorical. The propositional content of the analyst's doubt is therefore given by SUBS(c_3, *increase*) $\wedge$ TEMP(c_3, *present.0*) $\wedge$ EXP(c_3, c_4), which is the proper description of the embedded situation c_3.

Utility of Knowledge Types for NLP. In coreference resolution, one must distinguish between known information about a mention (i.e. occurrence of an NP denoting an entity of discourse), which helps identifying the antecedent, and novel information about the mention, which corresponds to the assertion made by the new sentence. The following example will illustrate this distinction: "*Peter bought (a bungalow)*$_i$*. (The house)*$_i$ *is on the hill.*" Knowing that the mention "*the house*" in the second sentence refers to a house makes it possible to identify the proper antecedent i (i.e. the bungalow) assuming the background knowledge that $\mathrm{SUB}(\mathit{bungalow}, \mathit{house})$. This information is contained in the (defining) capsule of the referring node and thus marked as categorical. The new information (location of the house) makes an assertion about the known object (regardless of how it was identified). Thus being located at the hill is treated as assertional (outside capsule) and marked as situational, i.e. $[\mathrm{KTYPE}\, s]$.

The KTYPE is of special importance to NL generation since it separates the core characteristics of a concept (i.e. the interior of the capsule with $[\mathrm{KTYPE}\, c]$), and the contingent knowledge with $[\mathrm{KTYPE}\, s]$. Thus, Kintzel [21] uses the KTYPE annotation for content selection in an NL generation component for German. In advanced question answering (QA) systems which involve NL generation, we can distinguish a first phase of determining the 'answer core', i.e. the queried entity, and a second phase of determining the best way of describing the found entity. The second problem does not call for logic but rather for some form of retrieval which selects the relevant material. The QA system InSicht [5] exemplifies such a combination of semantic network matching (for identifying the conceptual entity of interest) and NL generation.

5 Conclusion

In this paper, we outlined an integrated approach to knowledge representation and natural language semantics suitable for question answering and similar NLP tasks. While incorporating advanced features like generic entities and pluralities, the approach tries to keep the balance between expressiveness and manageability.

References

1. van Benthem, J., ter Meulen, A. (eds.): Handbook of Logic and Language. North-Holland, Amsterdam (1997)
2. Helbig, H.: Knowledge Representation and the Semantics of Natural Language. Springer, Berlin (2006)
3. Hartrumpf, S., Helbig, H., Osswald, R.: The semantically based computer lexicon HaGenLex – Structure and technological environment. Traitement automatique des langues 44(2), 81–105 (2003)
4. Hartrumpf, S.: Hybrid Disambiguation in Natural Language Analysis. Der Andere Verlag, Osnabrück, Germany (2003)
5. Hartrumpf, S.: Question answering using sentence parsing and semantic network matching. In: Peters, C., Clough, P., Gonzalo, J., Jones, G.J.F., Kluck, M., Magnini, B. (eds.) CLEF 2004. LNCS, vol. 3491, pp. 512–521. Springer, Heidelberg (2005)
6. Glöckner, I., Hartrumpf, S., Leveling, J.: Logical validation, answer merging and witness selection – a case study in multi-stream question answering. In: Proc. of RIAO 2007, Pittsburgh, PA (2007)

7. Leveling, J.: Formale Interpretation von Nutzeranfragen für natürlichsprachliche Interfaces zu Informationsangeboten im Internet. Der andere Verlag, Tönning (2006)
8. Hartrumpf, S., Helbig, H., Leveling, J., Osswald, R.: An architecture for controlling simple language in web pages. eMinds: Int. J. on Human-Computer Interaction 1(2), 93–112 (2006)
9. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 166–181. Springer, Heidelberg (2002)
10. Baader, F., Horrocks, I., Sattler, U.: Description logics. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 3–28. Springer, Heidelberg (2004)
11. Brachman, R.: What IS-A is and isn't: An analysis of taxonomic links in semantic networks. IEEE Computer 16, 30–36 (1983)
12. Artale, A., Franconi, E., Guarino, N., Pazzi, L.: Part-whole relations in object-centered systems: An overview. Data & Knowledge Engineering 20(3), 347–383 (1996)
13. Hobbs, J.: Ontological promiscuity. In: Proc. 23rd Annual Meeting ACL, pp. 61–69 (1985)
14. de Marneffe, M.C., MacCartney, B., Grenager, T., Cer, D., Rafferty, A., Manning, C.D.: Learning to distinguish valid textual entailments. In: Proc. of the 2nd Pascal RTE Challenge Workshop (2006)
15. Hickl, A., Bensley, J.: A discourse commitment-based framework for recognizing textual entailment. In: Proc. Workshop on Textual Entailment and Paraphrasing, Prague, ACL, pp. 171–176 (2007)
16. Bobrow, D., Condoravdi, C., Crouch, R., de Paiva, V., Kaplan, R., Karttunen, L., King, T., Zaenen, A.: A basic logic for textual inference. In: Proceedings of the AAAI Workshop on Inference for Textual Question Answering, Pittsburgh, PA (2005)
17. Nairn, R., Condoravdi, C., Karttunen, L.: Computing relative polarity for textual inference. In: Proceedings of ICoS-5 (Inference in Computational Semantics), Buxton, UK (2006)
18. Klutsch, A.: Axiomatische Behandlung von Faktizität in MultiNet unter besonderer Berücksichtigung faktiver und implikativer Verben. Master's thesis, FernUniversität Hagen (2007)
19. Glöckner, I., Hartrumpf, S., Helbig, H.: Automatic knowledge acquisition by semantic analysis and assimilation of textual information. In: Proc. of KONVENS 2006, Konstanz, Germany, pp. 36–43 (2006)
20. Glöckner, I., Helbig, H.: Wissensrepräsentation und Antwortfindung mit Mehrschichtigen Erweiterten Semantischen Netzen. KI 2, 49–55 (2005)
21. Kintzel, F.: Entwurf und Implementierung einer natürlichsprachlichen Generierungskomponente für mehrschichtige erweiterte semantische Netze. Master's thesis, FernUniversität in Hagen (2007)

Deep Lexical Semantics

Jerry R. Hobbs

Information Sciences Institute
University of Southern California
Marina del Rey, California

Abstract. In the project we describe, we have taken a basic core of about 5000 synsets in WordNet that are the most frequently used, and we have categorized these into sixteen broad categories, including, for example, time, space, scalar notions, composite entities, and event structure. We have sketched out the structure of some of the underlying abstract core theories of commonsense knowledge, including those for the mentioned areas. These theories explicate the basic predicates in terms of which the most common word senses need to be defined or characterized. We are now encoding axioms that link the word senses to the core theories. This may be thought of as a kind of "advanced lexical decomposition", where the "primitives" into which words are "decomposed" are elements in coherently worked-out theories. In this paper we focus on our work on the 450 of these synsets that are concerned with events and their structure.

1 Introduction

Words describe the world, so if we are going to draw the appropriate inferences in understanding a text, we must have underlying theories of aspects of the world and we must have axioms that link these to words. This includes domain-dependent knowledge, of course, but 70-80% of the words in most texts, even technical texts, are words in ordinary English used with their ordinary meanings. For example, so far in this paragraph, only the words "theories" and "axioms" and possibly "domain-dependent" have been domain-dependent.

Domain-independent words have such wide utility because their basic meanings tend to be very abstract, and they acquire more specific meanings in combination with their context. Therefore, the underlying theories required for explicating the meanings of these words are going to be very abstract.

For example, a core theory of scales will provide axioms involving predicates such as *scale*, $<$, *subscale*, *top*, *bottom*, and *at*. These are abstract notions that apply to partial orderings as diverse as heights, money, and degrees of happiness. Then, at the "lexical periphery" we will be able to define the rather complex word "range" by the following axiom:

$$\begin{aligned}&(\forall\, x,y,z)range(x,y,z) \equiv\\ &\quad(\exists\, s,s_1,u_1,u_2)scale(s) \wedge subscale(s_1,s) \wedge bottom(y,s_1)\\ &\qquad\wedge top(z,s_1) \wedge u_1 \in x \wedge at(u_1,y) \wedge u_2 \in x \wedge at(u_2,z)\\ &\qquad\wedge (\forall\, u \in x)(\exists\, v \in s_1)at(u,v)\end{aligned}$$

That is, x ranges from y to z if and only if there is a scale s with a subscale s_1 whose bottom is y and whose top is z, such that some member u_1 of x is at y, some member u_2 of x is at z, and every member u of x is at some point v in s_1. Many things can be conceptualized as scales, and when this is done, a large vocabulary, including the word "range", becomes available. For example, we can now use and interpret "range" in the sentences

The grades on the midterm ranged from 33 to 96.
The timber wolf ranges from New Mexico to Alberta.
Pat's behavior ranges from barely tolerable to deeply hostile.

by instantiating *scale* in different ways.

It would be good if we could learn relevant lexical and world knowledge automatically, and there has been some excellent work in this area (e.g., [8]). For example, we can automatically learn the correlation between "married" and "divorced", and maybe we can even learn automatically the corresponding predicate-argument structures and which way the implication goes and with what temporal constraints. But this is a very simple relation to axiomatize in comparison to the "range" axiom. The kinds of knowledge we need are in general much more complex than automatic methods can give us. Moreover, automatic methods do not always yield very reliable results. The word "married" is highly correlated with "divorced" but it is also highly correlated with "murdered".

If we construct the core theories and the linking axioms manually, we can achieve the desired complexity and reliability. It would not be feasible to axiomatize the meanings of 100,000 words manually. But it *is* feasible to axiomatize the meanings of several thousand words manually, and if the words are very common, this would result in a very valuable resource for natural language understanding.

This paper describes an effort in which a set of very common words somehow related to events and their structure are being linked with underlying core theories that have been developed. Section 2 describes previous work in identifying a "core WordNet" and subsequent efforts to examine and classify the words in various ways. This led to the identification of 446 words with senses that are primarily focused on events, viewed abstractly. In Section 3 we describe several of the core theories that are crucial in characterizing event words, including composite entities, scales, change, and causality. In Section 4 we illustrate the work being carried out now on linking WordNet and FrameNet word senses to each other and linking these to the core theories.

This work can be seen as an attempt at a kind of deep lexical semantics. Not only are the words "decomposed" into what were once called primitives, but also the primitives are explicated in axiomatic theories, enabling one to reason deeply about the concepts conveyed by the text.

2 Identifying the Core Event Words

WordNet ([6]) contains tens of thousands of synsets referring to highly specific animals, plants, chemical compounds, French mathematicians, and so on. Most of these are rarely relevant to any particular natural language understanding application. To focus on the more central words in English, the Princeton WordNet group has compiled a CoreWordNet, consisting of 4,979 synsets that express frequent and salient concepts. These were selected as follows: First, a list with the most frequent strings from the British National Corpus was automatically compiled and all WordNet synsets for these strings were pulled out. Second, two raters determined which of the senses of these strings expressed "salient" concepts ([3]). CoreWordNet is downloadable from

http://wordnet.cs.princeton.edu/downloads.html.

Only nouns, verbs and adjectives were identified in this effort, but subsequently 322 adverbs were added to the list.

We classified these word senses manually into sixteen broad categories, listed here with rough descriptions and lists of sample words in the categories. Word senses are not indicated but should be obvious from the category.

Composite Entities: the structure and function of things made of other things: perfect, empty, relative, secondary, similar, odd, ...

Scales: partial orderings and their fine-grained structure: step, degree, level, intensify, high, major, considerable, ...

Events: concepts involving change and causality: constraint, secure, generate, fix, power, development, ...

Space: spatial properties and relations: grade, inside, lot, top, list, direction, turn, enlarge, long, ...

Time: temporal properties and relations: year, day, summer, recent, old, early, present, then, often, ...

Cognition: concepts involving mental and emotional states: imagination, horror, rely, remind, matter, estimate, idea, ...

Communication: concepts involving people communicating with each other: journal, poetry, announcement, gesture, charter, ...

Persons: concepts involving persons and their relationships and activities: leisure, childhood, glance, cousin, jump, ...

Microsocial: social phenomena other than communication that would be present in any society regardless of their level of technology: virtue, separate, friendly, married, company, name, ...

Bio: living things other than humans: breed, oak, shell, lion, eagle, shark, snail, fur, flock, ...

Geo: geographical, geological and meteorological concepts: storm, moon, pole, world, peak, site, sea, island, ...

Material World: other aspects of the natural world: smoke, shell, stick, carbon, blue, burn, dry, tough, ...

Artifacts: physical objects built by humans to fulfill some function: bell, button, van, shelf, machine, film, floor, glass, chair, ...

Food: concepts involving things that are eaten or drunk: cheese, potato, milk, bread, cake, meat, beer, bake, spoil, ...

Macrosocial: concepts that depend on a large-scale technological society: architecture, airport, headquarters, prosecution, ...
Economic: having to do with money and trade: import, money, policy, poverty, profit, venture, owe, ...

These categories of course have fuzzy boundaries and overlaps, but their purpose is only for grouping together concepts that need to be axiomatized together for coherent theories.

Each of these categories was then given a finer-grained structure. The internal structure of the category of event words is given below, with descriptions and examples of each subcategory.

- **State:** Having to do with an entity being in some state or not: have, remain, lack, still, ...
- **Change**: involving a change of state:
  - Abstractly: incident, happen
  - A change of real or metaphorical position: return, take, leave, rise, ...
  - A change in real or metaphorical size or quantity: increase, fall, ...
  - A change in property: change, become, transition, ...
  - A change in existence: develop, revival, decay, break, ...
  - A change in real or metaphorical possession: accumulation, fill, recovery, loss, give, ...
  - The beginning of a change: source, start, origin, ...
  - The end of a change: end, target, conclusion, stop, ...
  - Things happening in the middle of a change: path, variation, repetition, [take a] break, ...
  - Participant in a change: participant, player, ...
- **Cause:** having to do with something causing or not causing a change of state:
  - In general: effect, result, make, prevent, so, thereby, ...
  - Causes acting as a barrier: restriction, limit, restraint, ...
  - An absence of causes or barriers: chance, accident, freely, ...
  - Causing a change in position: put, pull, deliver, load, ...
  - Causing a change in existence: develop, create, establish, ...
  - Causing a change in real or metaphorical possession: obtain, deprive.
- **Instrumentality:** involving causal factors intermediate between the primary cause and the primary effect: way, method, ability, influence, preparation, help, somehow, ...
- **Process:** A complex of causally related changes of state:
  - The process as a whole: process, routine, work, operational, ...
  - The beginning of the process: undertake, activate, ready, ...
  - The end of the process: settlement, close, finish, ...
  - Things that happen in the middle of a process: trend, continuation, steady, postpone, drift, ...

- **Opposition:**
  - Involving factors acting against some causal flow: opposition, conflict, delay, block, bar, ...
  - Involving resistance to opposition: resist, endure, ...
- **Force:** Involving forces acting causally with greater or lesser intensity: power, strong, difficulty, throw, press, ...
- **Functionality:** A notion of functionality with respect to some human agent's goals is superimposed on the causal structure; some outcomes are good and some are bad:
  - Relative to achieving a goal: use, success, improve, safe, ...
  - Relative to failing to achieve a goal: failure, blow, disaster, critical, ...
  - Relative to countering the failure to achieve a goal: survivor, escape, fix, reform, ...

As with the broad categories, these subcategories are intended to group together words that need to be defined or characterized together if a coherent theory is to result.

3 Some Core Theories

The enterprise is to link words with core theories. The last section gave an indication of the words involved in the effort, and a high-level analysis of the concepts needed for defining or characterizing them formally. This section sketches some of the principal core theories, including concepts used in Section 4. Descriptions of all the core theories, with axioms, can be found at

http://www.isi.edu/∼hobbs/csk.html

Currently, there are sixteen theories defining or characterizing 230 predicates with 380 axioms. The theories differ from other commonsense knowledge bases, such as Cyc [4] or SUMO [7], primarily in the abstract character and linguistic motivation of the knowledge.

Eventualities and their Structure: Eventualities are possible or actual states or events. As axiomatized, they are isomorphic to predications, and just as predications have arguments, eventualities have participants. We can define a predicate *relatedTo* that holds between two entities x and y when they are participants in the same eventuality, or equivalently, when they are arguments of the same predication.

Set Theory: This is axiomatized in a standard fashion, and provides predicates like *setdiff* and *deleteElt*, the latter expressing a relation between a set and the set resulting from deleting an element from it.

Composite Entities: This is a very general theory of things made of other things, one of the most basic notions one can imagine. A composite entity is characterized by a set of components, a set of properties of these components, and a set of relations among the components and between the components and the whole. With this theory we can talk about the structure of an entity by explicating its components and their relations, and we can talk about the environment of an entity by viewing the environment as composite and having the entity among its components. The predicate *partOf* is a very broad notion covering among other relations the *componentOf* relation. We also introduce in this theory the figure-ground relation *at* which places an external entity "at" some component in a composite entity.

Scales: This theory was mentioned in the introduction. In addition to defining the basic vocabulary for talking about partial orderings, we also explicate monotone-increasing scale-to-scale functions ("the more ... the more ..."), the construction of composite scales, the characterization of qualitatively high and low regions of a scale (related to distributions and functionality), and constraints on vague scales based on associated subsets (e.g., if Pat has all the skills Chris has and then some, Pat is more skilled than Chris, even though such judgments in general are often indeterminate).

Change of State: The basic predicate for change of state is *change*. The expression $change(e_1, e_2)$ means that state e_1 changes into state e_2. The states must share a participant, and a state cannot change into the same state without going through an intermediate different state. The concept of *change* is linked with the theory of time in the obvious ways. We also define one-argument predicates $changeFrom$ and $changeTo$, suppressing one or the other argument of *change*. We define events as eventualities that involve a change of state and states (the predicate *state* used below) as eventualities that don't. The concept *remove* referred to in Section 4 can be defined in terms of *change* and *deleteElt* as a change of state from a dynamic set having one set of elements to its having a subset of those elements.

Cause: We characterize a causal complex for an effect by two strict properties: If every eventuality in a causal complex happens, the effect happens, and everything in the causal complex is relevent to the effect in a way that can be made precise. The predicate *cause* then captures a defeasible notion that isolates the most significant element in a causal complex, often the element that does not normally hold. Most of our causal knowledge and causal reasoning is in terms of *cause* rather then the idealized notion of causal complex. The concept *cause* has the expected properties, such as defeasible transitivity. In addition, in this theory we define such concepts as *enable*, *prevent*, *help*, and *obstruct*. There are also treatments of attempts, success, failure, ability, and difficulty.

Events: This theory is about how changes of state and causality compose into more complex events, processes and scenarios. It includes definitions of conditional, iterative, cyclic, and periodic events, and is linked with several well-developed ontologies for event structure, e.g., PSL ([2]).

The other core theories that have been constructed include a well-developed theory of time, a rather sparse theory of space, and a large number of theories explicating a commonsense theory of cognition.

4 Some Word Senses Linked to Core Theories

This section provides three examples of very basic words and how their various word senses in WordNet and in FrameNet ([1]) can be linked axiomatically to each other and to the core theories.

"Have": In WordNet the verb "have" has 19 senses. But they can be grouped into three broad "supersenses". In its first supersense, X has Y means that X is in some relation to Y. The WordNet senses this covers are as follows:

1. a broad sense, including have a son, having a condition hold and having a college degree
2. having a feature or property, i.e., the property holding of the entity
3. a sentient being having a feeling or internal property
4. a person owning a possession
7. have a person related in some way: have an assistant
9. have left: have three more chapters to write
12. have a disease: have influenza
17. have a score in a game: have three touchdowns

The supersense can be characterized by the axiom

$$have\text{-}s1(x, y) \supset relatedTo(x, y)$$

In these axioms, supersenses are indexed with s, WordNet senses with w, and FrameNet senses with f. Unindexed predicates are from core theories.

The individual senses are then specializations of the supersense where more domain-specific predicates are explicated in more specialized domains. For example, sense 4 relates to the supersense as follows:

$$have\text{-}w4(x, y) \equiv possess(x, y)$$
$$have\text{-}w4(x, y) \supset have\text{-}s1(x, y)$$

where the predicate *possess* would be explicated in a commonsense theory of economics, relating it to the priveleged use of the object. Similarly, $have\text{-}w3(x, y)$ links with the supersense but has the restrictions that x is sentent and that the "relatedTo" property is the predicate-argument relation between the feeling and its subject.

The second supersense of "have" is "come to be in a relation to". This is our *changeTo* predicate. Thus, the definition of this supersense is

$$have\text{-}s2(x, y) \equiv changeTo(e) \wedge have\text{-}s1'(e, x, y)$$

The WordNet senses this covers are as follows:

10. be confronted with: we have a fine mess
11. experience: the stocks had a fast run-up
14. receive something offered: have this present
15. come into possession of: he had a gift from her
16. undergo, e.g., an injury: he had his arm broken in the fight
18. have a baby

In these senses the new relation is initiated but the subject does not necessarily play a causal or agentive role. The particular change involved is specialized in the WordNet senses to a confronting, a receiving, a giving birth, and so on.

The third supersense of “have” is “cause to come to be in a relation to”. The axiom defining this is

$$have\text{-}s2(x,y) \equiv cause(x,e) \wedge have\text{-}s2'(e,x,y)$$

The WordNet senses this covers are

5. cause to move or be in a certain position or condition: have your car ready
6. consume: have a cup of coffee
8. organize: have a party
13. cause to do: she had him see a doctor
19. have sex with

In all these cases the subject initiates the change of state that occurs.

FrameNet has five simple transitive senses for “have”. Their associated frames are

1. Have associated
2. Possession
3. Ingestion
4. Inclusion
5. Birth

The first sense corresponds to the first WordNet supersense:

$$have\text{-}f1(x,y) \equiv have\text{-}s1(x,y)$$

The second sense is WordNet sense 4.

$$have\text{-}f2(x,y) \equiv have\text{-}w4(x,y)$$

The third sense is WordNet sense 6. The fourth sense is the *partOf* relation introduced in Section 3. It is a specialization of WordNet sense 2.

$$have\text{-}f4(x,y) \equiv partOf(x,y)$$
$$have\text{-}f4(x,y) \supset have\text{-}w2(x,y)$$

The fifth sense is WordNet sense 18.

By relating the senses in this way, an NLP system capable of inference can tap into both resources, for example, by accessing the WordNet hierarchy or the WordNet glosses expressed as logical axioms ([5]), and by accessing the FrameNet frames, which are very close to axiomatic characterizations of abstract situations. In addition, it allows us to access the core theories explicating predicates like *relatedTo* and *partOf*.

“Remove:” The most general meaning of “remove” is that if x removes y from z, then x causes there to be a change from a state in which y is at z. This is the first FrameNet sense.

$$remove\text{-}f1(x,y,z) \equiv cause(x,e1) \wedge changeFrom'(e1,e2) \wedge at'(e2,y,z)$$

In the second FrameNet sense, x is restricted to persons, y to clothes and z to the body.

There are eight WordNet senses, all specializations of the general FrameNet sense.

1. The "at" relation is physical location; i.e., z is restricted to physical objects.
2. The "at" relation is position in an organization.
3. y is dysfunctional, e.g., trash.
4. The "at" relation is membership in or being a subset of a set.
5. The change is functional or strategic.
6. $x = y$, e.g., "He removed himself from the contest."
7. The "at" relation is being alive; e.g., "The Mafia don removed his enemy."
8. y is abstract and dysfunctional, such as removing an obstacle.

WordNet sense 8 is a specialization of WordNet sense 4.

"Remain:" There are four WordNet senses of the verb "remain":

1. Not change out of a state: He remained calm.
2. Not change out of being at a location: He remained at his post.
3. Entities in a set remaining after others are removed: Three problems remain.
4. A condition remains in a location: Some smoke remained after the fire was put out.

The first sense is the most general and subsumes the other three. We can characterize it by the axiom

$$remain\text{-}w1(x, e) \supset arg(x, e) \wedge \neg changeFrom(e)$$

By the properties of *changeFrom* it follows that x is in state e. In the second sense, the property e of x is being in a location.

$$remain\text{-}w2(x, e) \equiv remain\text{-}w1(x, e) \wedge at'(e, x, y)$$

The fourth sense is a specialization of the second sense in which the entity x that remains is a state or condition.

$$remain\text{-}w4(x, e) \equiv remain\text{-}w2(x, e) \wedge state(x)$$

The third sense is the most interesting to characterize. There is a process that removes elements from a set, and what remains is the set difference between the original and the set of elements that are removed. In this axiom x remains after process e.

$$remain\text{-}w3(x, e) \equiv remove\text{-}w4'(e, y, s_2, s_1) \wedge setdiff(s_3, s_1, s_2) \wedge member(x, s_3)$$

That is, x remains after e if and only if e is a removal event (in sense 4 of WordNet) by some agent y of a subset s_2 from s_1, s_3 is the set difference between s_1 and s_2, and x is a member of s_3.

There are four FrameNet senses of "remain". The first is the same as WordNet sense 1. The second is the same as WordNet sense 3. The third and fourth are two specializations of WordNet sense 3, one in which the removal process is destructive and one in which it is not.

There are two nominalizations of the verb "remain"—"remainder" and "remains". All of their senses are related to WordNet sense 3. The first WordNet noun sense is the most general.

$$remainder\text{-}w1(x, e) \equiv remain\text{-}w3(x, e)$$

That is, x is the remaineder after process e if and only if x remains after e. The other three senses result from specialization of the removal process to arithmetic division, arithmetic subtraction, and the purposeful cutting of a piece of cloth. The nominalization "remains" is similar to "remainder", but the process e has to be some kind of consuming process.

5 Summary

We understand language so well because we know so much, and our computer programs will only approach what we might call "understanding" when they have access to very large knowledge bases. Much of this knowledge will be of a technical nature and can perhaps be acquired automatically by statistical methods or from learning by reading. But the bulk of the inferences required for understanding natural language discourse involve very basic abstract categories. In the work described here, we have identified the words which because of their frequency are most demanding of explication in terms of the inferences they trigger. We have constructed abstract core theories of the principal domains that need to be elaborated in order to express these inferences in a coherent fashion. We are in the process of defining or characterizing the meanings of a moderately large set of words related to the structure of events in terms of the core theories. In combination with other knowledge resources, this work should take us a step closer to sophisticated, inference-based natural language processing.

Acknowledgements

I have profited from discussions with Peter Clark, Christiane Fellbaum, and Rutu Mehta. This work was performed under the IARPA (DTO) AQUAINT program, contract N61339-06-C-0160.

References

1. Baker, C., Fillmore, C., Cronin, B.: The Structure of the Framenet Database. International Journal of Lexicography 16(3), 281–296 (2003)
2. Bock, C., Gruninger, M.: PSL: A Semantic Domain for Flow Models. Software and Systems Modeling Journal 4(2), 209–231 (2005)

3. Boyd-Graber, J., Fellbaum, C., Osherson, D., Schapire, R.: Adding dense, weighted, connections to WordNet. In: Proceedings of the Third Global WordNet Meeting, Jeju Island, Korea (January 2006)
4. Guha, R., Lenat, D.: CYC: A Midterm Report. AI Magazine 11(3), 33–59 (1990)
5. Harabagiu, S., Moldovan, D.: Enriching the WordNet Taxonomy with Contextual Knowledge Acquired from Text. In: Shapiro, S., Iwanska, L. (eds.) Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language, pp. 301–334. AAAI/MIT Press, Cambridge (2000)
6. Miller, G.: WordNet: a lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
7. Niles, I., Pease, A.: Toward a Standard Upper Ontology. In: Welty, C., Smith, B. (eds.) Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS 2001), Ogunquit, Maine (October 2001)
8. Pantel, P., Lin, D.: Discovering Word Senses from Text. In: Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD 2002), Edmonton, Canada, pp. 613–619 (2002)

On Ontology Based Abduction for Text Interpretation

Irma Sofia Espinosa Peraldi, Atila Kaya, Sylvia Melzer, and Ralf Möller

Hamburg University of Technology,
Institute for Software Systems,
Hamburg, Germany
sofia.espinosa@tuhh.de, at.kaya@tuhh.de,
sylvia.melzer@tuhh.de, r.f.moeller@tuhh.de

Abstract. Text interpretation can be considered as the process of extracting deep-level semantics from unstructured text documents. Deep-level semantics represent abstract index structures that enhance the precision and recall of information retrieval tasks. In this work we discuss the use of ontologies as valuable assets to support the extraction of deep-level semantics in the context of a generic architecture for text interpretation.

1 Introduction

The growing amount of unstructured electronic documents is a problem found in proprietary as well as in public repositories. In this context, the web is a representative example where the need of logic-based information retrieval (IR) to enhance precision and recall is evident. Logic-based IR means the retrieval of unstructured documents with the use of abstract terms that are not directly readable from the surface of the text, but only between its lines. For example, *Chocolate Cake Recipe* is an abstract term for the following text:

> *Yield: 10 Servings, 5 oz. semisweet chocolate (chopped), 3 oz. unsweetened chocolate (chopped), 1/4 lb. (8 Tbs.) unsalted butter, 1/4 cup all-purpose flour, 4 eggs at room temperature, ...*

Relational index structures are crucial for IR. Therefore, the task of defining the necessary index structures for abstract terms to allow logic-based IR is unavoidable. In our work, the necessary structures for logic-based IR are called *deep-level semantics* and the process of extracting deep-level semantics from unstructured text documents is understood as *text interpretation*. In the course of the work presented here, we will highlight that a feasible architecture (see Figure 1) to enable the automatic extraction of deep-level semantics from large-scale corpora can be achieved through:

– A two phase process of information extraction (IE), where the first phase exploits state-of-the-art shallow text processing mechanisms to extract surface-level structures as input for the second phase. The second phase called deep-level interpretation, exploits reasoning techniques over ontologies to extract deep-level semantics.

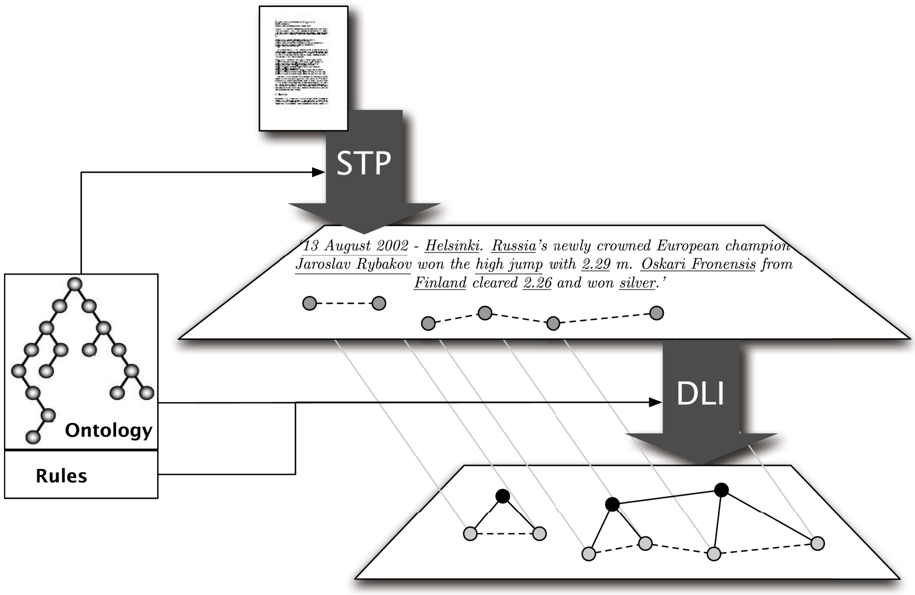

STP
'13 August 2002 - Helsinki. Russia's newly crowned European champion Jaroslav Rybakov won the high jump with 2.29 m. Oskari Fronensis from Finland cleared 2.26 and won silver.'
DLI
Ontology
Rules

Fig. 1. Ontology based text interpretation through the use of shallow text processing (STP) results as input for deep-level interpretation (DLI)

– The use of reasoning services, with abduction [1] as the key reasoning service for text interpretation.
– The use of ontologies, which provide the necessary index structures to represent surface-level, as well as deep-level semantics and to support logic-based IR.
– The use of ontologies as valuable and scalable assets, due to the use of Description Logics (DLs) [2] as their formal basis to support well studied reasoning tasks.
– The use of results from shallow text processing techniques, to extract surface-level semantics, are good enough to extract deep-level semantics and saves overhead to cope with large-scale corpora.

The previous hypotheses are the result of our work on the implementation and evaluation of a generic architecture for multimedia interpretation, which is described and evaluated here in the context of text interpretation. The approach followed for the design of this generic architecture is based on the combination of the works in [3], [4] and [5]. Different from [3] that employs processes for syntactic parsing, we argue that the results of shallow processing are good enough as input for reasoning techniques to extract deep-level semantics and to be able to deal with large-scale corpora. The evaluation results presented in this work confirm this hypothesis. In contrast to our work, in which we use DLs as knowledge representation formalism, [3] and [4] use first-order logics which have more expressive capabilities, but lack of automatic mechanism to prove for coherence. This is an important characteristic of DLs, therefore making this generic architecture a scalable one.

This work will focus on the most relevant part of the generic architecture, namely on the second phase of extraction where we show how the extraction of deep-level structures through abductive reasoning is useful for text interpretation. Section 2 introduces preliminaries for abduction as a core reasoning service for deep-level interpretation and presents its formalization in the context of DLs. Section 3 provides a real-world example to illustrate the interpretation process. Section 4 provides an empirical evaluation of the interpretation results generated by the architecture over a collection of web pages. Finally we conclude our work in Section 5

2 Abduction

Abduction is usually defined as a form of reasoning from effects to causes and aims at finding explanations (causes) for observations (effects). In this work text interpretation can be achieved through reasoning, more specifically through abduction. In general, abduction is formalized as

$$\Sigma \cup \Delta \models \Gamma \tag{1}$$

where background knowledge (Σ), and observations (Γ) are given and explanations (Δ) are to be computed.

DLs are a family of formal knowledge representation languages that support decidable reasoning problems [2]. If DLs are used as the underlying knowledge representation formalism, the background knowledge Σ is a knowledge base (KB) that consists of a Tbox $\mathcal{T}$ and an Abox $\mathcal{A}$: $\Sigma = (\mathcal{T}, \mathcal{A})$. In DL formalisms a KB consists of a Tbox that contains intentional knowledge in the form of a terminology and an Abox that contains the extensional (or assertional) knowledge that is specific to the individuals of the domain of discourse. Furthermore, Δ and Γ in Formula 1 are Aboxes and, therefore, they contain sets of role and concept instance assertions.

This work considers Abox abduction in DLs as the key inference service for text interpretation and the previous equation is modified to:

$$\Sigma \cup \Gamma_1 \cup \Delta \models \Gamma_2 \tag{2}$$

by splitting the assertions in Γ into two parts: bona fide assertions (Γ_1) and assertions requiring fiats (Γ_2). Bona fide assertions are assumed to be true by default, whereas fiat assertions are aimed to be explained.

In order to compute explanations, we use the implementation of Abox abduction as a non-standard retrieval inference service in DLs. Different from the standard retrieval inference services, answers to a given query cannot be found by simply exploiting the knowledge base. In fact, the abductive retrieval inference service has the task of acquiring what should be added to the knowledge base in order to positively answer a query.

To answer a given query, the abductive retrieval inference service can exploit non-recursive DL-safe rules with autoepistemic semantics [6] in a backward-chaining way. In this approach, rules are part of the knowledge base and are used to extend the expressivity of DLs. In order to extend expressivity and preserve decidability at the same time, the safety restriction is introduced for rules. Informally speaking, rules are DL-safe if they are only applied to Abox individuals, i.e., individuals explicitly named in the Abox [7]. In [8] a detailed discussion of the abductive retrieval inference service in DLs is presented.

The output of the abductive retrieval inference service should be a set Δs, which contains all explanations that are consistent w.r.t. Σ and Γ. However, in practice, one is not interested in retrieving every consistent explanation, but the most preferred explanation for every query. To achieve this goal, Δs is transformed into a poset according to a preference score. The preference score should reflect the two criteria proposed by Thagard for selecting explanations [9], namely simplicity and consilience. The less hypothesized assertions an explanation contains (simplicity) and the more fiat assertions (observations) an explanation involves (consilience), the higher its preference score should get. Therefore, the following formula to compute the preference score of each explanation is used: $S(\Delta) := S_f(\Delta) - S_h(\Delta)$. In this formula S_f is a term that reflects the involvement of fiat assertions in the explanation and S_h is a term that reflects the involvement of hypothesized assertions in the explanation. Thus, S_f and S_h can be defined as follows:

$$S_f(\Delta) := |\{i | i \in assertions(\Delta) \text{ and } i \in assertions(\Gamma_1)\}|$$
$$S_h(\Delta) := |\{i | i \in newAssertions(\Delta)\}|$$

The function *assertions* returns the set of all concept or role assertions found in a given Abox. The set *newAssertions* contains all concept or role assertions that are hypothesized during the generation of an explanation (hypothesized assertions).

3 An Example for Text Interpretation

In the context of a DL-based text interpretation architecture (like the one presented here) the results of shallow text processing can be represented as Aboxes. In order to extract deep-level semantics an Abox (hereafter called analysis Abox) is required as input. The input contains surface-level descriptions (see Figure 1) as results of shallow text processing. The interpretation process produces another Abox as output (interpretation Abox), which contains also deep-level besides surface-level semantic descriptions. The analysis Abox corresponds to Γ in the abduction formula (see Formula 1 in Section 2). The interpretation Abox is computed in an iterative process, and at the end of this process it contains the most preferred interpretation(s) of the text. The process starts with splitting

Γ into bona fide (Γ_1) and fiat assertions (Γ_2). Afterwards, the interpretation process proceeds with the following tasks in each iteration:

First, each fiat assertion in Γ_2 is transformed into a corresponding query and sent to the abductive retrieval inference service. The abductive retrieval inference service returns the most preferred interpretation to answer this query. Second, the explanation is added to Γ_1. Furthermore, the fiat assertion which has been used to constitute the query is removed from Γ_2 and added to Γ_1. Third, Γ_1 is checked to find out whether new information can be inferred through deduction. If such information can be found it is added to Γ_1 as well.

At the end of each iteration, the interpretation process analyzes Γ_1 to identify new fiat assertions and starts a new iteration. In particular, assertions that are added during the previous iteration step are identified as fiat assertions for the new iteration. If no new fiat assertions can be identified, the interpretation process returns Γ_1 as the interpretation Abox and terminates.

In the following we present a step by step interpretation of a text to discuss the details of the interpretation process. Figure 2 shows a text excerpt from a web page with athletics news. The underlined words in Figure 2 are keywords of this text, which have to be detected by shallow text processing. The results of the first phase in the architecture for the text in Figure 2 are represented in an Abox (analysis Abox), which is shown in Figure 3.

'<u>13 August 2002</u> - <u>Helsinki</u>. <u>Russia</u>'s newly crowned European champion <u>Jaroslav Rybakov</u> won the <u>high jump</u> with <u>2.29</u> m. <u>Oskari Fronensis</u> from <u>Finland</u> cleared <u>2.26</u> and won <u>silver</u>.'

Fig. 2. Sample text paragraph with underlined tokens from shallow text processing

To continue with the interpretation example, it is assumed that the ontology contains the axioms shown in Figure 4. In order to capture constraints among parts of aggregates, it is assumed that the ontology is extended with DL-safe rules (rules that are applied to Abox individuals only). In Figure 5 a set of rules for the athletics example is specified. Note that these rules define additional constraints on the concepts described in the ontology and, therefore, represent additional knowledge.

We assume that rules such as those shown in Figure 5 and the Tbox in Figure 4 constitute the background knowledge Σ. For the sake of brevity the Tbox and the set of rules show only a small excerpt of the athletics ontology, which is relevant for the text interpretation example discussed here.

To construct an interpretation for a text, explanations are searched to reveal why some words are related with some other words. Such explanations are then used to construct interpretation(s). Abox abduction (as presented in Section 2) is exploited to generate explanations and, therefore, constitutes the foundation of text interpretation in this architecture.

To start with the interpretation of the text paragraph in Figure 2, the shallow processing results for this text paragraph, namely the Abox in Figure 3, are

$$\begin{aligned}
date_1 &: Date \\
(date_1, \text{'13 August 2002'}) &: hasValue \\
country_1 &: Country \\
(country_1, \text{'Russia'}) &: hasValue \\
hjName_1 &: HighJumpName \\
(hjName_1, \text{'high jump'}) &: hasValue \\
perf_2 &: Performance \\
(perf_2, \text{'2.26'}) &: hasValue \\
country_2 &: Country \\
(country_2, \text{'Finland'}) &: hasValue \\
(hjName_1,\ date_1) &: sportsNameToDate \\
(hjName_1,\ city_1) &: sportsNameToCity \\
(pName_1,\ country_1) &: personNameToCountry \\
(pName_2,\ country_2) &: personNameToCountry \\
city_1 &: City \\
(city_1, \text{'Helsinki'}) &: hasValue \\
pName_1 &: PersonName \\
(pName_1, \text{'Jaroslav Rybakov'}) &: hasValue \\
perf_1 &: Performance \\
(perf_1, \text{'2.29'}) &: hasValue \\
pName_2 &: PersonName \\
(pName_2, \text{'Oskari Fronensis'}) &: hasValue \\
rank_1 &: Ranking \\
(rank_1,\ `silver') &: hasValue \\
(pName_1,\ perf_1) &: personNameToPerformance \\
(pName_2,\ perf_2) &: personNameToPerformance \\
(hjName_1,\ perf_1) &: sportsNameToPerformance
\end{aligned}$$

Fig. 3. Abox representing the results of shallow text processing

considered as Γ. The following Formula 2 Γ is divided into a part Γ_2 that the agent would like to have explained (fiat assertions), and a part Γ_1 that the interpretation agent takes for granted (bona fide assertions). In this example Γ_2 is:

$(hjName_1,\ date_1) : sportsNameToDate,$
$(pName_1,\ perf_1) : personNameToPerformance,$
$(hjName_1,\ city_1) : sportsNameToCity,$
$(pName_2, perf_2) : personNameToPerformance,$
$(pName_1,\ country_1) : personNameToCountry,$
$(hjName_1,\ perf_1) : sportsNameToPerformance,$
$(pName_2,\ country_2) : personNameToCountry.$

In the first step, these assertions are transformed into corresponding queries and the abductive retrieval inference service is asked for explanations. For example, from the role assertion $(hjName_1,\ date_1)$: $sportsNameToDate$ the following query is derived:

$Q := \{() \mid sportsNameToDate(hjName_1,\ date_1)\}$

$$
\begin{aligned}
Person &\sqsubseteq \exists hasName.PersonName \\
&\quad \sqcap \exists hasNationality.Country \\
Athlete &\sqsubseteq Person \\
HighJumper &\sqsubseteq Athlete \\
PoleVaulter &\sqsubseteq Athlete \\
HighJumpName &\sqsubseteq SportsName \sqcap \neg PoleVaultName \\
PoleVaultName &\sqsubseteq SportsName \\
SportsTrial &\sqsubseteq \exists hasParticipant.Athlete \\
&\quad \sqcap \exists hasPerformance.Performance \\
&\quad \sqcap \exists hasRanking.Ranking \\
HighJump &\sqsubseteq SportsTrial \sqcap \forall hasParticipant.HighJumper \\
&\quad \sqcap \neg PoleVault \\
PoleVault &\sqsubseteq SportsTrial \sqcap \forall hasParticipant.PoleVaulter \\
SportsRound &\sqsubseteq \exists hasName.RoundName \sqcap \exists hasDate.Date \\
&\quad \sqcap \exists hasPart.SportsTrial \\
HighJumpRound &\sqsubseteq SportsRound \sqcap \forall hasPart.HighJump \\
&\quad \sqcap \neg PoleVaultRound \\
PoleVaultRound &\sqsubseteq SportsRound \sqcap \forall hasPart.PoleVault \\
SportsCompetition &\sqsubseteq \exists hasPart.SportsRound \\
&\quad \sqcap \exists hasName.SportsName \\
&\quad \sqcap \exists takesPlace.City \\
HighJumpCompetition &\sqsubseteq SportsCompetition \\
&\quad \sqcap \forall hasPart.HighJumpRound \\
&\quad \sqcap \forall hasName.HighJumpName \\
&\quad \sqcap \neg PoleVaultCompetition \\
PoleVaultCompetition &\sqsubseteq SportsCompetition \\
&\quad \sqcap \forall hasPart.PoleVaultRound \\
&\quad \sqcap \forall hasName.PoleVaultName
\end{aligned}
$$

Fig. 4. A tiny example TBox Σ for the athletics domain

The abductive retrieval inference service has the task of computing what should be added to the KB in order to answer this query with true. In the given set of rules (see Figure 5), there are two rules that have the atom *sportsNameToDate* in the rule head (consequences). Both rules are applied in a backwardchaining way (i.e., from left to right) and corresponding terms are unified and variable bindings are obtained for X and Y. The unbound variable Z is instantiated with a new individual (e.g., new_ind_1). Note that for one of these rules, namely for the one that hypothesizes a pole vault competition, all bindings that are found for Y produce explanations that are inconsistent w.r.t. Σ. This is caused by the disjointness expressed in some of the concept description axioms in the TBox (e.g., the concepts *HighJumpName* and *PoleVaultName* are disjoint). The abductive retrieval service discards inconsistent explanations. Therefore, the explanation generated in order to answer Q with true is:

$$
\begin{aligned}
\Delta_1 = \{&new\_ind_1 : HighJumpCompetition, (new\_ind_1,\ date_1) : hasDate, \\
&(new\_ind_1,\ hjName_1) : hasSportsName\}
\end{aligned}
$$

$$\begin{aligned}
personNameToCountry\,(X,Y) \leftarrow\ & Person(Z),\\
& hasPersonName(Z,X),\\
& PersonName(X),\\
& hasNationality(Z,Y), Country(Y).\\
personToPerformance\,(X,Y) \leftarrow\ & Person(X),\\
& hasPersonName(X,Z),\\
& PersonName(Z),\\
& personNameToPerformance(Z,Y).\\
personToPerformance\,(X,Y) \leftarrow\ & SportsTrial(Z),\\
& hasParticipant(Z,X), Athlete(X),\\
& hasPerformance(Z,Y),\\
& Performance(Y).\\
sportsNameToCity\,(X,Y) \leftarrow\ & HighJumpCompetition(Z),\\
& hasSportsName(Z,X),\\
& HighJumpName(X),\\
& takesPlace(Z,Y), City(Y).\\
sportsNameToCity\,(X,Y) \leftarrow\ & PoleVaultCompetition(Z),\\
& hasSportsName(Z,X),\\
& PoleVaultName(X),\\
& takesPlace(Z,Y), City(Y).\\
sportsNameToDate\,(X,Y) \leftarrow\ & HighJumpCompetition(Z),\\
& hasSportsName(Z,X),\\
& HighJumpName(X),\\
& hasDate(Z,Y), Date(Y).\\
sportsNameToDate\,(X,Y) \leftarrow\ & PoleVaultCompetition(Z),\\
& hasSportsName(Z,X),\\
& PoleVaultName(X),\\
& hasDate(Z,Y), Date(Y).\\
sportsCompetitionToPerformance\,(X,Y) \leftarrow\ & SportsCompetition(X),\\
& hasSportsName(X,Z),\\
& SportsName(Z),\\
& sportsNameToPerformance(Z,Y).\\
sportsCompetitionToPerformance\,(X,Y) \leftarrow\ & HighJumpCompetition(X),\\
& hasPart(X,Z),\\
& HighJumpRound(Z),\\
& hasPart(Z,W), HighJump(W)\\
& hasPerformance(W,Y).\\
sportsCompetitionToPerformance\,(X,Y) \leftarrow\ & PoleVaultCompetition(X),\\
& hasPart(X,Z),\\
& PoleVaultRound(Z),\\
& hasPart(Z,W), PoleVault(W)\\
& hasPerformance(W,Y).
\end{aligned}$$

Fig. 5. Additional restrictions for text interpretation in the form of rules

The assertions shown in Δ_1 are added to Γ_1. Furthermore the assertion $(hjName_1,\ date_1) : sportsNameToDate$ is removed from Γ_2 and added to Γ_1. This procedure is applied to the remaining assertions in Γ_2 until Γ_2 is empty. At the end of the first interpretation step, Γ_1 contains (beside the assertions shown in Figure 3) the following newly created assertions:

$new\_ind_1 : HighJumpCompetition$, $(new\_ind_1,\ hjName_1) : hasSportsName$, $(new\_ind_1,\ date_1) : hasDate, (new\_ind_1,\ city_1) : takePlace, new\_ind_2 : Person$, $(new\_ind_2,\ pName_1) : hasPersonName, (new\_ind_2,\ country_1) : hasNationality$, $new\_ind_3 : Person, (new\_ind_3,\ pName_2) : hasPersonName$, $(new\_ind_3,\ country_2) : hasNationality$

Note that the preference score presented in Section 2 guarantees that explanations that involve less hypothesized individuals and more observations are preferred. This is why Γ_1 contains a single *HighJumpCompetition* instance at the end of the first interpretation step.

In the second step, the interpretation process applies the set of rules in a forward chaining way (from right to left) to check whether new information can be deduced. This yields the following assertions:

$(new\_ind_2,\ perf_1) : personToPerformance$,
$(new\_ind_3,\ perf_2) : personToPerformance$,
$(new\_ind_1,\ perf_1) : sportsCompetitionToPerformance$

which are also added to Γ_1. At this state, the interpretation process defines a new Γ_2 by selecting all newly inferred assertions as fiat assertions and starts a new iteration. The first interpretation step is applied to the assertions in the new Γ_2. At the end of this step, the following newly created assertions are added to Γ_1:

$new\_ind_4 : HighJumpRound, (new\_ind_1,\ new\_ind_4) : hasPart$,
$new\_ind_5 : HighJump, (new\_ind_4,\ new\_ind_5) : hasPart$,
$(new\_ind_5,\ perf_1) : hasPerformance, (new\_ind_5,\ new\_ind_2) : hasParticipant$,
$new\_ind_6 : SportsTrial, (new\_ind_6,\ new\_ind_3) : hasParticipant$,
$(new\_ind_6,\ perf_2) : hasPerformance$

In the second step of the second iteration no new information can be deduced by applying the set of rules in a forward chaining way. Therefore, the interpretation process terminates by returning the current Γ_1 as the interpretation Abox. Besides the assertions in Figure 3, the interpretation Abox contains also the following newly inferred assertions:

$new\_ind_1 : HighJumpCompetition, new\_ind_2 : Person, new\_ind_3 : Person$,
$new\_ind_4 : HighJumpRound, new\_ind_5 : HighJump, new\_ind_6 : SportsTrial$,
$(new\_ind_1,\ hjName_1) : hasSportsName, (new\_ind_1,\ date_1) : hasDate$,
$(new\_ind_1,\ city_1) : takePlace, (new\_ind_1,\ new\_ind_4) : hasPart$,
$(new\_ind_4,\ new\_ind_5) : hasPart, (new\_ind_5,\ perf_1) : hasPerformance$,
$(new\_ind_5,\ new\_ind_2) : hasParticipant$,
$(new\_ind_6,\ new\_ind_3) : hasParticipant, (new\_ind_6,\ perf_2) : hasPerformance$
$(new\_ind_2,\ pName_1) : hasPersonName, (new\_ind_2,\ country_1) : hasNationality$,
$(new\_ind_3,\ pName_2) : hasPersonName, (new\_ind_3,\ country_2) : hasNationality$

Note that in the interpretation Abox the person instance new_ind_2 participates in a high jump trial (new_ind_5) and, therefore, is also an instance of the concept $HighJumper$ (see the Tbox in Figure 4). Thus, information about abstract events, e.g. high jump trials, also influences information that is available about the related parts. With queries for $HighJumper$s the corresponding text would not have been found otherwise. Thus, recognizing abstract events means extracting deep-level semantics that without reasoning is not possible to obtain from the surface of the text.

4 Evaluation

In this section, the utility of the architecture is analyzed through an empirical evaluation of its results over a collection of web pages. For this purpose, the text interpretation architecture described in Section 1 was implemented. The core component of this implementation is the DL-reasoner RacerPro in version 1-9-2 [10] that supports various inference services. The abductive retrieval inference service, which is the key inference service for text interpretation, is integrated into the DL-reasoner. The architecture gets analysis Aboxes, exploits various inference services, and returns preferred interpretation Aboxes as deep-level semantic descriptions.

To test the implementation, an ontology about the athletics domain was used, and two different corpora of web pages containing daily news about athletics events. Furthermore, extractors that implement shallow text processing and machine learning techniques were trained in order to obtain concept instances as well as relations between the instances (see Figure 3). The training process was performed with the help of an annotation tool over the first corpus of web pages. The annotation process is two-fold, in the first step annotators manually associate words in the text, with corresponding concepts in the ontology. For this purpose, concepts such as the following have been annotated, i.e. *PersonName*, *Country*, *City*, *Age*, *Gender*, *Performance*, *Ranking*, *SportsName*, *RoundName*, *Date* and *EventName*. Second, the annotated concepts are filled into relational tables corresponding to *Athletes*, *SportTrials*, *Rounds* and *Events*, in order to train the extractors for the extraction of relations between concept instances. After finalizing the training process, the second corpus has been analyzed automatically to detect concept instances and relations between them, such that for each web page in the second corpus an analysis Abox with corresponding assertions has been generated without manual annotation effort.

As discussed in Section 3, given a set of fiat assertions (relations between concept instances), the interpretation process aims to extract deep-level semantic descriptions in the form of Abox structures. Therefore the criteria used for the evaluation is to prove that for every fiat assertion, the expected deep-level descriptions are generated. To set up the evaluation, a set of boolean queries is defined, such that for each fiat assertion a corresponding query is found in the set and is executed. If the query is answered with true, then the expected deep-level structures for the corresponding fiat assertions were correctly extracted. For example, given the fiat assertion $(pName_1,\ country_1) : personNameToCountry$ it is expected that index structures (deep-level) for a person with $pName_1$ as name and $country_1$ as nationality exists, therefore the following query is defined:

$$\begin{aligned} Q_1 := \{() \mid & Person(?x), hasPersonName(?x,\ pName_1), \\ & hasNationality(?x,\ country_1)\} \end{aligned}$$

For this evaluation a total of 85 web pages about athletics news were analyzed through shallow text processing. The results of shallow text processing was automatically analized to count the number of observations. According to the type of observation a set o queries was produced and executed in order to probe for the extraction of deep-level structures against the number of expected ones. The results of this evaluation can be observed in Figure 6.

Deep-level structures	**Expected**	**Extracted**
Person	48	48
SportsRound	10	10
Competition	99	99
SportEvent	189	189
SportsTrial with participant	326	326

Fig. 6. Results of deep-level interpretation

In this way the evaluation results, pointed out that all expected abstract concepts (explanations) and their relations were extracted as long as shallow text processing could deliver relations between surface-level instances and the neccesary rules for interpretation (abducibles) exist.

5 Conclusion

As observed in Section 3, ontologies are useful means to provide index structures in order to represent deep-level semantics. Furthermore, deep-level semantics is represented as relational structures representing the contents of the text, which can not be directly extracted from its surface (readable part of the text). The empirical evaluation of this work indicates that good ontology design is crucial for the architecture to extract the expected deep-level semantic structures. Thus, ontology designers should invest in producing a coherent ontology and a set of rules that define the space of abduceable predicates (explanations). While most of the work should be invested in the correct design of the ontology, in the long term, it means a good return of investment due to the support for ontology concistency check provided by existing well studied reasoning mechanisms. We believe that in the near future well designed ontologies will become a highly valuable asset to enhance the precision and recall of information retrieval. Furthermore, the use of ontologies provides a generic architecture to interpret different types of text. Ontologies can be tailored towards any domain of interest, depending on the targeted text documents. The results of shallow text processing techniques are good enough to extract deep-level semantics for dealing with large-scale corpora, however it is not discarded that results from other linguistic techniques, i.e. syntactic analysis, can improve the results of deep-level interpretation. Finally, as it can be observed in Figure 1, only the first phase of extraction is media (in this case text) dependent, while the second phase is media independent, therefore applicable for the interpretation of other types of multimedia, e.g. for images promising results were presented in [11].

References

1. Elsenbroich, C., Kutz, O., Sattler, U.: A Case for Abductive Reasoning over Ontologies. In: Proc. OWL-2006: OWL Experiences and Directions Workshop (2006)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, Cambridge (2003)
3. Hobbs, J.R., Stickel, M., Appelt, D., Martin, P.: Interpretation as abduction. Artificial Intelligence Journal 63 (1993)
4. Shanahan, M.: Perception as Abduction: Turning Sensor Data Into Meaningful Representation. Cognitive Science 29, 103–134 (2005)
5. Möller, R., Neumann, B.: On Scene Interpretation with Description Logics. In: Christensen, H.I., Nagel, H.-H. (eds.) Cognitive Vision Systems. LNCS, vol. 3948, pp. 247–278. Springer, Heidelberg (2006)
6. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Prentice-Hall, Englewood Cliffs (2003)
7. Möller, R., Neumann, B.: Ontology-based Reasoning Techniques for Multimedia Interpretation and Retrieval. In: Semantic Multimedia and Ontologies: Theory and Applications (to appear, 2007)
8. Espinosa, S., Kaya, A., Melzer, S., Möller, R., Wessel, M.: Multimedia Interpretation as Abduction. In: Proc. DL-2007: International Workshop on Description Logics (2007)
9. Thagard, R.P.: The best explanation: Criteria for theory choice. The Journal of Philosophy (1978)
10. Haarslev, V., Möller, R., Wessel, M.: RacerPro User's Guide and Reference Manual Version 1.9.2 (2007)
11. Espinosa, S., Kaya, A., Melzer, S., Möller, R., Wessel, M.: Towards a Media Interpretation Framework for the Semantic Web. In: Proc. WI-2007: The 2007 IEEE/WIC/ACM International Conference on Web Intelligence (2007)

Analysis of Joint Inference Strategies for the Semantic Role Labeling of Spanish and Catalan

Mihai Surdeanu[1], Roser Morante[2], and Lluís Màrquez[3]

[1] Barcelona Media Innovation Center
[2] Tilburg University
[3] Technical University of Catalonia

Abstract. This paper analyzes two joint inference approaches for semantic role labeling: re-ranking of candidate semantic frames generated by one local model and combination of two distinct models at argument-level using meta learning. We perform an empirical analysis on two recently released corpora of annotated semantic roles in Spanish and Catalan. This work yields several novel conclusions: (a) the proposed joint inference strategies yield good results even under adverse conditions: small training corpora, only two individual models available for combination, minimal output available from the individual models; (b) stacking of the two joint inference approaches is successful, which indicates that the two inference models provide complementary benefits. Our results are currently the best for the identification of semantic role for Spanish and Catalan.

1 Introduction

Semantic Role Labeling (SRL) is the task of analyzing clause predicates in open text by identifying arguments and tagging them with semantic labels indicating the role they play with respect to the verb, as in:

[Mr. Smith]$_{Agent}$ *sent* [the report]$_{Object}$ to [me]$_{Recipient}$ [this morning]$_{Temporal}$

Such sentence–level semantic analysis allows to determine "who" did "what" to "whom", "when" and "where", and, thus, characterize the participants and properties of the *events* established by the predicates. This semantic analysis in the form of event structures is very interesting for a broad spectrum of NLP applications.

The work proposed in this paper fits in the framework of supervised learning with joint inference for SRL. We introduce a stacking architecture that exploits several levels of global learning: in the first level we deploy two base SRL models that exploit only information local to each individual candidate argument; in the second level we perform re-ranking of the candidate frames generated by the base models; and lastly, we combine the outputs of the two individual models (after re-ranking) using meta-learning and sentence-level information.

The combination/joint inference models we introduce are not novel in themselves: all state-of-the-art SRL systems (see, e.g., [1,2,3,4]) include some kind of combination to increase robustness and to gain coverage and independence from parse errors. One may combine: 1) the output of several independent SRL basic systems [2,5], or 2) several outputs from the same SRL system obtained by changing input annotations or other internal parameters [4,3]. The combination can be as simple as selecting the best among the set of complete candidate solutions, but usually consists of combining fragments of alternative solutions to construct the final output. Finally, the combination component may or may not involve machine learning. So far, most of the SRL work has been performed on English, but recently, there have been remarkable efforts in other languages. The work by [6] studies semantic role labeling for Chinese, using the Chinese PropBank and NomBank corpora. Also, SemEval-2007 featured the first evaluation exercise of SRL systems for languages other than English, namely for Spanish and Catalan [7].

Nevertheless, our approach has several novel issues. First, we show that the global inference strategies analyzed perform well even under unfavorable training conditions: the training corpora are small and the global models have access to limited information (only two models available for combination, no output probabilities provided). We show that a crucial condition for the success of the joint inference models is the design of a feature set with low sparsity. We propose such feature sets for both the re-ranking and combination models and also show that some features previously proposed for English SRL –i.e., syntactic and lexical features extracted from the local models– are harmful in our setup. A second novelty of this work is the stacking architecture proposed: to our knowledge, this is the first work that provides empirical proof that stacking of several joint inference approaches is a successful strategy for SRL.

The paper is organized as follows. Section 2 overviews the proposed strategy. Section 3 describes the local SRL models used. Section 4 introduces the two joint inference approaches analyzed in this paper. We evaluate the whole stacking strategy in Section 5. Section 6 concludes the paper.

2 Approach Overview

The strategy introduced in this paper stacks two joint inference components on top of two individual SRL models. The intuition behind our approach is that we compensate for the small training corpus by taking advantage of information typically not available to the independent argument classifiers, i.e., global information available at frame and sentence level and redundancy between individual models. We detail the proposed approach in Figure 1.

The first layer in our system consists of two SRL models, Model 1 and Model 2. We call these models *local* because they classify each argument independently of the other arguments in the same frame or sentence. Each local model is followed by a re-ranking component, which re-scores candidate frames –i.e., complete sequences of arguments for one predicate– according to their properties. The re-ranking model performs *joint* or *global* inference because its re-ranking scores depend on joint properties of the set of arguments in one frame. We currently have implemented the re-ranking component for one local model (Model 1). Nevertheless, this is sufficient to prove one of our main claims, i.e., that stacking global models is a successful strategy even when only a small amount of training data is available. The post-processing steps implement various corrections of the local model outputs, e.g., here we implement a series of patterns to

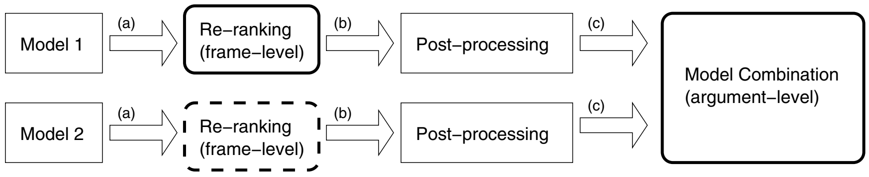


Fig. 1. Overview of the stacking architecture. The rounded boxes indicate joint inference components. The interrupted lines indicate components currently not implemented.

capture locative and temporal modifier arguments that are missed by the local classifiers. In our architecture, the re-ranking component is placed before post-processing because the re-ranking classifier requires the output probabilities generated by the local models and these are no longer consistent after the post-processing corrections. The proposed system pipeline concludes with another global model, which combines the outputs of the two branches. The combination model merges all the arguments generated for one sentence into one pool and re-scores them exploiting the redundancy of the two models and global information from the corresponding sentence.

3 Local Models

3.1 Model 1

Model 1 is an adaption of a SRL system we developed previously for English (third local model in [2]). This SRL approach maps each frame argument to one syntactic constituent and trains one-vs-all AdaBoost [8] classifiers to jointly identify and classify constituents in the full syntactic tree of the sentence as arguments. Similarly to other state-of-the-art SRL systems, our model extracts features from: (a) the argument constituent, (b) the target predicate, and (c) the relation between the predicate and argument syntactic constituents [9,10,11,12,3]. Features range from lexical –e.g., head words of the argument and predicate constituents– to syntactic –e.g., constituent labels and syntactic path between the predicate and argument constituents.

The model was adapted to the Spanish and Catalan corpora by removing the features that were specific either to English or PropBank and adding several new features:

- We removed the *governing category* feature [9] because it does not apply to the Spanish and Catalan corpora: in PropBank, agents are typically dominated by a `S` (sentence) phrase, whereas patients are attached to `VP` (verb) phrase. In our corpora both arguments are dominated by the `S` phrase that includes the predicate.
- We removed *temporal cue words* features [13] because they were based on an English-specific dictionary.
- We removed all features based on named-entity information [10] because we did not have a named-entity recognizer for the target languages.
- We implemented head phrase selection heuristics for Spanish and Catalan.

- We added *syntactic function* features. The syntactic functions available in the data often point to specific argument labels (e.g., the function `SUJ` usually indicates an `Arg0` argument).
- Because the set of part-of-speech (POS) tags and syntactic labels in Spanish and Catalan is much richer than the English Treebank set, we added *back-off* features, where the syntactic labels and POS tags are reduced to a simpler, Treebank-like set.

All the other features are similar to the English SRL system described in [2]. In addition to feature changes we implemented a novel candidate filtering heuristic to reduce the model search space: we select as candidates only syntactic constituents that are *immediate* descendants of *all* `S` phrases that include the corresponding predicate. For both languages, over 99.6% of the candidates match this constraint. An additional filtering constraint implemented is that the model inspects only candidate labels allowed for the given predicate.[1] To enforce the domain constraints, i.e., no overlap allowed between arguments of the same predicate and numbered arguments (`Arg0` to `Arg5`) can not repeat for the same predicate, we use a dynamic programming algorithm similar to the one proposed by Toutanova et al. [3].

The post-processing process in Model 1 recovers some temporal and locative modifier arguments missed during classification. The need for this component stemmed from the observation that in our initial experiments Model 1 performed poorly for the recognition of these two argument types on out-of-domain data. For example, simple constituents such as prepositional phrases starting with the preposition *durante* (*during*) were not recognized as temporal arguments even though samples existed in the training data. This happens because Model 1 focuses on other non-lexicalized features that appear to be more regular in the small training data available. To recover from this problem, we acquired a series of lexicalized patterns that can classify these modifier arguments with a precision higher than 60% in the training data.[2] The patterns have the following forms: (a) head word of candidate constituent, or (b) head word of constituent concatenated with the head word or POS tag of the first noun phrase in the constituent if the constituent is a prepositional phrase. During post-processing, these patterns are used to label constituents with temporal and locative argument labels only if they were not classified in any other class by the Model 1 classifiers.

3.2 Model 2

Unlike Model 1, Model 2 was developed from scratch for the two target languages. This model is an enhanced version of an earlier system [14] developed for the SemEval–2007 task *Multilevel Semantic Annotation of Catalan and Spanish* [15].

Candidate filtering in Model 2 has two significant differences from the strategy used in Model 1. First, Model 2 inspects only immediate descendants of the *most specific* `S` phrase that includes the target predicate. Second, the model skips constituents with the

[1] A verbal lexicon with this information is distributed with the corpora.

[2] This introduces some overfitting because we "know" that Model 1 performs poorly for these arguments on out-of-domain data. However, the overfitting is minimal because patterns are learned strictly from the training data. Using the post-processing component is important because it shows that local models can be successfully interleaved with global models.

syntactic functions AO, ET, MOD, NEG, IMPERS, PASS, and VOC, as these never carry a semantic role in the training corpora. During training, these constituents are assigned an additional semantic label, NONE, which is not reported in the output. The intuition behind Model 2's candidate filtering heuristic is that, by filtering out constituents more aggressively than Model 1, this approach may miss some valid argument constituents but it generates a cleaner set of candidates with fewer negative samples.

For classification, Model 2 uses memory-based learning, specifically the IB1 algorithm as implemented in TiMBL[3]. IB1 is a supervised inductive algorithm for learning classification tasks based on the k-nearest neighbor classification rule. The algorithm is parametrized by using Jeffrey Divergence as the similarity metric, gain ratio for feature weighting, using 11 k-nearest neighbors, and weighting the class vote of neighbors as a function of their inverse linear distance. Similarly to Model 1, Model 2 uses features extracted from the argument and predicate constituents and the relation between the two phrases. Additionally, Model 2 extracts a series of features from the entire clause that includes the predicate in focus: total number of predicate siblings with function CC; relative positions of siblings with functions SUJ, CAG, CD, CI, ATR, CPRED, and CREG in relation to the verb; boolean feature that indicates if the clause contains a verbal *se*; and total number of predicate siblings in the clause. Model 2's complete feature set is detailed in [14].

Motivated by the observation that not all features have the same relevance, i.e., the most informative features for SRL in Spanish and Catalan are the features that provide information about the syntactic constituent of the candidate argument [16], Model 2 is the result of a feature selection process. Feature selection was performed by starting with a set of basic features (the head words with their POS tags, in their local context) and gradually adding new features. Every new feature added to the basic system was evaluated in terms of average accuracy in a 10-fold cross-validation experiment; if it improved the performance on held-out data, it was added to the selection.

Post-processing in Model 2 is a minimal process: it removes arguments tagged with the NONE label and constructs the required bracketing structure for the output. Because Model 2 extracts its candidate arguments only from sibling constituents we do not need to enforce the non-overlapping constraint. Enforcing the non-repetition constraint for numbered arguments did not improve results, so we did not include it in the final system.

3.3 Differences between the Two Local Models

While both local models follow the same SRL approach, i.e., mapping each argument to one syntactic constituent and learning classifiers to assign valid argument labels to candidate constituents, there are significant differences between them: (a) The filtering strategy for candidate arguments is different: Model 2 uses an approach that generates fewer candidate arguments than Model 1. (b) Model 2 uses a richer feature set, e.g., it has features that exploit syntactic information from the whole clause that includes the predicate. Model 1 does not exploit clause-level information. (c) Model 2 performs feature selection whereas Model 1 does not. (d) The learning paradigm is different: AdaBoost versus memory-based learning. (e) Model 1 uses a series of post-processing

[3] http://ilk.uvt.nl/timbl/

Algorithm 1. Re-ranking Perceptron

```
$\mathbf{w} = \mathbf{0}$
for $i = 1$ to $n$ do
    for $j = 2$ to $n_i$ do
        if $\mathbf{w} \cdot \mathbf{h}(\mathbf{x_{ij}}) > \mathbf{w} \cdot \mathbf{h}(\mathbf{x_{i1}}) - \tau$ then
            $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{h}(\mathbf{x_{i1}}) - \mathbf{h}(\mathbf{x_{ij}})$
```

patterns to recover some modifier arguments that are not captured during classification. These differences ensure that there is sufficient variance between the two local models, a crucial condition for the success of the combination model.

4 Global Models

4.1 The Re-ranking Model

We base our re-ranking approach on a variant of the re-ranking Perceptron of Collins and Duffy [17]. We modify the original algorithm in two ways to make it more robust to the small training set available: (a) instead of comparing the score of the correct frame only with that of the frame predicted by the current model, we sequentially compare it with the score of *each* candidate frame, and (b) we learn not only when the prediction is incorrect but also when the prediction is not confident enough. Both these changes allow the algorithm to acquire more information about the problem to be learned, an important advantage when the training data is scarce.

The algorithm is listed in Algorithm 1: $\mathbf{w}$ is the vector of model parameters, $\mathbf{h}$ generates the feature vector for one example, and $\mathbf{x_{ij}}$ denotes the jth candidate for the ith frame in the training data. $\mathbf{x_{i1}}$, which denotes the "correct" candidate for frame i, is selected in training to maximize the F_1 score for each frame. The algorithm sequentially inspects all candidates for each frame and learns when the difference between the scores of the correct and the current candidate is less than a threshold τ. During testing we use the average of all acquired model vectors, weighted by the number of iterations they survived in training. We tuned all system parameters through cross-validation on the training data. For both languages we set $\tau = 10$ (we do not normalize feature vectors) and the number of training epochs to 2.

With respect to the features used, we focus only on global features that can be extracted independently of the local models. We show in Section 5 that this approach performs better on the small corpora available than approaches that include features from the local models, which are too sparse when the learning sample is an entire frame. We group the features into two sets: (a) features that extract information from the whole candidate set, and (b) features that model the structure of each candidate frame:

Features from the whole candidate set:
(1) Position of the current candidate in the whole set. Frame candidates consistent with the domain constraints are generated using a dynamic programming algorithm [3], and then sorted in descending order of the log probability of the whole frame (i.e., the sum of all argument log probabilities as reported by the local model). Hence, smaller positions

indicate candidates that the local model considers better.
(2) For each argument in the current frame, we store its number of repetitions in the whole candidate set. The intuition is that an argument that appears in many candidate frames is most likely correct.

Features from each candidate frame:
(3) The complete sequence of argument labels, extended with the predicate lemma and voice, similar to Toutanova et al. [3].
(4) Maximal overlap with a frame from the verb lexicon. Both the Spanish and Catalan TreeBanks contain a lexicon that lists the accepted sequences of arguments for the most common verbs. For each candidate frame, we measure the maximal overlap with the lexicon frames for the given verb and use the precision, recall, and F_1 scores as features.
(5) Average probability (from the local model) of all arguments in the current frame.
(6) For each argument label that repeats in the current frame, we add combinations of the predicate lemma, voice, argument label, and the number of label repetitions as features. The intuition is that argument repetitions typically indicate an error (even if allowed by the domain constraints).

4.2 The Combination Model

The combination model is an adaptation of a global model we previously introduced for English [2]. The approach starts by merging the solutions generated by the two local models into a unique pool of candidate arguments, which are then re-scored using global information fed to a set of binary discriminative classifiers (one for each argument label). The classifiers assign to each argument a score measuring the confidence that the argument is part of the correct solution. Finally, the re-scored arguments for one sentence are merged into the best solution –i.e., the argument set with the highest combined score– that is consistent with the domain constraints. We implemented the discriminative classifiers using Support Vector Machines[4] configured with linear kernels with the default parameters. We implemented the solution generation stage with a CKY-based dynamic programming algorithm [18].

We group the features used by the re-scoring classifiers into four sets:

FS1. Voting features – these features quantify the votes received by each argument from the two local models. This feature set includes: (a) the *label* of the candidate argument; (b) the *number of systems* that generated an argument with this label and span; (c) the *unique ids* (Model 1 or Model 2) of the models that generated an argument with this label and span; and (d) the *argument sequence* of the whole frame for the models that generated this argument candidate.

FS2. Overlap features (same predicate) – these features measure the overlap between different arguments produced by the two models for the same predicate: (a) the *number* and *unique ids* of the models that generated an argument with the same span but different label; (b) the *number* and *unique ids* of the models that generated an argument that is included, or contains, or overlaps the candidate argument in focus.

[4] `http://svmlight.joachims.org`

FS3. Overlap features (other predicates) – these features are similar with the previous group, with the difference that we now compare arguments generated for *different* predicates. The motivation for the overlap features is that no overlap is allowed between arguments attached to the same predicate, and only inclusion or containment is permitted between arguments assigned to different predicates.
FS4. Features from the local models – we replicate the features from the local models that were shown to be the most effective for the SRL problem: information about the syntactic phrase and head word of the argument constituent (and left-most included noun phrase if the constituent is a prepositional phrase) and the syntactic path between the argument and predicate constituents. The motivation for these features is to learn the syntactic and lexical preferences of the individual models.

5 Experimental Results

For all the experiments, we used the corpora provided by the SRL task for Catalan (ca) and Spanish (es) at SemEval-2007 [7]. This is a part of the CESS-ECE corpus consisting of about 100K words per language, annotated with full parsing, syntactic functions, and semantic roles, and also including named entities and noun senses. The source of the corpus is varied including articles from news agencies, newspapers, and balanced corpora of the languages involved. These corpora are split into training (90%) and test (10%) subsets. Each test set is also divided into two subsets: ‘in-domain’ (marked with the .in suffix) and ‘out-of-domain’ (.out) test corpora. The first is intended to be homogeneous with respect to the training corpus and the second is extracted from a part of the CESS-ECE corpus not involved in the development of the resources.

Although the task at SemEval-2007 was to predict three layers of information, namely, semantic roles, named entities and noun senses, from the gold standard parse trees, we only address the SRL subtask in this work. It is worth noting that the role set used contains labels that are composed by a numbered argument (similar to PropBank) plus a verb-independent thematic role label similar to the scheme proposed in VerbNet.

The data for training the global models was generated by performing 5-fold cross validation on the whole training set with the previous models in the pipeline of processors (i.e., the individual models after post-processing). Also, parameter tuning was always performed by cross validation on the training set.

5.1 Overall Results

For the clarity of the exposition we present a top-down analysis of the proposed approach: we discuss first the results of the overall system and then we analyze the two global models in the system pipeline in the next two sub-sections.

We list the results of the complete system in the “Combination” rows in Table 1, for the four test corpora (Spanish and Catalan, in and out of domain). In this table we report results using the best configurations of the global models (see the next sub-sections for details). Next to the F_1 scores we list the corresponding statistical significance intervals obtained using bootstrap re-sampling [19].[5] The F_1 scores range from 83.56 (ca.out) to

[5] F_1 rates outside of these intervals are assumed to be significantly different from the related F_1 rate ($p < 0.05$).

Table 1. Overall results for the combination model. The individual models are evaluated after re-ranking and post-processing (where applicable), i.e., position (c) in Figure 1.

	ca.in			ca.out		
	P	R	F_1	P	R	F_1
Combination	92.16	85.83	**88.88**±1.80	87.80	79.72	**83.56**±2.33
Model 1 (c)	87.83	83.61	87.22±2.10	82.83	79.25	81.00±2.71
Model 2 (c)	87.59	86.52	87.05±2.17	82.22	77.28	79.67±2.62
Baseline R	87.67	**87.22**	87.45±2.19	82.18	**82.25**	82.21±2.47
Baseline P	**94.30**	80.52	86.87±2.03	**92.11**	69.01	78.90±2.71
	es.in			es.out		
	P	R	F_1	P	R	F_1
Combination	89.22	81.09	**84.96**±1.80	89.75	83.46	**86.49**±1.79
Model 1 (c)	83.06	81.47	82.26±1.94	87.68	84.44	86.03±2.07
Model 2 (c)	83.42	82.47	82.94±2.10	85.41	85.41	85.41±1.89
Baseline R	82.88	**83.38**	83.13±2.02	84.92	**85.99**	85.45±1.78
Baseline P	**92.32**	73.66	81.94±2.15	**95.35**	77.82	85.70±1.79

88.88 (ca.in). These results are comparable to the best SRL systems for English, where the performance using correct syntactic information approaches 90 F_1 points for in-domain evaluation. We consider these numbers encouraging considering that our training corpora is 10 times smaller than the English PropBank and we have to label a larger number of classes (e.g., there are 33 core arguments for Spanish vs. 6 for English).

5.2 Analysis of the Combination Model

Table 1 details the contribution of the combination model, i.e., the right-most box in Figure 1. We compare the combination model against its two inputs, i.e., Models 1 and 2 after re-ranking and post-processing (position (c) in Figure 1), and against two baselines, one recall-oriented ("Baseline R") and one precision-oriented ("Baseline P"). Baseline R merges *all* the arguments generated by Models 1 and 2 (c). Baseline P selects only arguments where the two input models agreed.[6]

The combination model is better than its two inputs in all the setups. The increase in F_1 scores ranges from 0.46 points (es.out) to 2.56 points (ca.out). For in-domain data (ca.in and es.in), the F_1 score improvement is approximately 2 points, which is similar to the improvements seen for English, even though here we have less training data and fewer individual models that provide less information (e.g., no output probabilities are available). The performance of the combination model is always better than both of the baselines as well. As expected, the recall-oriented baseline achieves the highest recall and the precision-oriented baseline the highest precision, but the combination model obtains the best F_1 score. This is an indication that the model is capable of learning useful information beyond the simple redundancy used by the baselines.

Table 2 analyzes the contribution of the four proposed features groups. The analysis is cumulative, i.e., the "+ FS2" row lists the performance of the system configured

[6] Conflicts with the domain constraints are solved using the same strategy as [2].

Table 2. Feature analysis for the combination model

	ca.in			ca.out			es.in			es.out		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
FS1	92.28	84.17	88.04	88.54	71.83	79.32	**89.97**	79.71	84.53	90.62	81.81	85.99
+ FS2	92.22	85.57	88.77	87.76	79.44	83.39	88.64	**81.85**	**85.11**	89.54	**84.92**	**87.17**
+ FS3	92.16	**85.83**	**88.88**	87.80	**79.72**	**83.56**	89.22	81.09	84.96	89.75	83.46	86.49
+ FS4	**92.58**	84.61	88.41	**87.98**	77.00	82.12	89.73	79.63	84.38	**89.85**	81.81	85.64

Table 3. Analysis of the re-ranking model

	ca.in			ca.out			es.in			es.out		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Model 1 (a)	85.48	84.43	84.95	80.73	68.45	74.09	78.73	77.11	77.91	84.73	73.93	78.96
This paper	87.83	86.61	87.22	**82.17**	**69.67**	**75.41**	**83.06**	**81.47**	**82.26**	**86.63**	**76.26**	**81.12**
Collins	**87.92**	**86.70**	**87.30**	81.19	68.92	74.56	82.67	81.09	81.87	85.62	75.88	80.45
Toutanova	79.40	78.43	78.92	73.00	62.44	67.31	79.32	76.95	78.12	82.52	75.29	78.74

with the first two feature groups. The table indicates that in our setup the features from the local models (FS4) do not help. This is a significant difference from English SRL, where lexical and syntactic features extracted from the local models are known to help global strategies [2,3]. Our conjecture is that in our setup the combination model can not recover from the increased sparsity introduced by the features that model syntactic context and lexical information. Note that the sparsity of these features is much larger in the combination model than the local models because at this level we work only with the final output of the local models, whereas the individual models have a much larger space of candidate arguments. A somewhat similar observation can be made for FS3. But because we performed the tuning of the combination model on training data and there we saw a small improvement when using this feature group we decided to include this set of features in the best model configuration.

5.3 Analysis of Re-ranking

We analyze the proposed re-ranking model in Table 3. We compare the re-ranking performance against the corresponding local model (Model 1 (a)) and against two variations of our approach: in the first we used our best feature set but the original re-ranking Perceptron of Collins and Duffy [17], and in the second we used our re-ranking algorithm but we configured it with the features proposed by Toutanova et al. [3]. This feature set includes features (3) and (6) from Section 4.1 and all features from the local model concatenated with the label of the corresponding candidate argument.

We draw several observations from this analysis: (a) our re-ranking model always outperforms the local model, with F_1 score improvements ranging from 1.32 to 4.35 points; (b) the re-ranking Perceptron proposed here performs better than the algorithm of Collins and Duffy on three out of four corpora, and (c) the feature set proposed here achieve significant better performance on the SemEval corpora than the set proposed by

Table 4. Stacking results relative to Model 1

	ca.in			ca.out		
	P	R	F_1	P	R	F_1
Model 1 (a)	85.48	84.43	84.95±2.27	80.73	68.45	74.09±3.18
+ re-ranking	87.83	**86.61**	87.22±2.00	82.17	69.67	75.41±2.95
+ post-processing	–	–	–	82.83	79.25	81.00±3.16
+ combination	**92.16**	85.83	**88.88**±1.80	**87.80**	**79.72**	**83.56**±2.33

	es.in			es.out		
	P	R	F_1	P	R	F_1
Model 1 (a)	78.73	77.11	77.91±2.27	84.73	73.93	78.96±2.50
+ re-ranking	83.06	**81.47**	82.26±2.04	86.63	76.26	81.12±2.21
+ post-processing	–	–	–	87.68	84.44	86.03±2.23
+ combination	**89.22**	81.09	**84.96**±1.80	**89.75**	**83.46**	**86.49**±1.79

Toutanova et al., which never improves over the local model. These observations indicate that, while our modifications to the re-ranking Perceptron yield a minor improvement, the biggest contribution comes from the novel set of features proposed. Whereas the model configured with the Toutanova et al. feature set performs modestly because the features from the local models are too sparse in this global setup, we replicate the behavior of the local model just with feature (1), and all the other five global features proposed have a positive contribution. This conclusion correlates well with the analysis in the previous sub-section, where we also observed that syntactic and lexical features from the local model do not help in another global setup with small training corpora.

5.4 Putting It All Together: Analysis of Stacking

Table 4 summarizes the paper's main results. We show the results at every point in the pipeline for Model 1: after the local model, after re-ranking, after post-processing, and after the combination with Model 2. Note that we apply the post-processing patterns only on out-of-domain data because this is where we observed that the local model fails to recognize locative and temporal modifier arguments. The table re-enforces our claim that the stacking of global strategies is a successful way to mitigate the lack of training data, even (or more so) when the global models are interleaved with local strategies. On in-domain corpora (ca.in and es.in) we improve the performance of the local model with 3.93 and 7.05 F_1 points. On out-of-domain corpora (ca.out and es.out), where we applied the post-processing patterns, we increased the F_1 score of the local model with 9.47 and 7.53 points.

Another important observation is that the two global approaches can be stacked because they provide complementary benefits. Because re-ranking is configured to optimize F_1 it tends to improve recall, which is generally lower in the local model due to the insufficient coverage of the training data. On the other hand, the combination model tends to improve precision because a good part of its learning is driven by the redundancy between the two models, which is a precision-oriented feature.

6 Conclusions

In this paper we propose a SRL approach that stacks two joint (or global) inference components on top of two individual (or local) SRL models. The first global model re-ranks entire candidate frames produced by the local models. The second joint inference model combines the outputs of the two local models after re-ranking using meta-learning and sentence-level information.

We draw several novel conclusions from this work. First, we show that global strategies work well under unfavorable training conditions, e.g., our training corpora are 10 times smaller than the English PropBank, there are only two local models available for combination, and these models provide limited information (no output probabilities). We show that a key requirement for success in these conditions is to focus on global mostly-unlexicalized features that have low sparsity even in small training corpora. We propose such feature sets both for the re-ranking and the combination models. We also show that lexical and syntactic features from the local models, which tend to have high sparsity, do not help in our setup. A second novelty of our work is that we show that the proposed global strategies can be successfully stacked because they provide complementary benefits: in our configuration re-ranking tends to improve recall whereas the combination model boosts precision.

Our complete SRL system obtains the current best results for Spanish and Catalan: for in-domain data and correct syntactic information, our system obtains F_1 scores of 88.88 points for Catalan and 84.96 for Spanish. For out-of-domain data and gold syntax, our systems obtains 83.56 points for Catalan and 86.49 for Spanish.

Acknowledgements

Mihai Surdeanu is a research fellow within the Ramón y Cajal program of the Spanish Ministry of Science and Technology. Roser Morante was funded by the postdoctoral grant EX2005–1145 awarded by the latter institution to the project *Técnicas semiautomáticas para el etiquetado de roles semánticos en corpus del español*. This research has been partially funded by the Spanish Ministry of Education and Science, projects OpenMT (TIN2006-15307-C03-02) and TRANGRAM (TIN2004-07925-C03-02).

References

1. Pradhan, S., Hacioglu, K., Krugler, V., Ward, W., Martin, J., Jurafsky, D.: Support vector learning for semantic argument classification. Machine Learning 60, 11–39 (2005)
2. Surdeanu, M., Màrquez, L., Carreras, X., Comas, P.R.: Combination strategies for semantic role labeling. Journal of Artificial Intelligence Research (JAIR) 29, 105–151 (2007)
3. Toutanova, K., Haghighi, A., Manning, C.D.: A global joint model for semantic role labeling. Computational Linguistics 33 (forthcoming, 2008)
4. Punyakanok, V., Roth, D., Yih, W.: The importance of syntactic parsing and inference in semantic role labeling. Computational Linguistics 33 (forthcoming, 2008)
5. Pradhan, S., Hacioglu, K., Ward, W., Martin, J.H., Jurafsky, D.: Semantic role chunking combining complementary syntactic views. In: Proc. of CoNLL-2005 (2005)

6. Xue, N.: Labeling chinese predicates with semantic roles. Computational Linguistics 33 (forthcoming, 2008)
7. Màrquez, L., Villarejo, L., Martí, M., Taulé, M.: Semeval-2007 task 09: Multilevel semantic annotation of catalan and spanish. In: Proc. of SemEval 2007 (2007)
8. Schapire, R., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learning 37 (1999)
9. Gildea, D., Jurafsky, D.: Automatic labeling of semantic roles. Computational Linguistics 28, 245–288 (2002)
10. Surdeanu, M., Harabagiu, S., Williams, J., Aarseth, P.: Using predicate arguments structures for information extraction. In: Proc. of ACL (2003)
11. Xue, N., Palmer, M.: Calibrating features for semantic role labeling. In: Proc. of EMNLP (2004)
12. Carreras, X., Màrquez, L.: Introduction to the CoNLL-2005 shared task: Semantic role labeling. In: Proc. of CoNLL–2005 (2005)
13. Màrquez, L., Comas, P., Giménez, J., Català, N.: Semantic role labeling as sequential tagging. In: Proc. of CoNLL 2005 (2005)
14. Morante, R., Busser, B.: ILK2: Semantic role labelling for Catalan and Spanish using TiMBL. In: Proc. of SemEval 2007 (2007)
15. Màrquez, L., Villarejo, L., Martí, M., Taulé, M.: Semeval-2007 task 09: Multilevel semantic annotation of Catalan and Spanish. In: Proc. of SemEval-2007 (2007)
16. Morante, R., van den Bosch, A.: Memory–based semantic role labelling. In: Proc. of RANLP (2007)
17. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: Proc. of ACL (2002)
18. Younger, D.H.: Recognition and parsing of context-free languages in n^3 time. Information and Control 10, 189–208 (1967)
19. Noreen, E.W.: Computer-Intensive Methods for Testing Hypotheses. John Wiley & Sons, Chichester (1989)

A Preliminary Study on the Robustness and Generalization of Role Sets for Semantic Role Labeling

Beñat Zapirain[1], Eneko Agirre[1], and Lluís Màrquez[2]

[1] IXA NLP Group
University of The Basque Country
{benat.zapirain,e.agirre}@ehu.es
[2] TALP Research Center
Technical University of Catalonia
lluism@lsi.upc.edu

Abstract. Most Semantic Role Labeling (SRL) systems rely on available annotated corpora, being PropBank the most widely used corpus so far. Propbank role set is based on theory-neutral numbered arguments, which are linked to fine grained verb-dependant semantic roles through the verb framesets. Recently, thematic roles from the computational verb lexicon VerbNet have been suggested to be more adequate for generalization and portability of SRL systems, since they represent a compact set of verb-independent general roles widely used in linguistic theory. Such thematic roles could also put SRL systems closer to application needs. This paper presents a comparative study of the behavior of a state-of-the-art SRL system on both role role sets based on the SemEval-2007 English dataset, which comprises the 50 most frequent verbs in PropBank.

1 Introduction

Semantic Role Labeling is the problem of analyzing clause predicates in open text by identifying arguments and tagging them with semantic labels indicating the role they play with respect to the verb. Such sentence–level semantic analysis allows to determine "who" did "what" to "whom", "when" and "where", and, thus, characterize the participants and properties of the *events* established by the predicates. This kind of semantic analysis is very interesting for a broad spectrum of NLP applications (information extraction, summarization, question answering, machine translation, etc.), since it opens the avenue for exploiting the semantic relations among linguistic constituents.

The increasing availability of large semantically annotated corpora, like PropBank and FrameNet, has contributed to increase the interest on the automatic development of Semantic Role Labeling systems in the last five years. Since Gildea and Jurafsky's initial work "Automatic Labeling of Semantic Roles" [3] on FrameNet-based SRL, many researchers have devoted their efforts on this exciting and relatively new task. Two evaluation exercises on SRL were conducted by the 'shared tasks' of CoNLL-2004 and CoNLL-2005 conferences [1,2], bringing to scene a comparative analysis of almost 30 competitive systems trained on the PropBank corpus. From there, PropBank became the most widely used corpus for training SRL systems.

One of the criticisms to the PropBank corpus refers to the role set it uses, which consists of a set of numbered core arguments, whose semantic translation is verb-dependent. While Arg0 and Arg1 are intended to indicate the general roles of Agent and Theme, other argument numbers do not generalize across verbs and do not correspond to general semantic roles. This fact might compromise generalization and portability of SRL systems, specially when the training corpus is small and not very representative. Thematic roles (e.g., based on VerbNet) have been suggested to be more adequate for generalization and portability, since they represent a compact set of verb-independent general roles widely used in linguistic theory. Such thematic roles could also put SRL systems closer to application needs [11].

Thanks to a mapping from PropBank numbered arguments into VerbNet thematic roles, a version of the PropBank corpus with thematic roles has been released recently [6]. Using a part of this corpus, an English SRL task was proposed in SemEval-2007, which compared the results of the systems under both role sets [9]. Unfortunately, the number of participants in that task was too small to extract reliable conclusions.

In this paper, we go further in this direction and describe an experimental comparison between the two previous role sets (PropBank numbered arguments vs. VerbNet thematic roles). Having in mind the claim that general thematic roles should be more robust to changing domains and unseen predicates, we study the performance of a state-of-the-art SRL system training on either codification of roles and some specific settings, e.g., including/excluding verb-specific information in features, and labeling of unseen verb predicates. Although numerical results are not directly comparable we observe that the PropBank-based labeling is more robust in all previous experimental conditions (i.e., the performance decrease is less severe than in the VerbNet case). Finally, assuming that application-based scenarios would prefer dealing with general thematic role labels, we explore the best way to label a text with thematic roles, namely, by training directly on VerbNet roles or by using the PropBank SRL system and perform a posterior mapping into thematic roles.

The rest of the paper is organized as follows: Section 2 contains background on PropBank and VerbNet-based thematic roles. Section 3 presents the experimental setting of our experiments and the base SRL system used for the role set comparisons. In Section 4 the main comparative experiments on robustness are described. Section 5 is devoted to analyze the posterior mapping of PropBank-style output into VerbNet thematic roles. Finally, Sections 6 and 7, contain a discussion of the results in context and outline the main directions for future research.

2 Corpora and Semantic Role Sets

The PropBank corpus is the result of adding a shallow semantic layer to the syntactic structures of Penn Treebank II [8]. Specifically, it provides information about predicate-argument structures to all verbal predicates of the Wall Street Journal section of the treebank. The role set is theory–neutral and consists of a set of numbered core arguments (Arg0, Arg1, ..., Arg5). Each verb has a *frameset* listing its allowing role labels and mapping each numbered role to an English-language description of the semantics of the role, which is specific to that verb.

Different senses for a polysemous verb have different framesets, but the argument labels are semantically consistent in all syntactic alternations of the same verb–sense. For instance in "Kevin broke [the window]$_{Arg1}$" and in "[The door]$_{Arg1}$ broke into a million pieces", for the verb *broke.01*, both Arg1 arguments have the same semantic meaning, that is "broken entity". Nevertheless, argument labels are not necessarily consistent across different verbs (or verb senses). For instance, the same Arg2 label is used to identify the Destination argument of a proposition governed by the verb *send* and the Beneficiary argument of the verb *compose*. This fact might compromise generalization of systems trained on PropBank, which might be biased to acquire too verb-specific knowledge. In spite of that fact, and thanks to some annotation criteria, the most frequent arguments in PropBank, Arg0 and Arg1, are intended to indicate the general roles of Agent and Theme and are usually consistent across different verbs. Adjuncts (Temporal and Location markers, etc.) conform also a set of general and verb-independent labels. PropBank has become the most widely used corpus for training SRL systems due to two main reasons: first, PropBank provides a representative sample of general text with complete role-annotations; and second, the numerous international evaluations using PropBank highly promoted its usage among the researchers.

VerbNet [4] is a computational verb lexicon in which verbs are organized hierarchically into classes depending on their syntactic/semantic linking behavior. The classes are based on Levin's verb classes [5] and contain semantic and syntactic information about 4,526 verb senses (corresponding to 3,769 lexemes). Each class comprises a list of member verbs and associates their shared syntactic frames with semantic information, such as thematic roles and selectional constraints. There are 23 thematic roles (Agent, Patient, Theme, Experiencer, Source, Beneficiary, Instrument, etc.) which, unlike the PropBank numbered arguments, are considered as general verb-independent roles.

This level of abstraction makes them, in principle, more suited than PropBank numbered arguments for being directly exploited by general NLP applications. But, VerbNet by itself is not an appropriate lexical resource to train SRL systems. As opposed to PropBank, the number of tagged examples is far more limited in VerbNet. Fortunately, in the last years a twofold effort has been made in order to generate a large corpus fully annotated with thematic roles. Firstly, the SemLink[1] resource [6] established a mapping between PropBank framesets and VerbNet thematic roles. Secondly, the SemLink mapping was applied to a representative portion of the PropBank corpus and manually disambiguated [6]. The resulting corpus is currently available for the research community and makes possible comparative studies between role sets like [11] and the one in this paper.

3 Experimental Setting

3.1 Datasets

The data used in the experiments is the one provided by the SRL subtask of the English lexical sample in SemEval-2007[2]. The dataset comprises the occurrences of 50 different

[1] http://verbs.colorado.edu/semlink/

[2] http://www.cs.swarthmore.edu/semeval

verb lemmas from the WSJ portion of PropBank. It includes the part of speech and full syntactic information for each word as well as the hand tagged PropBank frame sense and the VerbNet class for verbs. The training data is a subsection from Sections 02-21 and the test data comprises Sections 01, 22, 23 and 24.

The corpus is annotated with two different semantic role sets, the PropBank role set and the VerbNet thematic role set. There is a total of 5 (core) role types for PropBank and 21 thematic roles for VerbNet. In a small number of cases, there is no VerbNet role available (e.g. when VerbNet does not contain the appropriate sense of the verb) so the PropBank role label is given instead. Apart from the argument role labels, both versions of the dataset are annotated with common adjunct like roles such as temporal, adverbial, location and so on.

The 50 verbs from the dataset cover a wide range of VerbNet classes (see table 1). Therefore, most of the classes are not strongly represented in the training set because of the relatively small size of the dataset and the large number of covered classes. Table 1 also shows the number of verb occurrences in those classes.

All in all, the training part has an average of 317.36 occurrences per verb, ranging from 8,365 for *say* to 23 for *regard*. The test has an average of 61.88 occurrences per verb, ranging from 1,665 for *say* to 4 for *grant*. The average polisemy for VerbNet is 1.71 and for PropBank is 1.70. The verbs are linked to a total of 44 VerbNet classes, with an average of 1.13 verbs per class.

3.2 SRL System

Our basic Semantic Role Labeling system represents the tagging problem as a Maximum Entropy Markov Model (MEMM). The system uses full syntactic information to select a sequence of constituents from the input text and tags these tokens with Begin/Inside/Outside (BIO) labels, using state-of-the-art classifiers and features [10]. The system achieves competitive performance in the CoNLL-2005 shared task dataset and ranked first in the SRL subtask of the SemEval-2007 English lexical sample task [12].

Maximum Entropy Markov Models are discriminative models for sequential tagging (i.e., the problem of assigning a sequence of labels $[s_1, \ldots, s_n]$ to a sequence of observations $[o_1, \ldots, o_n]$) that model the local probability distribution $P(s_i \mid s_{i-1}, \hat{o}_i)$ for each possible label s_i at position i, where $\hat{o}_i$ is the context of observation o_i and s_{i-1} the preceding label. Given a MEMM, the most likely state sequence is the one that maximizes the following formula

$$S = \text{argmax}_{[s_1,\ldots,s_n]} \prod_{i=1}^{n} P(s_i \mid s_{i-1}, \hat{o}_i)$$

Translating the problem to SRL, we have role/argument labels connected to each state in the sequence (or proposition), and the observations are the features extracted in these points (token features). We get the most likely label sequence finding out the most likely state sequence (using the Viterbi algorithm). All the conditional probabilities are given by the Maximum Entropy classifier with a tunable Gaussian prior from the Mallet Toolkit[3], which was empirically set to 0.1 in these experiments.

[3] http://mallet.cs.umass.edu

Table 1. Verbs in the dataset in alphabetic order. VerbNet (VN) class and Propbank (PB) senses are given, as well as the occurrences in the train and test sets. 'None' means that no VerbNet class was assigned.

Verb	VN	PB	train	test
affect	31.1	01	121	28
affect	None	02	1	0
allow	29.5	01	254	42
allow	29.5-1	03	1	0
allow	64	02	4	0
allow	None	02	4	0
announce	37.7	01	291	41
approve	31.3	01	173	35
ask	37.1-1	01	118	13
ask	37.1-1	02	149	20
ask	None	03	3	1
attempt	61	01	57	14
avoid	52	01	102	18
believe	29.4	01	325	54
build	26.1-1	01	224	38
build	26.2	02	39	5
build	None	03	7	4
build	None	05	5	0
buy	13.5.1	01	743	16
buy	13.5.1	02	4	0
buy	13.5.1	03	3	2
care	31.3	01	21	4
cause	27	01	195	46
claim	37.7	01	106	23
claim	None	02	2	0
complain	37.8	01	75	13
complete	55.2	01	167	30
contribute	13.2	01	103	30
describe	29.2	01	68	11
disclose	37.7	01	163	28
disclose	None	01	4	0
enjoy	31.2	01	40	8
estimate	54.4	01	255	45
examine	35.4	01	20	6
exist	47.1-1	01	105	11
explain	37.1	01	89	16
express	11.1-1	02	1	0
express	48.1.2	01	51	11
feel	29.5	02	52	17
feel	30.1	01	85	19
feel	30.4	01	6	2
feel	30.4	05	1	0
feel	31.3	03	13	1
find	13.5.1	01	195	29
find	29.4	01	170	27
find	None	01	9	1
fix	26.3-1	01	1	
fix	26.3-1	02	8	
fix	54.4	03	47	8
grant	13.3	01	30	4
hope	32.2	01	162	46
improve	45.4	01	149	28
improve	45.4	02	13	
join	22.1-2	01	120	20
kill	42.1-1	01	80	9
maintain	29.5	01	122	13
negotiate	36.1	01	82	16
occur	48.3	01	65	21
prepare	26.3-1	01	35	5
prepare	26.3-1	02	53	16
produce	26.4	01	262	52
promise	13.3	01	69	10
propose	37.7	01	198	42
prove	29.4	01	88	21
purchase	13.5.2-1	01	135	32
recall	10.2	01	6	2
recall	29.2	02	53	6
receive	13.5.2	01	326	67
regard	29.2	01	22	5
regard	None	01	1	0
remember	29.2	01	38	5
remove	10.1	01	44	3
remove	10.2	01	16	7
replace	13.6	01	84	25
report	29.1	01	455	99
report	37.7	01	74	19
report	None	01	9	1
rush	51.3.2	01	33	7
say	37.7	01	8,365	1,645

The full list of features used can be found in [12]. From that setting, we excluded the experimental semantic features based on selectional preferences, which could interfere with the interpretation of the results. The features are the same for both PropBank and VerbNet. In both cases a single MEMM classifier is trained for all verbs using all the available training data.

When searching for the most likely state sequence, the following constraints are observed[4]:

1. No duplicate argument classes for Arg0–Arg5 Propbank roles (or VerbNet roles) are allowed.
2. If there is a R-X argument (reference), then there has to be a X argument before (referent).

[4] Note that some of the constraints are dependent of the role set used, i.e., PropBank or VerbNet.

Table 2. Basic results using PropBank and VerbNet role sets

PropBank						
Experiment	correct	excess	missed	precision	recall	F_1
SemEval setting	5,703	1,009	1,228	84.97	82.28	83.60 $\pm$0.9
CoNLL setting	5,690	1,012	1,241	84.90	82.09	83.47 $\pm$0.8
CoNLL setting (no 5th)	5,687	1,019	1,244	84.80	82.05	83.41 $\pm$0.8
No verbal features	5,575	1,134	1,356	83.10	80.44	81.74 $\pm$0.9
Unseen verbs	5,125	1,282	1,639	79.99	75.77	77.82 $\pm$0.9

VerbNet						
Experiment	correct	excess	missed	precision	recall	F_1
SemEval setting	5,681	993	1,250	85.12	81.97	83.51 $\pm$0.9
CoNLL setting	5,650	1,042	1,281	84.43	81.52	82.95 $\pm$0.8
CoNLL setting (no 5th)	5,616	1,106	1,315	83.55	81.03	82.27 $\pm$1.0
No verbal features	4,941	1,746	1,990	73.89	71.29	72.57 $\pm$1.0
Unseen verbs	3,691	2,555	3,073	59.09	54.57	56.74 $\pm$0.9

3. If there is a C-X argument (continuation), then there has to be a X argument before.
4. Before a I-X token, there has to be a B-X or I-X token.
5. Given a predicate, only the arguments described in its Propbank (or VerbNet) lexical entry are allowed.

Regarding the last constraint, the lexical entries of the verbs were constructed from the training data itself. For instance, for the verb *build* the PropBank entry would only allow 4 core roles (Arg0-3), while the VerbNet entry would allow 6 roles (Product, Material, Asset, Attribute, Theme and Arg2). Note that in the cases where the PropBank (or VerbNet) sense is known (see below) we would constraint the possible arguments only to those that appear in the lexical entry of that sense, as opposed of using the arguments that appear in all senses.

4 On the Generalization of Role Sets

First, we wanted to have a basic reference of the comparative performance of the classifier on each role set. We performed two experiments. In the first one we use all the available information provided by the SemEval organizers, including the verb senses in PropBank and VerbNet. This information was available both in the test and train data, and was thus used as an additional feature by the classifier and to constraint further the possible arguments when searching for the most probable Viterbi path.

The results are shown in the 'SemEval setting' rows of Table 2. The correct, excess, missed, precision, recall and F_1 measures are reported, as customary. The significance intervals for F_1 are also reported, which have been obtained with bootstrap resampling [7]. F_1 scores outside of these intervals are assumed to be significantly different from the related F_1 score ($p < 0.05$). The precision is higher for VerbNet, but the recall is lower and the F_1 score is slightly better for PropBank. The differences are nevertheless very small, and given the confidence interval for F_1, negligible. The number of labels that the classifier has to learn in the case of VerbNet should make the task

Table 3. Detailed results on the SemEval setting for PropBank and VerbNet roles, left and right respectively. Excess and missed numbers, as well as reference arguments and verbs have been omitted for brevity. The rightmost rows show the figures for the 'no verb features' setting.

	SemEval setting								No verb feature			
	PropBank				VerbNet				PropBank		VerbNet	
	corr.	prec.	rec.	F_1	corr.	prec.	rec.	F_1	corr.	F_1	corr.	F_1
Overall	5703	84.97	82.28	83.60	5681	85.12	81.97	83.51	5575	81.74	4941	72.57
Arg0	2507	93.41	92.34	92.87					2492	91.82		
Arg1	2470	83.45	82.64	83.04					2417	81.34		
Arg2	115	72.33	65.71	68.86	0	0.00	0.00	0.00	76	48.10	0	0.00
Arg3	25	60.98	50.00	54.95	8	57.14	47.06	51.61	18	39.56	3	28.57
Arg4	0	0.00	0.00	0.00	0	0.00	0.00	0.00	0	0.00	0	0.00
Actor1					10	90.91	83.33	86.96			0	0.00
Actor2					1	100.00	100.00	100.0			1	66.67
Agent					2357	93.49	92.40	92.94			2339	89.24
Asset					15	68.18	71.43	69.77			12	52.17
Attribute					8	72.73	47.06	57.14			6	46.15
Beneficiary					14	66.67	58.33	62.22			6	33.33
Cause					36	78.26	75.00	76.60			1	3.64
Experiencer					118	90.08	89.39	89.73			5	7.14
Location					9	100.00	75.00	85.71			0	0.00
Material					1	100.00	14.29	25.00			0	0.00
Patient					28	96.55	75.68	84.85			3	14.29
Patient1					17	85.00	85.00	85.00			3	26.09
Predicate					124	73.81	68.51	71.06			58	37.42
Product					73	70.87	68.87	69.86			10	14.49
Recipient					39	88.64	81.25	84.78			36	67.29
Source					15	62.50	60.00	61.22			15	57.69
Stimulus					11	61.11	52.38	56.41			9	45.00
Theme					525	83.20	80.77	81.97			352	47.70
Theme1					52	85.25	75.36	80.00			4	10.39
Theme2					39	72.22	65.00	68.42			1	3.12
Topic					1594	86.16	85.38	85.77			1511	79.30
ArgM-ADV	97	56.40	51.60	53.89	96	55.81	51.06	53.33	97	54.19	95	53.67
ArgM-CAU	2	100.00	15.38	26.67	4	100.00	30.77	47.06	4	44.44	3	35.29
ArgM-DIR	2	100.00	50.00	66.67	2	100.00	50.00	66.67	2	66.67	2	66.67
ArgM-EXT	0	0.00	0.00	0.00	0	0.00	0.00	0.00	0	0.00	0	0.00
ArgM-LOC	104	63.03	68.87	65.82	105	61.40	69.54	65.22	103	64.38	105	65.83
ArgM-MNR	37	49.33	43.53	46.25	38	50.00	44.71	47.20	31	41.89	32	40.25
ArgM-PNC	7	58.33	25.00	35.00	8	57.14	28.57	38.10	5	26.32	6	29.27
ArgM-PRD	0	0.00	0.00	0.00	3	100.00	33.33	50.00	0	0.00	0	0.00
ArgM-REC	0	0.00	0.00	0.00	0	0.00	0.00	0.00	0	0.00	0	0.00
ArgM-TMP	265	74.65	68.65	71.52	263	73.06	68.13	70.51	263	70.79	265	70.48

harder. Given the fact that the same results are obtained with respect to PropBank could lead one to think that the VerbNet labels are easier to learn, perhaps because they are more consistent across verbs.

In fact, the detailed results for the roles in each of the sets in Table 3 (rows for the SemEval setting) seem to support this fact. The table shows the larger number of roles that need to be learned for VerbNet.

The performance for the most frequent roles is very similar in both sets. For instance, Arg0 and Agent (2,507 and 2,357 correct labels respectively) both have an F_1 score of 92%. Arg1 (with 2,470 correct labels) get 83% of F_1, but VerbNet Topic and Theme (with 1,594 and 525 correct labels) get 85% and 82% of F_1.

In the second experiment we restricted the use of hand annotated information. This setting is more natural, as it does not use any gold standard data in the test part in order to predict the roles. The results are shown in the 'CoNLL setting' rows of Table 2.

We can see that while the PropBank classifier did not suffer any appreciable loss, the thematic role classifier showed greater sensitivity to the absence of this kind of information. One possible reason could be that the VerbNet classifier is more sensitive to the argument filter (the 5th constraint) used in the Viterbi search, and lacking the sense information makes the filter less useful. In any case, neither differences are significant according to the confidence intervals.

In order to test how important is the 5th constraint, we run the CoNLL setting with the 5th constraint disabled (that is, allowing any argument). The results in the 'CoNLL setting (no 5th)' rows of Table 2 show that the drop for PropBank is negligible, but the drop in VerbNet is more important. In fact, the difference in performance from the SemEval setting to that obtained without the VerbNet class and argument constraints is statistically significant.

In the next subsections we examine the robustness and generalization capabilities for each of the role sets.

4.1 Generalization to Unseen Predicates

In principle, the PropBank core roles (Arg0–4) get a different interpretation depending of the verb, i.e. the meaning of each of the roles is described separately for each verb in the PropBank framesets. Still, the annotation criteria set for PropBank tried to make the two main roles accounting for most of the occurrences consistent across verbs. In VerbNet, to the contrary, all roles are completely independent of the verb, in the sense that the interpretation of the role does not vary across verbs. But, at the same time, each verbal entry lists the possible roles it accepts, and which combinations are allowed.

This experiment tests the sensitivity of the role sets when the classifier encounters a verb which does not occur in the training data. This is a realistic case, as in many cases, verbs without training data are found in the target corpora to be processes. In principle, we would expect the set which is more independent across verbs to be more robust. We artificially created a test set for unseen verbs. We first chose 10 verbs at random, and removed their occurrences from the training data, yielding 13,146 occurrences for the 40 verbs. In order to have a sizeable test set, we tested on the 2,723 occurrences of those 10 verbs in the train set (see Table 4).

The results obtained after training and testing the classifier are shown in the last rows in Table 2. Note that they are not directly comparable to the other results mentioned so far, as the test set is a subset of the original test set. The figures indicate that the performance of the PropBank argument classifier is considerably higher than the VerbNet classifier, with a 20 point gap.

Table 4. Verbs used in the *unseen verb* experiment

Train	*affect, announce, ask, attempt, avoid, believe, build, care, cause, claim, complain, complete, contribute, describe, disclose, enjoy, estimate, examine, exist, explain, express, feel, fix, grant, hope, join, maintain, negotiate, occur, prepare, promise, propose, purchase, recall, receive, regard, remember, remove, replace, say*
Test	*allow, approve, buy, find, improve, kill, produce, prove, report, rush*

This experiment shows that not knowing the verbal head, the classifier has a very hard time to distinguish among the fine-grained VerbNet roles. In order to confirm this, we performed further analysis, as described in the next subsection.

4.2 Sensitivity to Verb-Dependent Features

In this experiment we want to test the sensitivity of the sets when the classifier does not have any information of the main verb in the sentence where it is tagging the argument and adjuncts. We removed from the training and testing data all the features which make any reference to the verb, including, among others: the surface form, lemma and POS of the verb, and all the combined features that include the verb form (please, refer to [12] for a complete description of the feature set used).

The results are shown in the 'No verbal features' rows of Table 2. The performance drop in PropBank is small, on the fringe of being statistically significant, but the drop for VerbNet is dramatic, 10 points in precision, recall and F_1 with clear statistical significance. A closer look at the detailed role-by-role performances can be done if we compare the F_1 rows in the SemEval setting and in the 'no verb features' setting in Table 3. Those results show that both Arg0 and Arg1 are very robust to the lack of target verb information, while Arg2 and Arg3 get more affected. Given the relatively low number of Arg2 and Arg3 arguments, their performance drop does not affect much the overall PropBank performance. In the case of VerbNet, the picture is very different. While the performance drop for Agent and Topic is of 2 and 5 points respectively, the other roles get very heavy losses: Theme and Predicate get their F_1 halfed, and the rest of roles are barely found. It is worth noting that the adjunct labels get very similar performances in all cases.

The robustness of the PropBank roles can be explained by the fact that the PropBank taggers tried to be consistent when tagging Arg0 and Arg1 across verbs. We also think that both Arg0 and Arg1 can be detected quite well relying on unlexicalized syntactic features only, i.e. not knowing which are the verbal and nominal heads. On the other hand, distinguishing between Arg2–4 is more dependant on the subcategorization frame of the verb, and thus more sensitive to the lack of verbal information.

In the case of VerbNet, the more fine-grained distinction among roles seems to depend more on the meaning of the predicate. For instance, distinguishing between Theme and Recipient, not to say about Theme, Theme1 and Theme2. The lack of the verbal head makes it much more difficult to distinguish among those roles.

5 Mapping into VerbNet Thematic Roles

As mentioned in the introduction, the interpretation of PropBank roles depends on the verb, and that makes them less suitable for NLP applications. VerbNet roles, on the other hand, have a direct interpretation. In this section, we test the performance of two different approaches to tag input sentences with VerbNet roles:

1. train on corpora tagged with VerbNet, and tag the input directly
2. train on corpora tagged with PropBank, tag the input with PropBank roles, and use a PropBank to VerbNet mapping to output VerbNet roles

The results for the first approximation are already available (cf. Table 2). For the second approximation, we just need to map PropBank roles into VerbNet roles using Semlink [6]. We devised two experiments. In the first one we use the hand-annotated verb class in the test set. For each verb governing a proposition we translate PropBank roles into VerbNet roles making use of the SemLink mapping information corresponding to that verb lemma and its verbal class.

For instance, consider an occurrence of *allow* in a test sentence. If the occurrence has been manually annotated with the VerbNet class 29.5, we can use the following entry in Semlink to add the VerbNet role Predicate to the argument labeled with Arg1, and Agent to the Arg0 argument.

```
<predicate lemma="allow">
    <argmap pb-roleset="allow.01" vn-class="29.5">
      <role pb-arg="1" vn-theta="Predicate" />
      <role pb-arg="0" vn-theta="Agent" />
    </argmap>
</predicate>
```

The results obtained using the hand-annotated VerbNet classes (and the SemEval setting for Propbank), are shown in the first row of Table 5. If we compare these results to those obtained by VerbNet in the SemEval setting (second row of Table 5), they are only 0.1 lower, and the difference is not statistically significant.

Table 5. Results on VerbNet roles using two different strategies. The 'PropBank to VerbNet' rows show the results using the mapping. The results for directly using VerbNet are taken from Table 2.

experiment	corr.	excess	missed	prec.	rec.	F_1
PropBank to VerbNet (hand)	5,680	1,009	1,251	84.92	81.95	83.41 $\pm$0.9
VerbNet (SemEval setting)	5,681	993	1,250	85.12	81.97	83.51 $\pm$0.9
PropBank to VerbNet (most frequent)	5,628	1,074	1,303	83.97	81.20	82.56 $\pm$0.8
VerbNet (CoNLL setting)	5,650	1,042	1,281	84.43	81.52	82.95 $\pm$0.8

In a second experiment, we discarded the sense annotations from the dataset, and tried to predict the VerbNet class of the target verb using the most frequent class for the verb in the training data. The accuracy of choosing the most frequent class is of 97% on the training. In the case of *allow* the most frequent class is 29.5 (cf. Table 1), so we would use the same Semlink entry as above. The third row in Table 5 shows the results using the most frequent VerbNet class (and the CoNLL setting for PropBank). The performance drop compared to the use of the hand-annotated VerbNet class, is small, and barely statistically significant, and only 0.4 from the results obtained directly using VerbNet on the same conditions (fourth row of the same Table).

All in all, the second experiment shows that, in realistic conditions, using VerbNet directly provides the same results than tagging with PropBank roles, disambiguating with the most frequent VerbNet class and then using Semlink for mapping. These results may imply that the classifier is not able to learn better from VerbNet roles rather than PropBank roles.

6 Related Work

As far as we know, there are only two other works doing an extensive comparison of different role sets on the same test data.

Gildea and Jurafsky [3] mapped FrameNet frame elements into a set of *abstract thematic roles* (i.e., more general roles such as Agent, Theme, Location), and concluded that their system could use these thematic roles without degradation in performance.

Yi and Loper [11] is a closely related work, and as far as we know, the only other work doing an extensive comparison of different role sets on the same test data. The authors also compare PropBank and VerbNet role sets, but they focus on the performance of Arg2. The authors show that splitting Arg2 instances into subgroups based on thematic roles improves the performance of the PropBank-based classifier, especially in out-of-domain experiments (Brown corpus).

Note that the authors do not use purely VerbNet roles, but a combination of grouped VerbNet roles (for Arg2) and PropBank roles (for the rest of arguments). In contrast, our study compares both role sets as they stand, without modifications, and our results show that VerbNet roles are less robust and not easier to learn than PropBank roles. While not in direct contradiction, both studies show different angles of the complex relation between the different role sets.

7 Conclusion and Future Work

In this paper we present a preliminary study of the performance of a state-of-the-art SRL system training on either codification of roles and some specific settings, e.g., including/excluding verb-specific information in features, and labeling of infrequent and unseen verb predicates. We observed that the PropBank-based labeling is more robust in all previous experimental conditions (i.e., the performance decrease is less severe than in the VerbNet case). Finally, assuming that application-based scenarios would prefer dealing with general thematic role labels, we explore the best way to label a text with thematic roles, namely, by training directly on VerbNet roles or by using the PropBank SRL system and perform a posterior mapping into thematic roles. In this case, we find that the difference is not statistically significant.

Regarding future work, we want to extend this work to all the verbs in VerbNet. Among other things, we would like to test whether having more verbs to train affects the relative performance of PropBank and VerbNet. We would also like to improve the results for the VerbNet role set using role groupings in order to reduce the sparsity of the data. Finally, we would like to revisit the portability results of [11] using our setting.

Acknowledgements

This work has been partially funded by the Department of Education, Universities and Research of the Basque Government (IT-397-07) and by the Ministry of Education Projects KNOW and OpenMT (TIN2006-15049-C03-01; TIN2006-15307-C03-02). Beñat is supported by a PhD grant from the University of the Basque Country.

References

1. Carreras, X., Màrquez, L.: Introduction to the conll-2004 shared task: Semantic role labeling. In: Ng, H., Riloff, E. (eds.) Proceedings of the Eigth Conference on Computational Natural Language Learning (CoNLL-2004), Boston, MA, USA, May 2004, pp. 89–97. Association for Computational Linguistics (2004)
2. Carreras, X., Màrquez, L.: Introduction to the CoNLL-2005 shared task: Semantic role labeling. In: Dagan, I., Gildea, D. (eds.) Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005), Ann Arbor, Michigan, USA, June 2005, pp. 152–164. Association for Computational Linguistics (2005)
3. Gildea, D., Jurafsky, D.: Automatic labeling of semantic roles. Computational Linguistics 28(3), 245–288 (2002)
4. Kipper, K., Dang, H.T., Palmer, M.: Class based construction of a verb lexicon. In: Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-2000), Austin, TX (July 2000)
5. Levin, B.: English Verb Classes and Alternations: A Preliminary Investigation. The University of Chicago Press, Chicago (1993)
6. Loper, E., Yi, S.-T., Palmer, M.: Combining lexical resources: Mapping between propbank and verbnet. In: Proceedings of the 7th International Workshop on Computational Linguistics, Tilburg, the Netherlands (2007)
7. Noreen, E.W.: Computer-Intensive Methods for Testing Hypotheses. John Wiley & Sons, Chichester (1989)
8. Palmer, M., Gildea, D., Kingsbury, P.: The proposition bank: An annotated corpus of semantic roles. Computational Linguistics 31(1), 71–105 (2005)
9. Pradhan, S., Loper, E., Dligach, D., Palmer, M.: Semeval-2007 task-17: English lexical sample, SRL and all words. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, June 2007, pp. 87–92. Association for Computational Linguistics (2007)
10. Surdeanu, M., Màrquez, L., Carreras, X., Comas, P.R.: Combination strategies for semantic role labeling. Journal of Artificial Intelligence Research (JAIR) 29, 105–151 (2007)
11. Yi, S.-T., Loper, E., Palmer, M.: Can semantic roles generalize across genres? In: Proceedings of the Human Language Technology Conferences/North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL-2007) (2007)
12. Zapirain, B., Agirre, E., Márquez, L.: Sequential SRL using selectional preferences: An aproach with Maximum Entropy Markov Models. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval 2007), Prague, Czech Republic, Association for Computational Linguistics (2007)

XTM: A Robust Temporal Text Processor

Caroline Hagège[1] and Xavier Tannier[2]

[1] Xerox Research Centre Europe, 6 Chemin de Maupertuis, 38240 Meylan , France
`Caroline.Hagege@xrce.xerox.com`
[2] LIMSI, 91403 Orsay, France
`Xavier.Tannier@limsi.fr`

Abstract. We present in this paper the work that has been developed at Xerox Research Centre Europe to build a robust temporal text processor. The aim of this processor is to extract events described in texts and to link them, when possible, to a temporal anchor. Another goal is to be able to establish temporal ordering between the events expressed in texts. One of the originalities of this work is that the temporal processor is coupled with a syntactic-semantic analyzer. The temporal module takes then advantage of syntactic and semantic information extracted from text and at the same time, syntactic and semantic processing benefits from the temporal processing performed. As a result, analysis and management of temporal information is combined with other kinds of syntactic and semantic information, making possible a more refined text understanding processor that takes into account the temporal dimension.

1 Motivation

Although interest in temporal and aspectual phenomena is not new in NLP and AI, temporal processing of real texts is a topic that has been of growing interest in recent years (see [5]). The usefulness of temporal information has become clear for a wide range of applications like multi-document summarization, question/answering systems (see for instance [10]) and information extraction applications. For presenting search results, Google also offers now, in an experimental way, a timeline view to provide results of a search (see www.google.com/experimental). Temporal taggers and annotated resources such as TimeBank ([7]) have been developed. An evaluation campaign for temporal processing has also been organized recently (see [11]).

But still, it remains a challenge to associate automatically with a temporal anchor, all the events denoted in texts, and to be able to compute in many cases temporal relations holding between the different events.

Some reasons for this difficulty are:

- Temporal information is conveyed by a wide range of different sources (lexical semantic knowledge, grammatical aspect, morphological tenses) that have to be combined in order to resolve the temporal value.

- Extra-linguistic knowledge is necessary to process temporal ordering properly (e.g. in "*He opened the door and went out*", world-knowledge tells us that *opening the door* occurred just before *going out* while in "*He ate and drank*" is an assertion of a general level and no temporal order can be stated here.
- Some reasoning is necessary (e.g. if an event occurred before another event which is simultaneous to a third event, then it is possible to state that the first event happened before the third one).

The work we perform concerning temporal processing of texts is part of a more general text understanding process. Temporal processing is integrated into a more general tool, XIP, which is a general purpose linguistic analyzer [2]. Temporal analysis is thus intertwined with syntactico-semantic text processing including deep syntactic analysis and determination of thematic roles [4].

In the first part of this paper, we present our temporal processor. Details on how we perform our three-level temporal processing are then given. Then, we present the results obtained by our system in the context of the TempEval campaign [11]. As a conclusion, we give some directions for future work.

2 XTM a Temporal Module for Robust Linguistic Processing

Our temporal processor, called XTM (for XIP Temporal Module), is an extension of XIP [2]. XIP performs robust and deep syntactic analysis. *Robust* means here that any kind of text can be processed by XIP (including output of an OCR system or ill-formed input). And *deep* means that linguistic information extracted by the parser can be of a subtle nature and not necessarily straightforward. XIP extracts not only superficial grammatical relations in the form of dependency links, but also general thematic roles between a predicate (verbal or nominal) and its arguments. For syntactic relations, long distance dependencies are taken into account and arguments of infinitive verbs are handled. See [3] for details on deep linguistic processing using XIP.

Temporal processing is first performed in parallel with incremental linguistic processing and then in an independent way for temporal inference and calculations. We will first give a brief reminder of XIP and explain why it is an advantage to consider linguistic and temporal processing simultaneously.

2.1 XIP – A General Purpose Deep Syntactic Analyzer

XIP is rule-based and its architecture can be roughly divided into the three following parts:

- A pre-processing stage is integrated into XIP and handles tokenization, morphological analysis and POS tagging.
- A surface syntactic analysis stage consists in chunking the input. This stage also includes a Named Entity Recognition (NER) process.
- A deeper processing performs first a generic syntactic dependency analysis (detection of main syntactic relations as "subject", "direct object", "determination"

etc.) and then, based on the result of this generic stage, a deeper analysis (some thematic roles, clause embedding, etc.)

Further extensions to the core XIP analysis tool, dealing for example with pronominal co-reference or metonymy of named entities, have been developed and can be plugged in.

2.2 Intertwining Temporal Processing and Linguistic Processing

Temporal processing is integrated into XIP. We consider that temporal processing is one step in a more general task of text understanding. For this reason, all temporal processing at the sentence level is performed together with other tasks of linguistic analysis. Association between temporal expressions and events is considered as a particular case of the more general task of attaching thematic roles to predicates (the TIME and DURATION roles). On the other hand, a proper tagging of temporal expressions is beneficial to the task of parsing, because the proper handling of these complex expressions avoids possible errors in general chunking and dependency computatin. For instance, chunking a complex temporal expression like "*2 days before yesterday*" as a single unit in a sentence like "*They met 2 days before yesterday*" allows us to avoid having an erroneous adjunct *two days* attached to *met*.

We will detail in sections 3.2.1 and 3.2.2 how low-level (i.e. sentence level) temporal processing is combined with the rest of general purpose linguistic processing. But a temporal annotation that aims at ordering events appearing in text along a time line cannot be performed only at the sentence level. In section 3.2.3 we detail how we perform temporal processing at the level of the whole document, and how temporal calculations and inference are done.

3 Details on Temporal Processing

Before entering into details on how temporal processing is handled in XTM, some preliminary definitions are necessary. More precisely, because one of the final goals is to be able to time stamp and to order chronologically events denoted in the text, we have to clarify what we consider as "temporal relations" and as "events".

3.1 Preliminary Definitions

Temporal Relations

The set of temporal relations we use is the following: AFTER, BEFORE, DURING, INCLUDES, OVERLAPS, IS_OVERLAPPED and EQUALS (see Figure 1). They are defined as equivalent to or disjunctions of Allen's 13 relations [1]. They are simpler than Allen's relations, which makes sense in most fuzzy natural language situations, but they preserve the basic properties of Allen algebra, such as mutual exclusivity, exhaustivity, inverse relations and the possibility to compose relations. This choice is explained in more details in [6].

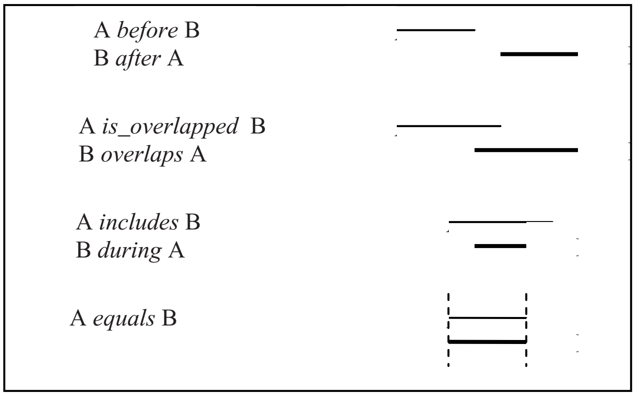


Fig. 1. Temporal relations used in XTM

Events

Temporal expressions are attached to, and temporal ordering applies to, events. It is not straightforward to define what is an event. The question of how to consider stative verbs (temporally annotable or not), as well as deverbal nouns like *destruction* or *birth*, is a difficult one. In our approach, we decided to consider as events (that can be temporally anchored) the following linguistic elements:

- Any verb (expressing either an action or a state[1])
- Any deverbal noun, when there is a clear morphological link between this noun and a verb (e.g. "interaction" is derived from the verb "interact").
- Any noun which is not a deverbal noun and that can be either:
  - An argument of preposition *during* (e.g. *during the war*)
  - A subject of verbs *to last, to happen* or *to occur*, when these verbs are modified by an explicit temporal expression (e.g. *the siege lasted three days*).

We call this last class of nouns "time span nouns". Examples of such nouns are words like *sunrise* or *war*, which intuitively correspond to nouns denoting events of certain duration. A list of these nouns (whichmay not be exhaustive) has been obtained by applying the above-mentioned heuristics to the Reuters corpora collection at NIST and by removing all deverbal nouns from the obtained list.

3.2 A Three-Level Temporal Analysis

We distinguish in our system three main levels during the processing of temporal expressions. This temporal processing has the following purposes:

- Recognizing and interpreting temporal expressions (section 3.2.1)
- Attaching these expressions to the corresponding events they modify and ordering events appearing in the same sentence (section 3.2.2)
- Ordering events in the whole document. (section 3.3.3)

[1] Although stative verbs and action verbs have different semantic properties that may impact temporal inference as stated in [5].

3.2.1 Local Level

At this level, the main task is the recognition of temporal expressions and the attribution of a value to these expressions.

The first question that is raised concerns the definition of boundaries (tokenization) of complex temporal expressions. Should complex temporal expressions like *10 days ago yesterday*, or *during 10 days in September* be considered as a whole or should they be split into different tokens?

In the standard TimeML [9], signals (prepositions "in", "during", "after", adverb "ago", etc.) are not included in temporal expressions, so these kinds of tokens are generally split. But this is not our approach. Indeed, our aim is to produce temporal tokens that are semantically consistent, and that can be associated with a normalized representation.

We consider the following criteria, which are syntactically and semantically motivated:

A complex temporal expression has to be split into minimal temporal tokens if:

1. each minimal temporal token is syntactically valid when attached to the modified event
2. each combination event + minimal temporal expression must be logically implied by the combination event + complex temporal expression.

Here are some examples illustrating this definition:

each week in "*We met each week*"
The expression *each week* cannot be split into *each* and *week* as condition 1 is not satisfied (The expression *We met each* is not syntactically valid).

twice each week in "*We met twice each week*"
This expression could be split into two minimal expressions *twice* and *each week* according to condition 1. However, condition 2 is not satisfied as *we met twice* is not implied by *we met twice each week.* For this reason, this expression has to be considered as a whole.

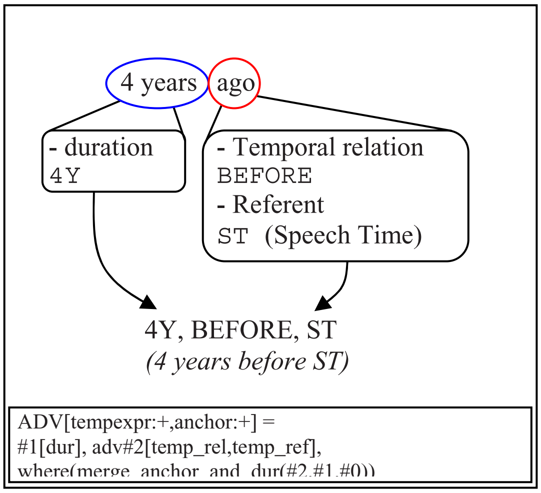
10 days in September in "*We traveled 10 days in September*"
This expression has to be split into two minimal temporal tokens (*10 days* and *in September*). Both condition 1 and 2 are verified (*we traveled 10 days in September* implies both *we traveled 10 days* and *we traveled in September).*

Having defined these criteria for determining precisely what a minimal temporal token is, we perform recognition of temporal expressions by local rules to which optional left and right contexts can be added. This is done using the XIP formalism, and this processing stage occurs just before general chunking rules. Some actions are associated with the contextual rewriting rules. These actions are meant to attribute a value to the resulting temporal expression (left hand side of the rule). Technically, these actions are calls to Python functions that can be executed directly from the parser [8].

Figure 2 illustrates this stage with an example rule for a simple anchor date.The rule builds an ADV (adverbial) node with associated Boolean features (on the left hand side of the “=” symbol) from linguistic expressions such as “4 years ago” (which matches the right hand side of the rule between “=” and the keyword “where”). Note that there is a call to function “merge_anchor_and_dur” whose parameters are three linguistic nodes (#0 represents the resulting expression on the left hand side of the rule).

3.2.2 Sentence Level

The sentence level corresponds roughly to the post-chunking stage in a XIP grammar. Once chunks and local grammar expressions have been delimited, relations between linguistic nodes are established. These relations represent syntactic and semantic dependencies between linguistic elements. For instance, the grammatical relation SUBJECT is established between the head of a subject noun phrase (NP) and the verb.



```
ADV[tempexpr:+,anchor:+] =
#1[dur], adv#2[temp_rel,temp_ref],
where(merge_anchor_and_dur(#2,#1,#0))
```

Fig. 2. Local level processing, anchor date

This is the natural place where some links between temporal expressions and the events they modify are established, as well as temporal relations between events in the same sentence. Verbal tenses are also explicitly extracted at this stage by using morphological information coming from the pre-processing stage. Furthermore, at this stage, some underspecified normalization is performed at a local level.

Attaching temporal expressions to events

As a XIP grammar is applied in an incremental way, in a first stage, any prepositional phrase (PP), including temporal PP, is attached to the predicate it modifies through a very general MOD (modifier) dependency link. Then, in a later stage, these dependency links are refined considering the nature and the linguistic properties of the linked constituents.

In the case of temporal expressions, which have been previously recognized at the local level, a specific relation TEMP links each temporal expression to the predicate it is attached to.

For instance, in the sentence "*People began gathering in Abuja Tuesday for the two day rally",* the following dependencies are extracted:

TEMP(began, Tuesday)
TEMP(rally, two day)

Tuesday being recognized as a date and *two day* as a duration.

Temporal relations between events in the same sentence
Using the results of the linguistic analysis, which gives the structure of a sentence (i.e. what is the main verb, where are the embedded clauses depending on this main verb, what kind of subordination holds between the verbs, what is the sequence of tenses), some intra-sentential temporal ordering of events is possible.

Using the temporal relations presented above, the system can detect in certain syntactic configurations if predicates in the sentence are temporally related and what kind of relations exist between them. When it is explicit in the text, a temporal distance between the two events is also calculated.

The following two examples illustrate these temporal dependencies:

This move <u>comes</u> ***a month after*** *Qantas <u>suspended</u> a number of services.*

In this sentence, the clause containing the verb *suspended* is embedded into the main clause headed by *comes*. These two events have a temporal distance of one month which is expressed by the expression *a month after*. We obtain the following relationships.

ORDER[before](suspended, comes)
DELTA(suspended, comes, a month)

They express that the event *suspended* is before the event *comes* with an interval of *a month* (analyzed as a duration whose value has been calculated at the local level, see section 3.1).

In the second example:
After ***ten years*** *of <u>boom</u>, they're <u>talking</u> about layoffs.*

boom is embedded in the *talking* clause, and an ordering can be inferred, as well as a duration of the event *boom*:

ORDER[before](boom, talking)
TEMP(boom, ten years)

Verbal tenses and aspect
Morphological analysis gives some information about tenses. For instance, the form "said" bears the feature "past:+" indicating that this form is a past tense. However this information is not enough because it is only attached to a single lexical unit.

As verbal forms appear very often as a combination of different lexical units (auxiliaries, past participles, gerunds, bare infinitives etc.) together with morphological inflection on the finite forms, we have to take all these elements into account in order to decide what the final tense of the whole verb chain is. This final tense may be underspecified in the absence of sufficient context.

3.2.3 Document Level

Beyond sentence-level, the system is only at the first stage of development. We are only able to complete relative dates in some cases, and to infer new relations with the help of composition rules, by saturating the graph of temporal relations [6].

Dates which are relative to speech time can be calculated from the document creation time (DCT), when available. We use a fine-grained but fuzzy temporal calculus module. For example, considering a DCT on March 30, 2007, the expression "2 years ago" rarely refers to March 30, 2005 (unless explicit adverbs like "exactly").

Each unit of time has a "fuzzy granularity". For example, for minutes:

"17 minutes ago" means "exactly 17 minutes ago", not 16 or 18

"15 minutes ago" or "20 minutes ago" can be understood as fuzzy, because the "fuzzy granularity" (FG) of minutes is 5 minutes.

For years, the FG is also 5 (cf "17 years ago" versus "20 years ago").

4 Evaluation

Our temporal processor has been evaluated in the context of the first evaluation campaign for temporal relation TempEval, which has been organized in 2007, within the scope of SemEval (Verhagen et al., 2007). The participants were proposed three tasks:

- Task A: identifying temporal relations holding between time and event expressions within the same sentence
- Task B: identifying temporal relations holding between event expressions and Document Creation Time (DCT)
- Task C: identifying temporal relations holding between main events of adjacent sentences.

For each task, events and temporal expression boundaries were provided to the participants together with information about tense and verbal aspects for the events. The conversion of linguistic temporal expressions into absolute dates was also provided. We chose not to use this information as we had our own event linking (integrated into the parser) and also our own processing for temporal expression normalization. We also decided to rely on our own morphological information about tense and aspect. In this way, we also indirectly evaluated the capability of our module to extract events and temporal expressions, to link them and to normalize them. We simply mapped our results to the TempEval framework afterwards. We participated in the three tasks and obtained the following results:

Tasks A and B were evaluated together. We obtained the best precision for relaxed matching (0.79 for task A, 0.82 for task B), but with a low recall (respectively 0.50 and 0.60). Strict matching is not very different. Another interesting figure is that less than 10% of the relations are totally incorrect (e.g.: BEFORE instead of AFTER).

Task C was more exploratory. The document-level stage of our system is not fully developed yet. Even more than for task AB, the fact that we chose not to use the provided TIMEX3 values makes the problem harder. Our gross results are quite low, and we used a default of OVERLAP for each unfound relation. The result was equal precision and recall of 0.58, which was the second best score.

However, assigning OVERLAP to all 258 links of task C led to a baseline precision and recall of 0.508; no team managed to bring a satisfying trade-off in this task.

Full results are given in the TempEval overview paper [11].

5 Conclusion and Further Work

We have developed a temporal processing module integrated within a more general tool for syntactic and semantic analysis. This module has been evaluated in the context of the TempEval initiative and we feel that the results are encouraging considering that we obtained good results and that we do not use all of the information that was provided for the competition. Furthermore, preliminary tests have shown that our system can also handle texts from any style and genres producing the same kind of results. One of the advantages of our approach is that temporal processing and syntactico-semantic processing can benefit from each other (linking temporal expressions and events is a special case of syntactic attachment and at the same time an early and correct chunking and characterization of temporal expression avoids errors in syntactic analysis (e.g. temporal noun phrases are generally neither subject nor direct object of a verbal predicate).

Another advantage of our incremental approach (three levels of processing) is that we can, according to different application needs, tune our module so that we can have a partial temporal processing (From a simple linking of events to full temporal inference).

However, many problems remain. Some of them are typical problems of temporal processing, others are more general but their solution should be beneficial for a proper temporal treatment:

- How to determine the temporal focus ? (i.e. the temporal reference changes according to the discourse).
- Anaphora between events is not detected by our system. If this were done, we would be able to use time anchor of one event to determine the time anchor of the co-referent event.

We hope to be able in the future to address some of these problems in order to have a more and more refined time processor able to take into account rich semantic information.

References

1. Allen, J.: Toward a general theory of action and time. Artificial Intelligence 23, 123–154 (1984)
2. Aït-Mokhtar, S., Chanod, J.P., Roux, C.: Robustness beyond Shallowness: Incremental Deep Parsing. Natural Language Engineering 8, 121–144 (2002)

3. Brun, C., Hagège, C.: Normalization and Paraphrasing using Symbolic Methods. In: 2nd Workshop on Paraphrasing. Proceedings of ACL 2003, Sapporo, Japan (2003)
4. Hagège, C., Roux, C.: Entre syntaxe et sémantique: Normalisation de l'analyse syntaxique en vue de l'amélioration de l'extraction d'information. In: Proceedings of TALN 2003, Batz-sur-Mer, France (2003)
5. Mani, I., Pustejovsky, J., Gaizauskas, R. (eds.): The Language of Time A reader. Oxford University Press, Oxford (2005)
6. Muller, P., Tannier, X.: Annotating and measuring temporal relations in texts. In: Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004), Geneva, Switzerland, pp. 50–56 (2004)
7. Pustejovsky, J., Hanks, P., Saurí, R., See, A., Gaizauskas, R., Setzer, A., Sundheim, B.: The TIMEBANK Corpus. Corpus Linguistics, Lancaster, U.K (2003)
8. Roux, C.: Coupling a linguistic formalism and a script language. CSLP-2006, Coling-ACL, Sydney, Australia (2006)
9. Saurí, R., Littman, J., Knippen, B., Gaizauskas, R., Setzer, A., Pustejovsky, J.: TimeML Annotation Guidelines (2006)
10. Schilder, F., Habel, C., Versley, Y.: Temporal information extraction and question answering: deriving answers for where-questions. In: 2nd CoLogNET-ElsNET Symposium, Amsterdam (2003)
11. Verhagen, M., Gaizauskas, R., Schilder, F., Hepple, M., Katz, G., Pustejovsky, J.: SemEval-2007 – Task 15: TempEval Temporal Relation Identification. In: SemEval workshop in ACL (2007)

What We Are Talking about and What We Are Saying about It

Eva Hajičová

Charles University
Institute of Formal and Applied Linguistics
Malostranské nám. 25, 118 00 Prague, Czech Republic
hajicova@ufal.mff.cuni.cz

Abstract. In view of the relationships between theoretical, computational and corpus linguistics, their mutual contributions are discussed and illustrated on the issue of the aspect of language related to the information structure of the sentence, distinguishing "what we are talking about" and "what we are saying about it".

1 Introduction

The name of the research domain of Computational Linguistics seems to be self-explanatory; however, there has always been a dispute what exactly 'computational' means (especially from the point of view of the relation between its theoretical and applied aspects and from the point of view of its supposedly narrowing scope due to the prevalent use of statistical methods). In addition, with the expansion of the use of computers for linguistic studies based on very large empirical language material, and, consequently, with the appearance of an allegedly new domain, corpus linguistics, a question has emerged what is the position of corpus linguistics with regard to computational linguistics.

After a summary of some of the issues related to the problem of 'how many linguistics there are' (Sect. 2), we briefly sketch in which respects the different 'linguistics' can mutually contribute to each other (Sect. 3). The main objective of our paper is to illustrate on an example of a linguistically based multi-layered annotation scenario (Sect. 4) and of a selected linguistic phenomenon, namely the information structure of the sentence (Sect. 5.1), how linguistic theory can contribute to a build-up of an integrated scenario of corpus annotation (Sect. 5.2) and, in the other direction, how a consistent application of such a scenario on a large corpus of continuous texts can provide a useful feedback for the theory (Sect. 5.3). In Section 6, some conclusions will be drawn from the personal experience with working with the given theory and scenario.

2 How Many Linguistics?

If the terms *computational linguistics* and *corpus linguistics* are understood rather broadly, as covering those domains of linguistics that are based on the use of computers and on the creation and use of corpora, respectively, then it can be seen that the intersection of the two domains is very large. (Speaking of corpora, what we have in mind are corpora implemented in computers and patterned as data bases.) However, it is important to be aware also of a third domain that develops along with the two mentioned ones, and this is *theoretical linguistics*.

Certainly, there is no descriptive framework universally accepted; there are many different trends in linguistics, as there were a hundred years ago.[1] This diversity, which perhaps is even growing, offers certain advantages, among which there is the possibility of fruitful discussions. Different points of view help to throw light on problems discussed and to make choice between the available approaches or their parts. However, the diversity of views also constitutes a source of possible misunderstandings, especially if one of the various potential aims of research is seen as the only goal worth of serious studies, or as a goal of itself, standing higher than others.

If the different points of view and goals are soberly examined, then a highly effective collaboration of researchers working in the different domains can be achieved. It is important to look for reliable results, not deciding *a priori* whether they may be found in this or that trend, but rather basing the discussions on arguments. We do not understand it as appropriate to distinguish between "computational", "corpus" and "real" linguists, the more so if the latter were to be seen as those who avoid using computers for other aims than for the creation of large corpora and of search procedures, without using clear operational criteria for classifying the items to be described. The discussion on theoretical characterization of linguistic phenomena and the computerized checking of the adequacy of descriptive frameworks belong to fundamental goals in linguistics.

In the context of computational linguistics, dependency based grammar always has played an important role, competing with phrase structure (or transformation) based approaches. A framework of this kind offers a way to conceiving the core of language, based on prototypical phenomena, as patterned in a way that comes close to elementary logic, and thus to general human mental capabilities.[2]

3 Mutual Enrichment: Task of Corpus Annotation

From what has been said above, it is certainly significant to be aware of the requirements of a systematic, intrinsic collaboration (if not a symbiosis) of corpus oriented and computational linguistics with linguistic theory.

Several linguists still prefer to work without computers and computer corpora, or to avoid statistical methods, since these may appear as attempts to do without linguistic analyses, using just the outer "brute force". Nowadays, however, statistical methods do bring important results, thanks to factors such as their

[1] We are not concerned here with what is sometimes called hyphenated linguistics - socio-linguistics, ethno-linguistics, pragmalinguistics etc.

[2] On the other hand, contextually restricted rules are then needed for the handling of the large and complex periphery, containing secondary items of different levels, as well as all asymmetries between (underlying) sentence structure and morphemics (ambiguity, synonymy, irregularities, exceptions).

connection with different possibilities of automatic learning and of a computerized, more or less automatic search for appropriate classifications of linguistic phenomena on different layers.

Other researchers see an attractive goal, or even the center of all appropriate uses of computers in linguistics, in gathering large corpora with searching procedures.

Still others are aware of the fact that, along with the mentioned goals, there is also the need to use corpora for theoretical studies. According to their views, a linguist studying e.g. the system of tenses of the English verb should not only collect the occurrences of forms in *-ed*, *-ing*, etc., from a corpus and then select, comment and classify those of them that express tenses, but should also work with procedures that identify the forms of preterit, future, etc. and enable the researcher to start immediately to analyze their functions or their combinatorics, and so on. Similarly, it is of advantage to get at once all the occurrences of subjects, direct or indirect objects, etc. in a corpus. Such tasks require not only to assign part-of-speech (POS) annotations, but also to integrate syntactic annotations into the work with large corpora. As H. Uszkoreit ([1]) has put it: time has come for deep parsing, and thus, let us add, also for deep corpus annotation. A qualified choice between the existing theoretical approaches (or their parts and ingredients) is necessary to make it possible to use corpora effectively for the aims of theoretical linguistics, as well as of frameworks oriented towards pedagogical and other applications.

Such use of corpora in theoretical linguistic studies includes aims as the following:

(i) to offer new, substantially better conditions for most diverse kinds of research in linguistics itself as well as in neighboring domains ranging from theory of literature to information retrieval;
(ii) to check existing descriptive frameworks or their parts, having in mind improvements of their consistency, their enrichment or, in the negative case, the abandonment of falsified hypotheses;
(iii) on the basis of aligned corpora to compare descriptions of two or more languages, attempting at a formulation of procedures that would serve as sources for transfer components of translation systems or, as soon as the large multilingual lexical systems such as Wordnet become effectively usable, even as sources for the construction of an interlingua helping translate among whole groups of languages;
(iv) for all such and similar goals one of the most important ingredients is the search for suitable combinations of structural and statistically based procedures of most different kinds and levels, starting from an adequate linguistic background of a POS system with disambiguation; however, it is important to see the typologically determined differences between languages: if E. forms such as *give* (vs. *gives* or *gave*) are classified just as basic verb forms, without distinguishing their values of person and number, then the large set of tags used for a language with a rich morphology (as e.g. Czech, Russian, etc.) gives a much richer set of data (among which then

different ambiguities between morphemic cases and verb forms cause many difficulties); morphemic disambiguation is to be accompanied by procedures of syntactic annotation having their automatic and intellectual parts, the former being partly of a structural nature and partly stochastic.

4 A Concrete Example of a Linguistically Based Annotation Scheme: Prague Dependency Treebank

Prague Dependency Treebank (PDT; for an overall characterization see e.g. [2]) is an annotated collection of Czech texts, randomly chosen from the Czech National Corpus, with a mark-up on three layers: (a) morphemic, (b) surface shape ”analytical”, and (c) underlying (tectogrammatical). The current version (the description of which is publicly available on http://ufal.mff.cuni.cz/pdt2.0, with the data themselves available at LDC under the catalog No. LDC2006T01), annotated on all three layers, contains 3,165 documents (text segments mainly from journalistic style) comprising of 49,431 sentences and 833,195 occurrences of tokens (word forms and punctuation marks) - Figure 1 illustrates sentence annotation on three layers.

On the tectogrammatical layer, which is our main concern in the present paper, every node of the tectogrammatical representation (TGTS, a dependency tree) is assigned a label consisting of: the *lexical value* of the word, its ’(*morphological*) *grammatemes*’ (i.e. the values of morphological categories), its *’functors’* (with a more subtle differentiation of syntactic relations by means of *subfunctors*, e.g. ’in’, ’at’, ’on’, ’under’), and the topic-focus articulation (TFA) attribute containing values for *contextual boundness* (for a motivation for the introduction of this value see below Sect. 5.1). In addition, some basic coreferential links (including intersentential ones) are also added. It should be noted that TGTSs may contain nodes not present in the morphemic form of the sentence in case of surface deletions.

Dependency trees on the tectogrammatical layer are projective (unimportant exceptions aside), i.e. for every pair of nodes in which a is a rightside (leftside) daughter of b, every node c that is less (more) dynamic than a and more (less) dynamic than b depends directly or indirectly on b (where *indirectly* refers to the transitive closure of *depend*). This strong condition together with similar conditions holding for the relationship between dependency, coordination and apposition, makes it possible to capture the tectogrammatical representations in a linearized way. Projective trees thus come relatively close to linear strings; they belong to the most simple kinds of patterning.

In the annotation of PDT, we work also with (surface) analytic representation, a useful auxiliary layer, on which the dependency trees include nodes representing the function words and the tree reflects the surface word order. This combination allows for non-projective structures in cases such as *A neighbour came in, who told us this* (with the relative clause dependent on the subject noun). We assume that such cases can be described as surface deviations from the underlying word order (i.e. in a tectogrammatical representation corresponding

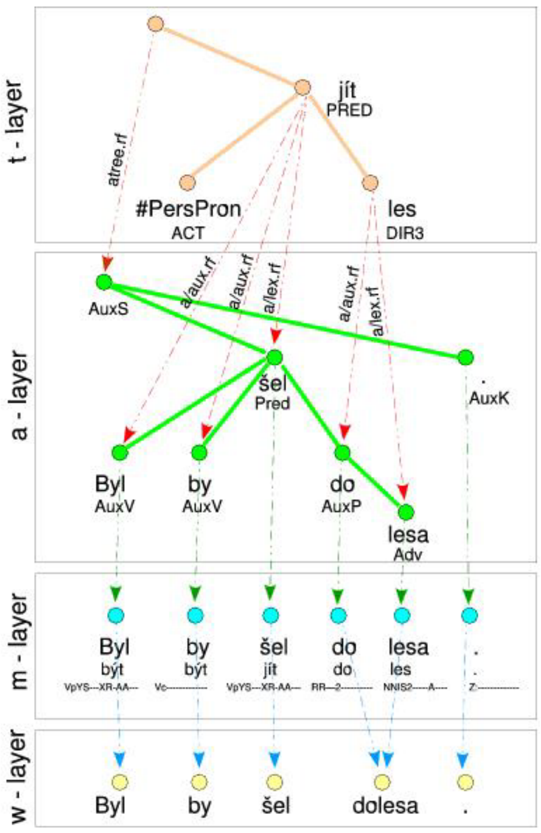


Fig. 1. Annotation layers of the Prague Dependency Treebank

to the example given above, the main verb is not placed between the subject and the dependent clause).

5 Illustration on a Concrete Linguistic Phenomenon

For our discussion of the mutual interlinking of theoretical linguistic description, corpus annotation and computational aspects, we have chosen the linguistic phenomenon of information structure as a universal feature of natural language pertaining to its function as a means of communication expressed in the surface shape of sentences in different ways, mostly dependent on the typological character of the language in question. A description of information structure (be it under the traditional terms of functional sentence perspective, theme-rheme articulation, topic and comment, or, as is the case in the theory we subscribe to, topic-focus articulation, TFA in the sequel) is nowadays regarded as a necessary part of language description in any linguistic theory, though the position within the framework and the detail in elaboration, the scope and depth of the description differs from theory to theory. However, the different treatments of information structure share the underlying idea: a description of the structure reflecting the functioning of language in communication, which is different from the subject-verb-object structure (as described in any formalism)

5.1 The Phenomenon Under Scrutiny: Topic-Focus Articulation

The theoretical framework we subscribe to and on which the above mentioned annotation scenario of PDT is based is the Functional Generative Description (see [3], [4], [5], [6], [7]). This theoretical model works with an underlying syntactic level called tectogrammatics which is understood as the interface level connecting the system of language (cf. the notions of *langue*, linguistic competence, I-language) with the cognitive layer, which is not directly mirrored by natural languages. Language is understood as a system of oppositions, with the distinction between their prototypical (primary) and peripheral (secondary, marked) members. We assume that the tectogrammatical representations of sentences can be captured as dependency based structures the core of which is determined by the valency of the verb and of other parts of speech. Syntactic dependency is handled as a set of relations between head words and their modifications (arguments and adjuncts). However, there are also the relations of coordination (conjunction, disjunction and other) and of apposition, which we understand as relations of a "further dimension". Thus, the tectogrammatical representations are more complex than mere dependency trees.

The tectogrammatical representations reflect also the topic-focus articulation (information structure) of sentence, including the scale of communicative dynamism (underlying word order) and the dichotomy of contextually bound (CB) and non-bound (NB) items, which belong primarily to the topic and the focus, respectively. The scale is rendered by the left-to-right order of the nodes; in the surface structure of the sentence, the most dynamic item, i.e. focus proper, is indicated by a specific (falling) pitch and not necessarily by the word order.

The core of a tectogrammatical representation is a dependency tree the root of which is the main verb. Its direct dependents are arguments (primarily obligatory), i.e. Actor, Objective (Patient), Addressee, Origin and Effect, and adjuncts (of location and direction, time, cause, manner, and so on). Actor primarily corresponds to a cognitive (intentional) Agentive (or Experiencer, i.e. Bearer of a state or process). If the valency frame of a verb contains only a single participant, than this participant is its Actor, even though (in marked cases) it corresponds to a cognitive item that primarily is expressed by some other participant.

In a tectogrammatical representation, there are no nodes corresponding to the function words (or to grammatical morphs). Correlates of these items (especially of prepositions and function verbs) are present there only as indices of node labels: the syntactic functions of the nodes (arguments and adjuncts) are rendered as functors and subfunctors, and the values of their morphological categories (tense, number, and so on) have the forms of grammatemes.

In annotating texts from the Czech National Corpus in the frame of the project of the Prague Dependency Treebank, we work with several specific deviations from theoretically conceived TRs described above. The most important of these deviations is that the tectogrammatical tree structures (TGTSs) we work with in PDT differ from TRs in that they have the form of trees even in cases of coordination; this is made possible by the coordinating conjunctions being handled as specific nodes (with a specific index).

In terms of the communicative function of language, an adequate explanation of information structure of the sentence may be based on the relation of *aboutness*: the speaker communicates something (the Focus of the sentence) about something (the Topic of the sentence), schematically: F(T): the Focus holds about the Topic F(T): negation: (in the prototypical case) the Focus does not hold about the Topic

A supportive argument for such a treatment is offered by the discussions on the *kinds of entailments* as opened by [8](esp. p. 173ff.), who distinguishes a formal logical relation of entailment and a formal logical relation of *presupposition.* He illustrates this distinction on the analysis of the sentence (1a): according to Strawson, (1a) as well as its negation (1b) implies (2). If John's children were not asleep, the sentence (1a) would be *false*; however, if John did not have children, then (1a) as well as its negation (1b) would not be false but *meaningless.* Strawson concludes that (2) is a presupposition of (1a) and as such it is not touched by the negation contained in (1b).

(1) a. All John's children are asleep.
    b. All John's children are not asleep.

(2) John has children.

In a similar vein, [9] discusses the classical example (3a) and, most importantly, notices the difference between (4a) and (5a) by saying (p.96) "we might ... have felt a shade more squeamish if we had written (4a) instead of (5a)".

(3) a. The King of France is bald.
    b. The King of France is not bald.

(4) a. The King of France visited the exhibition yesterday.
    b. The King of France did not visit the exhibition yesterday.

(5) a. The exhibition was visited yesterday by the King of France
    b. The exhibition was not visited yesterday by the King of France.

In his analysis of identifying reference in statements, Strawson (p. 98) suggests that a speech episode "He was saying that the King of France visited the exhibition yesterday." might be described as "he was saying what the king of France is like", in which the clause beginning with "what" specifies "the topic of the statement, what it can be said ... to be *about*"; while *what is said about its topic* is eliminated from the description in favour of the interrogative expression". He adds (imprecisely, influenced apparently by his native tongue) that "the placing of an expression at the beginning of a sentence, in the position of grammatical subject, serves, as it were, to announce the statement's topic" (p. 99).

Applying Strawson's considerations to an analysis of (3a) and its negation (3b), we may say that (3a) is *about* the King of France and therefore the King's existence (referential availability) is *presupposed* and entailed also by its negative counterpart (3b); otherwise (3a) would have no truth value, it would be meaningless. The same holds true about (4a). However, no such existential (referential) presupposition is present in (5a). The truth/falsity of (5a) and (5b) does not depend on the referential availability of the entity ”King of France”. These sentences are not about the King of France but about the exhibition; the existence (referential availability) of the King of France is not presupposed.

To describe the difference between the cases such as in (4a) and in (5a), we have introduced([10]; see also the commentary by [11]) a third kind of entailment in addition to *meaning proper* and presupposition, namely the so-called allegation. While (i) meaning proper can be characterized as an assertion A entailed by an assertion carried by a sentence S, the negation of A being entailed by the negation of S, and (ii) *presupposition* as an assertion A entailed by an assertion carried by a sentence S, and also by the negation of S, (iii) an *allegation* is an assertion A entailed by an assertion carried by a sentence S, with which the negative counterpart of S entails neither A nor its negation.

This distinction can be further illustrated by examples (6a) and (8a). Both (6a) and (6b) implies that we were defeated (i.e. (7) is a presupposition of both of them), they are statements about our defeat.

(6) a. Our defeat was caused by John.
    b. Our defeat was not caused by John.

(7) We were defeated.

The situation is different with (8a) and (8b): it is possible to imagine that (8b) can be used in both contexts (9) and (10), which indicates that (7) is an allegation rather than a presupposition of (8a). In terms of the ’aboutness’ relation, (8a) and (8b) are statements about John rather than about the defeat.

(8) a. John caused our defeat.
    b. John did not cause our defeat.

(9) We were defeated because the whole team performed badly.

(10) Though it is true that John has a reputation of a rather bad player, Paul was in a very good shape and we won.

Returning to our presentation of the relation between the communicative function of language and the information structure of the sentence given at the beginning of the preceding section, we can explain the difference between (6a) and (8a) in terms of the scope of negation and the ’aboutness’ relation as reflected by TFA as follows:

(i) in the prototypical case: the scope of negation constituted by the Focus: Focus (F) does not hold of Topic: F(T).
(ii) in a secondary case, the assertion holds about a negative Topic: F(T)

Compare the possible interpretations of (11) implied by the questions (12a) and (12b):

(11) Bert did not come because he was out of money.

(12) a. What about Bert? I am saying about Bert that he did not come because he was out of money
Topic: Bert
Focus: (he) did not come because he was out of money

b. Why didn't Bert come? I am saying about the fact that Bert did not come that this was caused by the fact that he was out of money:
Topic: Bert did not come
Focus: (because) he was out of money

c. Bert came, but for some other reason I am saying about the fact that Bert came (i.e.about his presence) that it was not because he was out of money but because ...
Topic: Bert came
Focus: not because he was out of money

In the interpretation indicated by (12b), the scope of negation is restricted to the Topic part of the sentence; the assertion triggered (on this reading) by the *because*-clause in Focus is not touched by negation (the reason of Bert's not-coming (absence) is ...).

However, there is another possible reading of (11), namely (12c), e.g. if the sentence is followed by: ... *but because he was on his leave of absence.*

Under this interpretation, Bert's being out of money is neither entailed nor negated. The scope of negation again concerns Focus, schematically: F(T). What is in the scope of negation is neither asserted, nor presupposed; the *because*-clause triggers as allegation.

These considerations – in addition to examples of evident semantic differences between sentences such as (15) through (21) quoted below - have led us to the conclusion that TFA undoubtedly is a *semantically relevant* aspect of the sentence as such should be represented at a level of an integrated language description capturing the meaning of the sentence. This level can be understood as the *'highest'* level of the *language description* viewed from the point of view of the hierarchy from *function to form.* The inclusion of TFA into this level can serve well as a starting point for connecting this layer with an interpretation in terms of *intensional semantics* in the one direction and with a description of the *morphemic and phonemic means* expressing TFA in the other direction (see below Sect. 5.1).

The semantico-pragmatic interpretation of sentences (for which the tectogrammatical representations represent a suitable input) may then include an application of Tripartite Structures (Operator - Restrictor - Nuclear Scope), as outlined by B. H. Partee in [7]. Let us briefly recall some of the characteristic sentences discussed there (with their relevant tectogrammatical representations, TRs) and specify (in a maximally simplified notation) which parts of their individual readings belong to the Operator (O), Restrictor (R) and Nuclear Scope (N) of the corresponding tripartite structures. We assume that in the interpretation of a declarative sentence, O corresponds to negation or to its positive counterpart (the assertive modality)[3] or to some other operators such as focusing particles, R corresponds to Topic (T), and N to Focus (F).

(13) a. John sits by the TELEVISION.
b. O ASSERT, R John, N sits by the TELEVISION.
c. O ASSERT, R John sits, N by the TELEVISION.

Sentence (13a) may be analyzed in two ways: ether (i) it conveys an information about John (i.e. John being its Topic and the rest its Focus), or (ii) it conveys an information about John's sitting (i.e. with both John and the verb in the Topic). If the sentence includes a focusing particle such as *only, also, even* etc., the particle occupies its prototypical position in the TR, so that the focus of the particle is identical with the F of the sentence on either reading. If the focusing particle is included in T, its own focus (which differs from the sentence F in such marked cases) does not cross the boundary between the T and the F of the sentences, see (14) in the context indicated in the brackets (and discussed in more detail below as sentence (22)).

(14) (Everyone already knew that Mary only eats vegetables.) If even Paul knew that Mary only eats vegetables, then he should have suggested a different restaurant.

In linguistic literature, many examples have been adduced which indicate that the difference of the meaning between the members of the given pairs of sentences is given by their topic-focus structure, though not always the difference in this structure is being referred to (see ex. (15a, 15b) and (14)). Let us give here just a couple of examples (the original sources of the examples are given in brackets; the capitals denote the assumed position of the intonation centre, which is crucial for the interpretation of the given sentences).

(15) a. Everybody in this room knows at least two LANGUAGES.
b. At least two languages are known by everybody in this ROOM. ([12], [13])

(16) a. Many men read few BOOKS.
b. Few books are read by many MEN. ([14])

(17) a. Londoners are mostly at BRIGHTON.
b. At Brighton, there are mostly LONDONERS. ([15])

(18) a. I work on my dissertation on SUNDAYS.
b. On Sundays, I work on my DISSERTATION.

[3] In the interpretation, we use the ASSERT operator introduced by Jacobs (1984).

(19) a. English is spoken in the SHETLANDS.
b. In the Shetlands, ENGLISH is spoken. ([6])

(20) a. I only introduced BILL to Sue.
b. I only introduced Bill to SUE. ([16])

(21) a. Dogs must be CARRIED.
b. DOGS must be carried. ([17])
c. Carry DOGS. (a warning in London underground, around 2000)
d. CARRY dogs.

We have discussed these and several other sentences in our previous writings on TFA (see the References below) and therefore we present them here without a more detailed analysis, just for a support for our claim that the differences in the surface shape of these sentences have a common denominator, i.e. that they are due to the differences of the means of expression of an underlying phenomenon of TFA. These means of expression may concern (a) the surface *order of words*, (b) the sentence prosody, (c) the syntactic constructions, and (d) morphemic means. It goes without saying that this is an open list, especially if languages belonging to other than the Indoeuropean type are taken into account.

The most frequently and extensively discussed means of expression of the information structure is the order of words; as a matter of fact, in some approaches, the differences in the information structure are even identified with the differences in the order of words in the surface shape of the sentence. It has been sometimes claimed that with respect to the order of elements, the presence of a quantifying expression is crucial; as the examples quoted above demonstrate, there are no quantifiers present in (18) and (19) and yet the difference in meaning cannot be excluded. (18b) is about my work on dissertation, and may be true also in a context when I am preoccupied also by other things on Sundays, while this is not the case in (18a) which is about Sundays and indicates that my (only) preoccupation on Sundays in working on my dissertation. Such an "exhaustive listing" (for this notion, see [18], esp. p. 307) is also implied by (19a), and the sentence cannot stand alone e.g. in a textbook on geography since it would not convey a true information (it brings a false information about English), while (19b) is true about the Shetlands rather than about English.

(b)The order of words in the surface shape of the sentence might be the same and yet the sentences acquire different information structure and differ in their meanings which is reflected by *sentence prosody* including the placement of the intonation center. This holds e.g. about sentences in (20) and (21) above. Sentences (20a) and (20b) differ in their truth conditions: leaving aside the possible ambiguities of the placement of the verb within topic or focus, (20a) can be uttered in a situation when the speaker did not introduce other people to Sue except for Bill, this is not the case of (20b): the speaker may have introduced other people to Sue but the only person he introduced Bill to, was Sue.

M.A.K.Halliday quotes in his pioneering analyses of the relations between grammar and intonation ([17]) the example given above as (21a) and (21b). (21a) is a warning at the bottom of the escalators in London underground and Halliday jokingly remarks that if pronounced as in (21bb), it would lead to a false assumption that (on the escalator) everybody has to carry a dog. His warning apparently has not reached the ears/eyes of the builders of the new underground stations around 2000, since these stations have been equipped by a shortened warning the natural pronunciation of which would be as indicated in (21c). This, however, would lead to the same funny interpretation as (21b) rather than to the intended interpretation (21a), unless the inscription is pronounced with the placement of the intonation centre (unusual for English) at the beginning (as in (21d)).

The respect to the prosodic expression is most perspicuously reflected in the above mentioned doctoral dissertation on *'association with focus'* by [16]. Rooth postulates the so-called 'association with focus" connected with E. particles (called focussing particles or focalizers) such as '*only*', '*even*', '*also*', etc and its assumed realization by a pitch accent (typically with a falling intonation contour).The question arises whether these particles always stipulate association with a focussed element in their scope. As demonstrated by [7], an association with the Focus of the sentence in not necessarily the case, see (14) above reproduced here as (22).

(22) a. Everyone already knew that Mary only eats vegetables.
b. If even PAUL knew that Mary only eats vegetables, then he should have suggested a different restaurant.

There are two 'focalizers' in B, namely only which introduces material repeated from the previous sentence (sometimes called a second occurrence focus), and even associated with *Paul*, which carries the intonation center. [19] observed that the acoustic realization of "second- occurrence focus" is different from the 'regular' focus; [20] refer to her analysis and claim that the second occurrence focus is not only marked differently from the 'regular' focus but also differs acoustically from the non-focused expressions. This confirms the suggestions given in [7]: the authors differentiate focus of the focussing particle (its scope) from the Focus of the sentence (i.e. the part of the sentence which is about its Topic) and illustrate this distinction by (23a) and its interpretation in terms of the tripartite structure in (23b), within the context indicated in the brackets.

(23) a. What did even PAUL realize? Even Paul realized that Jim only admired MARY.
b. O ASSERT, R (O even, R realized, N Paul), N (O only, R Jim admired, N Mary)

When deciding on the status of the given elements of the sentence in its TFA, not only the position of the intonation center should be taken into account but the whole intonation contour of the sentence (its F0 characteristics) should be considered. Such an evaluation of the F0 characteristics has led us to introduce the notion of "contrastive topic" (see e.g. [21], [22]).

As Firbas in [23] noticed, it is not always the case that the most dynamic element of Focus is to be prosodically marked. (24) is his example of an 'automatic placement' of the intonation center at the end of the sentence even if the subject is (a part) of the focus .

(24) A boy came into the room.

Since the grammatically fixed word order of English does not always allow to linearly order the elements of a sentence as to reflect the information structure of the sentence (and passivization as in (15) and (16) above or some other syntactic restructuring cannot be applied), the use of *italics* in written English can be used to denote the position of IC. This has been observed by Alena Skaličková in the 1970's and her observation reoccurred in a paper by [24], analyzing the use of italics to mark focus in English translations of Spanish and Portuguese original texts.

(c) Among the specific *syntactic constructions* as the means of expression of TFA in English, the *it*-clefts (in contrast to the pseudo-clefts (as *wh*-clefts) are often referred to, which make it possible to 'prepose' the focussed element and thus to give it some kind of prominence. The rest of the sentence is then understood as being in a kind of 'shadow', backgrounded. The 'preposing' of the focused element is prototypically accompanied by the shift of the intonation center to the clefted element, see (25a).

(25) a. It was JOHN who talked to few girls about many problems.
b. With few girls talked about many problems
S málo děvčaty mluvil o mnoha problémech
John-Nominative.
HONZA.

Cleft constructions may serve also as an additional support for the view that not only the division of the sentence into its Topic and Focus, but also the degrees of communicative dynamism (underlying word order, see below in Sect. 5.2) as such play their role in the semantic interpretation of the sentence.

(26) a. It was JOHN who talked about many problems to few girls.
b. About many problems talked with few girls
O mnoha problémech mluvil s málo děvčaty
John-Nominative.
HONZA.

The (preferred) interpretation of (25a) indicates that there was a group of few girls with which John talked about many problems, not necessarily the same set of many problems; the (preferred) interpretation of (26a) suggests that there was a (single) set of many problems about which John talked with few girls (not necessarily with a single group of girls).

(d) Notorious examples of *morphemic* means expressing the TFA are the Japanese particles *ga* and *wa* discussed in linguistic literature since Kuno's ([18]; [25]) pioneering analysis of the function of these particles in the information structure of Japanese (most recently, the thematic function of 'wa' was analyzed e.g. by [26]).

There are many other examples of languages where morphemics serves as (one of the means of expression) of information structure quoted in linguistic literature up to now. Let us only mention two of them discussed by [27](p. 177) referring also to [28]. Information structure is expressed obligatorily and by using morphological means in Yukaghir, a Paleo-Asiatic language ([29]). There are three series of forms for each transitive verb there (distinguished from one another by the presence or absence of personal inflection, by morphological exponents, and by the presence or absence of certain prefixes) which are used whether the rheme-component coincides with the subject of the verb, its object, or the verb itself, respectively. In addition, a suffix is attached to the subject or object under conditions that pertain to the distribution of the rheme. In Tagalog, an Indonesian language, the theme of the sentence is distinguished by means of certain particles (articles) and word order; the syntactic roles of the given participants are indicated by an appropriate from of the verb ([30]).

5.2 From the Theory to an Annotation Scheme

For the theoretical description of TFA, the crucial issue is which basic oppositions are to be captured. In the approach of the Functional Generative Description to TFA, which we subscribe to, the basic opposition is seen in the opposition of contextual boundness. This opposition is represented in the underlying structure: for every autosemantic lexical item in the sentence (i.e. for every node of its tectogrammatical representation) it is specified whether it is (a) contextually bound (*cb*), i.e. an item presented by the speaker as referring to an entity assumed to be easily accessible by the hearer(s), more or less predictable, readily available to the hearers in their memory, or (b) contextually non-bound (*nb*), i.e. an item presented as not directly available in the given context, as cognitively 'new'. While the characteristics 'given' and 'new' refer only to the cognitive background of the distinction of contextual boundness, the distinction itself is an opposition understood as a grammatically patterned feature, rather than in the literal sense of the term. This point is illustrated by (27): both Tom and his friends are 'given' by the preceding context (indicated here by the preceding sentence in the brackets), but their linguistic counterparts are structured in the given sentence as non-bound (which is reflected in the surface shape of the sentence by the position of the intonation center).

(27) (Tom entered together with his friends.) My mother recognized only HIM, but no one from his COMPANY.

In the prototypical case, the head verb of the sentence and its immediate dependents (arguments and adjuncts) constitute the Topic of the sentence if they are contextually bound, whereas the Focus consists of the contextually non-bound items in such structural positions (and of the items syntactically subordinated to them). Also the semantically relevant scopes of focus sensitive operators such as *only, even*, etc. can be characterized in this way.

The bipartition of the sentence into the Topic and Focus (reflecting the aboutness relation as discussed above in Sect. 5.1) can then be specified by the following set of the rules determining the appurtenance of a lexical occurrence to the Topic (T) or to the Focus (F) of the sentence (see [31]; [6], pp. 216ff)

(a) the main verb (V) and any of its direct dependents belong to F iff they carry index *nb*;
(b) every item i that does not depend directly on V and is subordinated to an element of F different from V, belongs to F (where "subordinated to" is defined as the irreflexive transitive closure of "depend on");
(c) iff V and all items k_j directly depending on it carry index *cb*, then those items k_j to which some items l_m carrying f are subordinated are called 'proxy foci' and the items l_m together with all items subordinated to one of them belong to F, where $1 \leq j, m$;
(d) every item not belonging to F according to (a) - (c) belongs to T.

There are two reasons why to distinguish the opposition of contextual boundness as a primary (primitive) one and to derive the Topic-Focus bipartition from it. First, and most importantly, the Topic/Focus distinction exhibits – from a certain viewpoint - some recursive properties, exemplified first of all in sentences which contain embedded (dependent) clauses. The dependent clause D functions as a sentence part of the clause containing the word on which D depends, so that the whole structure has a recursive character; one of the questions discussed is whether the T-F articulation should be understood as recursive, too. Several situations arise: (i) one of the clauses may be understood as the F of the whole sentence, though each of the clauses displays a T-F articulation of its own; (ii) in a general case the boundary between T and F may lie within one of the clauses.

The second argument is related to the fact that Topic/Focus bipartition cannot be drawn on the basis of an articulation of the sentence into constituents but requires a more subtle treatment. In early discussions on the integration of the topic-focus articulation into a formal description of grammar, the proponents intended to specify this aspect of the structure of the sentence in terms of the type of formal description they subscribed to. Within the framework of generative transformational grammar, [32] (p. 205) defined focus as "a phrase containing the intonation center", i.e. in terms of constituency (phrase-structure) based description (see also Jackendoff 1972, p. 237). Such a description served as a basis also for several studies on the relationship between syntax and semantics (e.g. [33]; [34]; [35]): the boundaries between topic and focus or some more subtle divisions were always supposed to coincide with the boundaries of phrases. Sgall and his followers (see already [15]) work within a framework of dependency grammar and define the boundary between the two parts on the basis of syntactic dependency, of the opposition of contextual boundness and of the left-to-right order of nodes. The boundary between Topic and Focus can then be characterized as intersecting an edge between a governor and its dependent (the latter may be a single node or a subtree), with the provision that whatever is to the right of the given dependent in the tectogrammatical dependency tree, belongs to the Focus, the rest to the Topic (see Sgall's definition above).

However, the definition of Focus (and of presupposition, in Chomskyan terms) as a phrase is untenable since it is not always possible to assign the focus value to a part of the sentence that constitutes a phrase. This claim is supported by examples as those adduced by [36]: in the given context, the Focus of the sentence is *for a week to Sicily*, which would hardly be specified as a constituent under the standard understanding of this notion. These examples, however, bring no difficulties for a dependency-based description.

(28) John went for a week to Sicily. (He didn't't go only for a weekend to his parents.)

It was convincingly argued by [37]; [38]; [39] that it is advisable to postulate a common structure for accounting both for the syntactic structure of the sentence as well as for its information structure. For that purpose, he proposes a modification of categorial grammar, the so-called combinatory categorial grammar. A syntactic description of a sentence ambiguous as for its information structure should be flexible enough to make it possible to draw the division line between Topic and Focus also in other places that those delimiting phrases; in [38] (p.5), the author claims that e.g. for the sentence *Chapman says he will give a policeman a flower* his "theory works by treating strings like *Chapman says he will give, give a policeman*, and *a policemen a flower* as grammatical constituents" and thus defining "a constituent" in a way that is different from the "conventional linguistic wisdom". In other words, Steedman proposes to work with non-standard constituents, as can be illustrated by (29) with the assumed intonation center at the last element of the sentence: the division of (29) into Topic and Focus is ambiguous because the verb may belong either to the topic or to the focus part of the sentence.

(29) Fred ate the BEANS.

The representation of such an ambiguity in a dependency framework like that of the Praguian Functional Generative Description causes no difficulty. In case the root of the tree (the verb) is cb, then it depends on the *cb/nb* feature of its dependents whether *Fred ate* or just *ate* are the elements of the Topic (answering the question *What did Fred eat?*, or *Who did eat what?*, respectively. If the verb is *nb*, then again two divisions are possible: either the whole sentence is the Focus (*What happened?*), or the verb and the object are the elements of the Focus (*What did Fred do?*). In the underlying tree structure, the *cb* nodes depend on the verb from the left, the *nb* nodes from the right. A division line between Topic and Focus is then drawn as characterized above.

In (29), we assumed the (normal) placement of the intonation center on the object *beans*. However, as also discussed by Steedman, the sentence may have different intonation patterns, and this may reduce its ambiguity: if the intonation center is on *Fred*, then *Fred* is the sentence Focus and the rest is the Topic (*Who ate the beans? Fred.*). If the intonation center is on the verb, then only the verb is the Focus the rest being the Topic (*What did Fred do with the beans? (He) ate (them).*) This again can be easily captured in the dependency representation of the meaning of the sentence by the assignment of the primary opposition of *cb/nb* nodes.

The above considerations of the theoretical status of TFA within a formal descriptive framework have led us to introduce, in the annotation scheme of the underlying layer of PDT, a specific TFA attribute as a part of the annotation of each node of the tectogrammatical tree structure, with the choice of one of the following three values: t for a non-contrastive contextually bound node, c for a contrastive contextually bound node, and f for a contextually non-bound node.

5.3 From the Annotation Scheme to the Theory

Any modern linguistic theory has to be formulated in a way that it can be tested by some testable means. One of the ways how to test a theory is to use it as a basis for a consistent annotation of large language resources, i.e. of text corpora. Annotation may concern not only the surface and morphemic shapes of sentences, but also (and first of all) the underlying sentence structure, which elucidates phenomena hidden on the surface although unavoidable for the representation of the meaning and functioning of the sentence, for modeling its comprehension and for studying its semantico-pragmatic interpretation. One of the aims the PDT was designed for was to use it as a testbed for the theoretical assumptions encapsulated in the Functional Generative Description as briefly sketched in Sect. 5.1 above.

As mentioned in Sect. 5.2, one of the hypotheses of the TFA account in FGD concerned the possibility of the derivation of the bipartition of a sentence into its Topic and Focus on the basis of the feature of contextual boundness of the individual lexical items contained in the sentence. To illustrate the hypothesis on a PDT example, let us take the Czech sentence (30) and its (very simplified) annotation on the tectogrammatical layer (in the preferred reading) as given in Figure 2.

(30)	Nenadálou	finanční	krizi	podnikatelka	řešila
	The sudden	financial	crisis(Acc.)	the entrepreneur(Nom.)	solved
	jiným	způsobem.			
	by other	means.			

The application of the rules quoted in Sect. 5.2 gives the following result: Topic: *Nenadálou finanční krizi podnikatelka* [the sudden financial crisis the enterpreneur] Focus: *řešila jiným způsobem* [solved by other means]

The implementation of an algorithm based on the quoted rules has led to a differentiation of five basic types of Focus and it significantly supported the hypothesis that in Czech the boundary between T and F is signalized by the position of the verb in the prototypical case (the boundary between T and F: immediately before the verb in 95% of the cases) and it has also been confirmed that the TFA annotation leads to satisfactory results even with rather complicated "real" sentences in the corpus.

Another hypothesis that has already been tested on our annotated corpus concerns the order of elements in the Focus. It is assumed that in the focus part of the sentence the complementations of the verb (be they arguments or adjuncts)

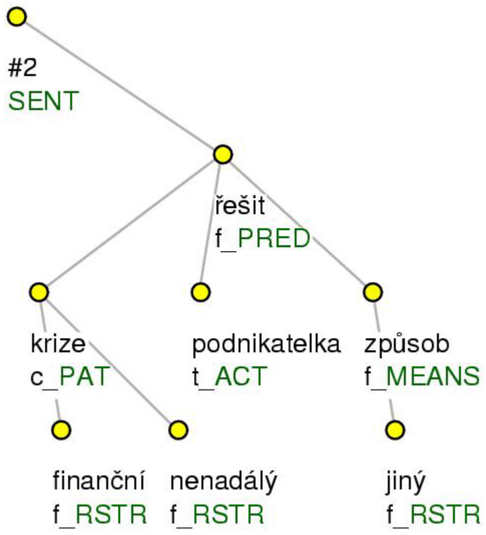


Fig. 2. The preferred TGTS of sentence (30)

follow a certain canonical order in the TRs, the so-called systemic ordering (not necessarily the same for all languages). In Czech, also the surface word order in Focus corresponds to the systemic ordering in the prototypical case.

For Czech, the following systemic ordering is postulated (see [6]): Actor – Time:*since-when* – Time:*when* – Time: *how-long* – Time:*till-when* – Cause – Respect – Aim – Manner – Place – Means – Dir:*from-where* – Dir:*through-where* – Addressee – Origin – Patient – Dir:*to-where* – Effect.

Systemic ordering as a phenomenon is supposed to be universal; however, languages may differ in some specific points: the validity of the hypothesis has been tested with a series of psycholinguistic experiments (with speakers of Czech, German and English); for English most of the adjuncts follow Addressee and Patient ([40]). However, PDT offers a richer and more consistent material; preliminary results have already been achieved based on (a) the specification of F according to the rules mentioned above, (b) the assumed order according to the scale of systemic ordering (functors in TGTS), and (c) the surface word order ([41]). These results have led to a fruitful reconsideration and possible modification of the theoretical assumptions.

A general assumption common to any postulation of a deep (underlying) layer of syntactic description is the belief that languages are closer to each other on that level than in their surface shapes. This idea is very attractive both from the theoretical aspects as well as from the point of view of possible applications in the domain of natural language processing: for example, a level of language description considered to be ”common” (in its structure, not of course in their repertoire of features) to several (even if typologically different) languages might serve as a kind of ”pivot” language in which the analysis of the source and the synthesis of the target languages of an automatic translation system may meet.

With this idea in mind, it is interesting (again, both from the theoretical and the applied points of view) to design and test an annotation scheme by means of which parallel text corpora can be annotated in an identical or at least easily comparable way.

These considerations have motivated one of our current project in which the PDT scenario (described above in Sect. 3) is being applied to English texts in order to find out whether such a task is feasible and if the results may be used for a build-up of a machine translation system (or other multilingual systems).

To this end, a parallel Czech and English corpus (Prague Czech-English Dependency Treebank, see [42]) is built, with the intention to apply of the original annotation scheme designed for the annotation of Czech sentences on the tectogrammatical layer to English parallel texts.

It is well known from classical linguistic studies (let us mention here – from the context of English-Czech contrastive studies – the writings of Czech anglicists Vilém Mathesius, Josef Vachek and Libuše Dušková) that one of the main differences between English and Czech concerns the degree of condensation of the sentence structure following from the differences in the repertoire of means of expression in these languages: while in English this system is richer (including also the forms of gerund) and more developed (the English nominal forms may express not only verbal voice but temporal relations as well), in Czech, the more frequent means expressing the so called second predication (and sometimes the only possible one, see (32) below) is a dependent clause (see [43], p. 542 ff.).

It is no wonder then that in our project, secondary predication has appeared as one of the most troublesome issues. Therefore, we devote our attention to two typical nominal forms serving for the expression of secondary predication in English and look for their adequate representation on the tectogrammatical layer of PDT, namely (1a) infinitive (see (31)) and (2) gerunds (see (32)). The leading idea of our analysis is that we aim at a representation that would make it possible to capture synonymous constructions in a unified way (i.e. to assign to them the same TGTS, both in the same language and across languages) and to appropriately distinguish different meanings by the assignment of different TGTSs.

(31) Jan slyší Marii(Acc.) plakat(Inf.).
John hears Mary cry.

(32) Jan očekává, že Marie odejde.
John expects Mary to leave.
or:
John expects that Mary leaves.

(33) Viděl jsem, že jeho úspěch roste.
(I) saw that his success grows.

I saw his success growing.

This is still a work in progress ([44]) but the preliminary investigations in this direction and a consistent effort to confront the application of the PDT annotation on both Czech and English as typologically different languages scenario have brought several interesting stimuli for the theoretical considerations.

6 Conclusion

Our experience has convinced us that a corpus annotation on an underlying level is a feasible task, not only if the predicate – argument structure is to be captured but also with respect to the information structure of the sentence reflecting the communicative function of language, which indicates **what we are talking about and what we are saying about it**. To this aim strong **interconnections** between theoretical research and corpus annotation efforts as well as a due regard to computational aspects of the enterprise are necessary and mutually enriching. Such cooperation is also fruitful for **applications** such as automatic and machine assisted translation on different layers of complexity, communication with intelligent systems, information retrieval, grammar checking and so on.

Acknowledgments

The author gratefully acknowledges the most useful comments and suggestions given by Jan Hajič, Jarmila Panevová, Petr Sgall, Barbora Vidová Hladká and Šárka Zikánová after having read the pre-final version of the manuscript. The present paper has been written under the support of the grant MSM0021620838 of the Ministry of Education of the Czech Republic, and the EU project Euromatrix (IST-FP6-034291-STP).

References

1. Uszkoreit, H.: New Chances for Deep Linguistic Processing. In: Huang, C.R. (ed.) Frontiers in Computational Linguistics, Shangwu Press Beijing (2004)
2. Hajič, J.: Building a Syntactically Annotated Corpus: The Prague Dependency Treebank. In: Hajičová, E. (ed.) Issues of Valency and Meaning. Studies in Honour of Jarmila Panevová, pp. 106–132. Karolinum, Charles University Press, Prague, Czech Republic (1998)
3. Sgall, P.: Zur Frage der Ebenen im Sprachsystem. In: Travaux linguistiques de Prague 1, pp. 95–106 (1964)
4. Sgall, P.: Generative bschreibung und die ebenen des sprachsystems. Zeichen und System der Sprache III, 225–239 (1966)
5. Sgall, P.: Generativní popis jazyka a česká deklinace. Academia, Prague (1967)
6. Sgall, P., Hajičová, E., Panevová, J.: The Meaning of the Sentence in Its Semantic and Pragmatic Aspects. Reidel Publishing Company, Academia and Dordrecht, Prague (1986)
7. Hajičová, E., Partee, B.H., Sgall, P.: Topic-focus articulation, tripartite structures, and semantic content. Kluwer, Amsterdam (1998)
8. Strawson, P.P.: Introduction to Logical Theory. Methuen, London (1952)

9. Strawson, P.P.: Identifying Reference and Truth Values. Theoria, 96–118 (1964)
10. Hajičová, E.: On presupposition and allegation. In: Partee, B.H., Sgall, P. (eds.) Discourse and Meaning, pp. 99–122. John Benjamins Publ. House, Amsterdam (1996)
11. Partee, B.H.: Allegation and Local Accommodation, pp. 65–86 (1996)
12. Chomsky, N.: Aspects of the Theory of Syntax. The MIT Press, Cambridge (1965)
13. Chomsky, N.: Syntactic Structures. Mouton, The Hague (1957)
14. Lakoff, G.: On Generative Semantics, pp. 232–296 (1971)
15. Sgall, P.: Functional Sentence Perspective in a Generative Description of Language. The Prague Bulletin of Mathematical Linguistics, 203–225 (1967)
16. Rooth, M.: Association with Focus. PhD thesis, Univ. of Massachusetts, Amherst (1985)
17. Halliday, M.A.K.: Intonation and Grammar in British English. Mouton, The Hague (1967)
18. Kuno, S.: Functional sentence perspective. Linguistic Inquiry, 296–320 (1972)
19. Bartels, C.: Acoustic correlates of second occurrence focus: Towards and experimental investigation. In: Kamp, H., Partee, B.H. (eds.), pp. 11–30 (1997)
20. Beaver, D., Clark, Z.B., Flemming, E.: T.Jaeger, F., Wolters, M.: When semantics meets phonetics: Acoustical studies of second-occurrence focus. Language, 245–276 (2007)
21. Hajiéová, E., Sgall, P.: Degrees of Contrast and the Topic-Focus Articulation. In: Steube, A. (ed.) Information Structure - Theoretical and Empirical Aspects, pp. 1–13. Walter de Gruyter, Berlin (2004)
22. Veselá, K., Peterek, N., Hajiéová, E.: Topic-Focus Articulation in PDT: Prosodic Characteristics of Contrastive Topic. The Prague Bulletin of Mathematical Linguistics, 5–22 (2003)
23. Firbas, J.: Functional Sentence Perspective in Written and Spoken Communication. Cambridge University Press, Cambridge (1992)
24. Gaby, S.: The use of italics as stylistic devices marking information focus in English translation. In: Proceedings of the Corpus Linguistics Conference, Birmingham, p. 55 (2007)
25. Kuno, S.: The structure of the Japanese language. Cambridge, Mass (1973)
26. Fazuo, F.: A consideration of the thematiser 'wa' (in Japanese), pp. 147–160 (2003)
27. Novák, P.: Remarks on devices of functional sentence perspective. Papers on Functional Sentence Perspective, pp. 175–178. Academia, Prague (1974)
28. Dahl, O.: Topic and comment: a study in Russian and general transformational grammar. Slavica Gothoburgensia 4, Göteborg (1969)
29. Krejnovič, E.A.: Jukagirskij jazyk. Leningrad, Moscow (1958)
30. Bowen, D.: Beginning Tagalog. Berkeley and Los Angeles (1965)
31. Sgall, P.: Towards a Definition of Focus and Topic. The Prague Studies of Mathematical Linguistics, 173–198 (1981)
32. Chomsky, N.: Deep Structure, Surface Structure and Semantic Interpretation, pp. 193–216 (1971)
33. Schmerling, S.: Aspects of English Sentence Stress. University of Texas Press, Austin, Texas (1971)
34. Selkirk, E.: Phonology and Syntax: The Relation between Sound and Structure. MIT Press, Cambridge (1984)
35. Selkirk, E.: Sentence Prosody: Intonation, Stress and Phrasing. In: Goldsmith, A. (ed.) Handbook of Phonological Theory, pp. 550–569. Blackwell, London (1995)
36. Hajièová, E., Sgall, P.: Topic and Focus in Transformational Grammar. Papers in Linguistics, 3–58 (1975)

37. Steedman, M.: Structure and Intonation. Language, 260–296 (1991)
38. Steedman, M.: Surface Structure and Interpretation. The MIT Press, Cambridge (1996)
39. Steedman, M.: Information structure and the syntax-phonology interface. Linguistic Inquiry, 649–689 (2000)
40. Sgall, P., et al.: Experimental Research on Systemic Ordering. Theoretical Linguistics, 97–239 (1995)
41. Zikánová, v.: What Do the Data in PDT Say about Systemic Ordering in Czech? The Prague Bulletin of Mathematical Linguistics, 39–46 (2006)
42. Cuřín, J., et al.: The Prague Czech-English Dependency Treebank 1.0 CD-ROM (2004) CAT: LDC2004T25, Linguistic Data Consortium (2004)
43. Dušková, L.: Mluvnice současné angličtiny na pozadí češtiny. Academia, Prague (1988)
44. Cinková, S., et al.: The tectogrammatics of English: on some problematic issues from the viewpoint of Prague Dependency Treebank (in preparation)

Trusting Politicians' Words (for Persuasive NLP)

Marco Guerini, Carlo Strapparava, and Oliviero Stock

FBK-irst, I-38050, Povo, Trento, Italy
{guerini,strappa,stock}@itc.it

Abstract. This paper presents resources and lexical strategies for persuasive natural language processing. After the introduction of a specifically tagged corpus of political speeches, some forms of affective language processing in persuasive communication and prospects for application scenarios are provided. In particular *Valentino*, a prototype for valence shifting of existing texts, is described.

1 Introduction

In order to automatically produce and analyze persuasive communication, specific resources and methodologies are needed. For persuasive NLP we built a resource called CORPS that contains political speeches tagged with audience reactions. A key role in persuasive communication is played by affects: we have focused on lexical choice and we present here a tool for modifying existing textual expressions towards more positively or negatively valenced versions, as an element of a persuasive system.

The paper is structured as follows: Section 2 gives an overview of key concepts connected to persuasion and briefly describes the state of the art in related areas. Section 3 describes the resources we built for statistical acquisition of persuasive expressions. Finally, Section 4 describes how this approach can be used for various persuasive NLP tasks, while Section 5 presents the *Valentino* prototype, built upon the resources we presented.

2 Persuasion, Affect and NLP

According to Perelman and Olbrechts-Tyteca [1], persuasion is a skill that human beings use - in communication - in order to make their partners perform certain actions or collaborate in various activities. Here below we introduce some related key concepts.

Argumentation and Persuasion. In AI the main approaches focus on the argumentative aspects of persuasion. Still, argumentation is considered as a process that involves "rational elements", while persuasion includes also elements like emotions. In our view, a better distinction can be drawn considering their different foci of attention: while the former focuses on message correctness (its being a valid argument) the latter is concerned with its effectiveness.

Natural Argumentation. The recent area of natural argumentation [2] tries to bridge argumentation and persuasion by focusing, for example, on the problem of the adequacy - effectiveness - of the message.

Emotions and Persuasion. Since persuasion includes non-rational elements as well, it is a "superset" of argumentation, but this does not rule out that there is a role for emotion within argumentation (Miceli *et al.* [3]): through arousal of emotions or through appeal to expected emotions. Indeed, emotional communication has become of increasing interest for Persuasive NL Generation.

Rhetorics. The study of how language can be used effectively. This area of studies concerns the linguistic means of persuasion (one of the main means, but not the only one). This is the area we are focusing on in this paper.

Irony. It refers to the practice of saying one thing whilst meaning another. Irony occurs when a word or phrase has a surface meaning, but another contradictory meaning beneath the surface. Irony is a widely used rhetorical artifice, especially in advertisement.

Past works on persuasion and NLP has focused mainly on text generation (a notable exception being *Araucaria* [4]). Persuasive text generation deals with the production of texts that are meant to affect the behavior of the receiver. For example STOP, one of the best known NLG systems [5], uses domain specific rules, based on expert knowledge acquisition for the clinical smoking domain [6]. *Promoter* instead [7] uses strategies gathered from different persuasive theories and subsumed in a general planning framework. Other persuasive NLG systems are more argumentation oriented. In these cases "theoretical expert knowledge" is used (e.g. Toulmin [8], Perelman and Olbrechts-Tyteca [1], Walton [9]). NAC [10], for example, is concerned with the abstract form of the unfolding of the argument - strategic planning -. The system presented by Reed *et. al.* [11] uses two modules, Argument Structure (strategic planning) and Eloquence Generation (tactical planning), leaving the problem of message effectiveness to the latter module. The PORTIA [12] and ARGUER [13] systems focuses on dialogical aspects of argumentation. PORTIA uses Walton's argumentation schemata, extended to formalize a-rational aspects of persuasion. ARGUER is also based on argumentation schemata to detect attack or support relations among participants' moves.

Since emotional reasoning is usually performed in order to modify/increase the impact of the message, affective NLP is strictly connected to persuasive NLP. An annotated bibliography on affective NL generation can be found in [14]. de Rosis and Grasso [15] focus on the technological aspects for an affectively "richer" NL production. Their model uses plan operators - for text structuring - combined with rule based heuristics for revising both strategic and tactic planning. Carofiglio and de Rosis [16] instead use a dynamic belief network for modeling activations of emotional states during dialogical interactions. This model of emotional activation is inserted in an argumentation framework.

Opinion mining is a topic at the crossroads of information retrieval and computational linguistics concerned with the opinions expressed in a document. Recent research has tried to automatically identify whether a term has a positive or a negative connotation (see for example [17] and [18]). In [17] a method for feature extraction that draws on an existing unsupervised method is introduced. The work in [18] presents methodologies that use a wide range of features, including new syntactic features, for opinion recognition. Opinions, once extracted, must be summarized (in case) and presented to the user. In [19] the authors argue that an effective method for summarizing evaluative arguments must synthesize sentence extraction-based approaches and language generation-based approaches. Even though opinion mining deals with texts that are meant to persuade its focus is on polarity (valence) recognition for evaluative language retrieval. Instead persuasive expression mining deals with the extraction of pieces of text that are meant to persuade, regardless of their possible evaluative use.

3 Aims and Resources

Authors such as Radev and McKeown [20] rely on automatic acquisition of sentences mapped on their functional description, to overcome the problem of simple canned texts extraction. In this paper we adopt persuasive expression mining techniques and refinement as a component for persuasive NLP systems in an unrestricted domain. As for emotions, we restrict our focus on valenced expressions (i.e. those that have a positive or negative connotation). For us the task of producing affective expressions, as a component of persuasive systems, involves changing appropriately the valence of existing expressions. We collected specific resources aimed at persuasion[1]:

– A CORpus of tagged Political Speeches (CORPS), as examples of long and elaborated persuasive texts
– A Corpus of labeled advertising or political slogans (*SloGun*), as examples of short, high impact, sentences
– A resource containing terms gathered by semantic similarity and ordered by valence (Ordered Vectors of Valenced Terms - OVVT).

The resources we focus on in this paper are CORPS and OVVTs.

3.1 CORPS

In collecting this corpus we relied on the hypothesis that tags about public reaction, such as `APPLAUSE`, are indicators of hot-spots, where persuasion attempts succeeded (or, at least, a persuasive attempt has been recognized by

[1] In fact, it is difficult to state if a text is persuasive per se: let us consider the following inform sentence: "Monte Bondone is half an hour by car from Trento". It can be considered as a persuasive utterance if emitted as a reply to the sentence: "I'm in Trento and I have a spare afternoon. I'd like to go skiing".

Table 1. List of main tags

Tag	Note
{APPLAUSE}	Main tag in speech transcription.
{SPONTANEOUS-DEMONSTRATION}	Tags replaced: "reaction" "audience interruption"
{STANDING-OVATION}	-
{SUSTAINED APPLAUSE}	Tags replaced: "big applause" "loud applause" etc.
{CHEERS}	Cries or shouts of approval from the audience. Tags replaced: "cries" "shouts" "whistles" etc.
{BOOING}	In this case, the act of showing displeasure by loudly yelling "Boo" Tags replaced: "hissing"
{TAG1 ; TAG2 ; ...}	In case of multiple tagging, tags are divided by semicolon. Usually there are at most two tags.
Special tags	**Note**
{AUDIENCE-MEMBER} [text] {/AUDIENCE-MEMBER}	Tag used to signal a single audience member's intervention such as claques speaking.
{OTHER-SPEAK} [text] {/OTHER-SPEAK}	Tag used to signal speakers other than the subject (like journalists, chairmen, etc.)
{AUDIENCE} [text] {/AUDIENCE}	Tag used to signal audience's intervention.

the audience; on this point see the bibliography on mistimed applause in political speeches [21]). We can then perform specific analyses - and extractions - of persuasive linguistic material that causes the audience reaction.

At present, there are about 900 speeches in the corpus and about 2.2 millions words (see Figure 1 for a survey on main speakers' number of speeches). These speeches have been collected from internet, and an automatic conversion of tags - to make them homogeneous in formalism and labelling - has been performed (see Table 1 for a summary of the tags and their conversion). Given that the tags represent audience reactions this is the case in which there is an "evident" high inter-annotators agreement. Metadata regarding the speech has also been added (title, event, speaker, date, description).

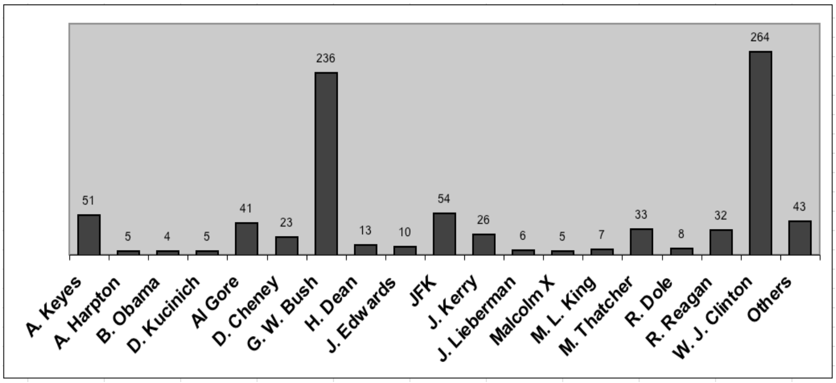


Fig. 1. Number of speeches per speaker

3.2 OVVTs

We drove a preliminary study with human subjects to understand how people perform the task of modifying the valence of existing texts. The insight gained from the study showed that (a) people usually modify single words, (b) sometimes use paraphrases (c) sometimes add or subtract words that play the role of down toners or intensifiers.

We also found that in point (a) there are different classes of valenced terms that are addressed, like adjectives, adverbs, quantifiers, terms indicating strength of belief, etc. We built a resource that gathers these terms in vectors (OVVTs). We used the WordNet `antonymy` relation as an indicator of terms that can be "graded". We built four groups of terms that can be potentially used (one group for each POS). Moreover, we populated the vectors using other specific WN relations (`similar_to` relation for adjectives, `hyponym` relation for verbs and nouns). Finally the valence of WN synsets (taken from SentiWordNet[2] scores [22]) was added to the corresponding lemmata. Thus, an OVVT is composed of several "terms" (lemmata) with similar semantic reference (e.g. beauty) but different valence (see Figure 2, each entry in the OVVTs takes the form `lemma#pos#sense-number`).

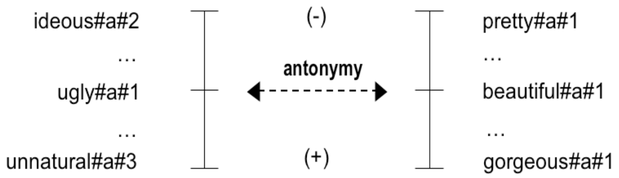


Fig. 2. An example of OVVT

4 Exploiting the Corpus

CORPS has been used both for analysis and generation (the latter use will be briefly discussed in Section 5).

We considered: (a) windows of different width *wn* (where *wn* is the number of tokens considered) of terms preceding tags; and (b) the typology of persuasive communication. We individuate three main groups of tags according to the characteristics of the reaction induced in the audience:

– *Positive-Focus*: this group indicates a persuasive attempt that sets a positive focus in the audience. Tags considered (about 16 thousand): {APPLAUSE}, {SPONTANEOUS-DEMONSTRATION}, {STANDING-OVATION}, {SUSTAINED APPLAUSE}, {AUDIENCE INTERVENTION}, {CHEERING}.

[2] SentiWordNet is a lexical resource in which each WordNet synset is associated to three numerical scores: Obj(s), Pos(s) and Neg(s). These scores represent the objective, positive and negative valence of the synset.

- *Negative-Focus*: It indicates a persuasive attempt that sets a negative focus in the audience. Note that the negative focus is set towards the object of the speech and not on the speaker herself (e.g. "Do we want more taxes?") Tags considered (about 1 hundred): {`BOOING`}, {`AUDIENCE`} `No!` {`/AUDIENCE`}.
- *Ironical*: Indicate the use of ironical devices in persuasion. Tags considered (about 4 thousand): {`LAUGHTER`}[3].

We conducted a preliminary analysis of the corpus focusing on the relation between valence and persuasion: the phase that leads to audience reaction (e.g. `APPLAUSE`), if it presents valence dynamics, is characterized by a valence crescendo. That is to say: not necessarily persuasion is achieved via modification of valence intensity, but, when this is the case, it is by means of an increasing in the valence of the fragment of speech.

To come to this result we calculated, for every window, its mean valence ($\overline{w}$), and subtracted the mean valence of the corresponding speech ($\overline{s}$). In this way we obtained two classes of windows:

- Windows with mean-valence above the mean-valence of the speech ($\overline{w} > \overline{s}$)
- Windows with mean-valence below the mean-valence of the speech ($\overline{s} > \overline{w}$)

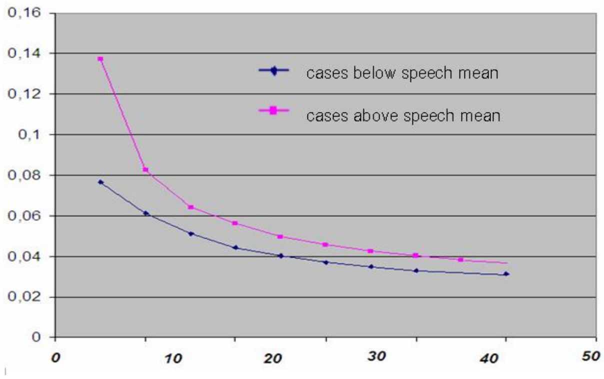


Fig. 3. Relation between valence and persuasion

We then summed up all the values for the two classes and normalized the results by dividing it for the total number of cases in the class (n_c). We repeated the procedure for various window widths ($5 < wn < 40$), see Figure 3 and Formula 1. The results show that cases above the speech mean are fewer but far stronger. We are planning to have a finer grained analysis by means of cluster-based approaches and variable window width.

$$y = \frac{\sum abs\,|\overline{w} > \overline{s}|}{n_c} \qquad x = wn \tag{1}$$

[3] If `LAUGHTER` appears in a multiple tag (e.g. together with `APPLAUSE`) by default this tag is associated to the ironical group. This is not the case for `BOOING` that occurs always alone.

Analysis of public reaction can substantiate intuitions about the speakers' rhetorical style; for example:

How do political speeches change after key historical events? Analyzing the speeches of George W. Bush before and after 9/11 (70 speeches before and 70 after, from 12 months before to 16 months after) at the lexical level we found that: while the positive valence mean remains totally unvaried, the negative increases by 15% (t-test; $\alpha < 0.001$).

What can be said of the lexical choices of a specific speaker that obtains a certain characteristic pattern of public reaction? By considering 30 of Ronald Reagan's (also know as "the great communicator") speeches we found that the mean tag density of this collection was 1/2 of the mean tag density of the whole corpus (t-test; $\alpha < 0.001$). Interestingly, focusing only on the subgroup of ironical tags we found that the density in Reagan's speeches is almost double as compared to the whole corpus (t-test; $\alpha < 0.001$).

How does the perception of the enemy change in different historical moments? A specific analysis on the valence of the lexical context surrounding named entities that elicit negative-focus audience reactions in different period of times can provide interesting insights.

Persuasive Opinion Mining. Not all the opinions expressed in speeches or texts have the same persuasive impact. "Successful" opinions (for example G. W. Bush speaking about W. J. Clinton) can be extracted considering those followed by a reaction of the audience. The role of rhetorical constructs will be taken into account in future research.

We extracted "persuasive words" by using a weighted tf-idf (see Formula 2).

$$tf_i = \frac{n_i \times \sum_{n_i} s_i}{\sum_k n_k} \qquad idf_i = \log \frac{|D|}{|\{d : d \ni t_i\}|} \tag{2}$$

To calculate the tf-idf weight, we created a "virtual document" by unifying all the terms inside all the windows (of dimension wn) preceding the tags, and considering the number of documents in the corpus as coincident to the number of speeches plus one (the virtual document). Obviously from the speeches we subtracted those pieces of text that were used to form the virtual documents. Given this premise we can now define the terms in Formula 2:

- n_i = number of times the term ti appears in the virtual document
- $\sum_{n_i} s_i$ = sum of the scores of the term (the closer to the tag the higher the score)
- $\sum_k n_k$ = the number of occurrences of all terms = $wn \times |tags\, number|$
- $|D|$ = total number of speeches in the corpus
- $|\{d : d \ni t_i\}|$ = number of documents where the term ti appears (we made an hypothesis of equidistribution).

Four lists of words were created according to the group of tags they refer to (positive-focus-words, negative-focus-words, ironical-words and a persuasive-words list - computed by considering all tags together). Analyzing the 100 top words of these lists we found that the negative valence mean of positive-focus and negative-focus groups is the same, while for the the negative-focus group the positive valence mean is about 1/4 with regard to the positive-focus group (t-test; $\alpha < 0.01$). In Table 2 a comparison between the positive-focus and negative-focus top 50 most persuasive words is given (note that named entities have not been discarded).

Table 2. List of top most persuasive words

Positive-focus words	**Negative-focus words**
bless#v deserve#v victory#n justice#n fine#a relief#n November#n win#v help#n thanks#n glad#a stop#v better#r congressman#n lady#n regime#n fabulous#a uniform#n military#a wrong#a soul#n lawsuit#n welcome#v appreciate#v Bush#n behind#r grateful#a 21st#a defend#v responsible#a safe#a terror#n cause#n bridge#n prevail#v choose#v hand#n love#v frivolous#a sir#n honor#n defeat#v end#v fight#n no#r Joe#n ready#a wear#v future#a direction#n foreign#a death#n single#a democratic#a	horrible#a criticize#v waste#n opponent#n timidity#n shuttle#n erode#v torpor#n Soviets#n invasion#n scout#n violation#n Castro#n troop#n authority#n Guevara#n Kaufman#n Sachs#n Goldman#n ferociously#r solvent#n page#n front#a international#a direction#n monstrosity#n Cambodia#n unbearable#a drilling#n Soviet#a increase#v intelligence-gathering#a Carolina#n Gerald#n trusted#a drift#n operation#n WTO#n entry#n mcgovern#v coward#n household#n Neill#n

For lexical choice in text generation micro-planning, there are approaches (e.g. Jing [23]) which use corpus and domain information for choosing appropriate lemmata inside synsets. For persuasive NLG, the lists of words we collected allow us to decide, given a synset and an affective/persuasive goal, which lemma to choose inside which list, to maximize the impact of the message.

With a similar approach we also extracted chunks of persuasive sentences. In this case the window width was based on the number of sentences instead of the number of tokens. We plan to use these chunks in two different ways: for extracting linguistic/rhetorical patterns and rhetorical relations pattern among sentences.

5 The *Valentino* Prototype

In this section we present *Valentino* (VALENced Text INOculator) a tool for modifying existing textual expressions toward more positively or negatively valenced versions as an element of a persuasive system. For instance a strategic planner may decide to intervene on a draft text with the goal of "coloring" it emotionally. When applied to a text, the changes invoked by a strategic level may be uniformly negative or positive; they can smooth all emotional peaks; or they can be introduced in combination with deeper rhetorical structure analysis, resulting in different types of changes for key parts of the texts. Valentino is meant to be an easily pluggable component. The only information it requires in input is a coefficient (included between 1 and -1) that represents the designed valence for the final expression.

At the current stage of implementation only a simple POS analysis (together with named entity recognition and morphological analysis) without contextual information is performed. For this task we used the TextPro package (see [24,25]). Various strategies have been implemented, mimicking those performed by humans.

Paraphrase: if a lemma has only one sense, then the gloss of the word is inserted in the text. The gloss is then valenced, but no more paraphrases are allowed. This augments (a) variety in the output text and (b) the possibility of further valencing the original text (see Table 3 for an example).

Table 3. An example of paraphrase

Original expression	**Selected gloss**	**Shifted Output**
likely he would go ...	with considerable certainty	*with* {*wide*} {*certitude*} He would go

Use of OVVTs considering only the most frequent senses: for every lemma the candidate substitutes are chosen by searching in the OVVTs up to the third sense of that lemma (e.g. given `big#a` it is first searched `big#a#1`, in case of failure `big#a#2` and eventually `big#a#3`).

Candidate lemmas selection: After these two steps there is the necessity to choose among the candidates lemmas. This choice is performed by using the lists of persuasive words that we collected from CORPS. If the shifting is toward positive (negative) valence the list on positive (negative) focus words is accessed first and the candidate with highest ranking is selected.

Strengthening/weakening by modifying adjectives grade: if the chosen lemma is "too weak" (e.g. the output valence should be -1 but the most valenced candidate for substitution is -0.125), the superlative form is used. Also the opposite situation is considered: if the chosen lemma should be in the superlative form (according to the morphology of the substituted term), but the output valence is already met, then the superlative is discarded.

Morphology synthesis: As a final step the chosen lemma is synthesized according to the chosen morphology (either the morphology of the original lemma, or the modified morphology as defined in the aforementioned strategy).

Named entity blocking: Named entities are not valenced to prevent cases like "*Super* Bowl" shifting to "*Giant* Bowl".

In Table 4 various examples of valence shifting of the sentence "He is absolutely the best guy" are given; lemmata chosen from OVVTs are between curly brackets and adjectives that underwent grade modification are between parentheses.

5.1 Advantages and Limits

Even though there are missing scores in SentiWordNet (i.e. words that should be -clearly- valenced that are not, words that are too much valenced) *Valentino* performs reasonably well.

Table 4. An example of Valentino shifting capabilities

CF 1.0	He is {absolutely} (a superb) {hunk}
CF 0.5	He is {highly} the {redeemingest} {signor}
CF 0.0	He is {highly} (a well-behaved) {sir}
CF -0.5	He is {nearly} (a well-behaved) {beau}
CF -1.0	He is {pretty} (an acceptable) {eunuch}

The advantages of using only the most frequent senses of words can be appreciated starting from the sentence "he was a *great* singer":

1. without taking into account the senses frequencies order: "he was a *pregnant*[4] singer"
2. by searching among most frequent senses (1^{st} to 3^{rd}): "he was a *giant* singer"

A strategy based on LSA similarity techniques will further improve the performances of our system, preventing cases like "newspaper *article*" that is (negatively) shifted to "newspaper *lemon*" because "article" is taken in the primary sense of "artifact". Another filter (still using LSA techniques) can rule out cases of incongruence between adjacent words once chosen. For example "toughest eunuch" is a correct but incongruent realization (with coefficient -1) of "tough guy".

The Advantages of using the list of persuasive words can be seen considering the word "giant". It has been chosen from the following bunch of candidate lemmas (score 0.375): `elephantine#a#1 - gargantuan#a#1 - giant#a#1 - jumbo#a#1`

For the second stage of implementation -insertion or deletion of words by considering context- we plan to use WordNet and machine learning techniques to build connections between, for example, semantic typology of verbs and associated adverbs for VP valence modification. E.g. from "He is convinced" to "He is *firmly* convinced".

5.2 Application Scenarios

There are several application scenarios: edutainment systems that should adapt the output to the audience, news agencies wishing to deliver valenced information, conflict management systems that adapt the messages according to the stage of the conflict (fostering escalation or de-escalation) and so on.

An interesting technological scenario is for Embodied Conversational Agents' applications. Often these applications rely on canned, pre-compiled text. Different emotion intensity realizations of the same message are obtained via facial expression (see for example [7]). With *Valentino* the text can be automatically valenced according to emotion intensity, producing a more effective output.

[4] Here "pregnant" is in the secondary sense of "significant" which is correct but sounds odd.

6 Conclusions

We have presented some resources (in particular the corpus CORPS, that we plan to put freely available for research purposes) and techniques for statistical acquisition of persuasive expressions with a view of contributing to various persuasive NLP tasks. Affective expressions are of paramount importance.

We implemented a prototype named *Valentino* that uses a term extraction and transformation approach: given a term in the text to be modified, the system accesses the OVVT containing that term and chooses the most appropriate transformation in agreement with the valence shift for the persuasive goal.

References

1. Perelman, C., Olbrechts-Tyteca, L.: The new Rhetoric: a treatise on Argumentation. Notre Dame Press (1969)
2. Reed, C., Grasso, F.: Recent advances in computational models of argument. International Journal of Intelligent Systems 22(1), 1–15 (2007)
3. Miceli, M., deRosis, F., Poggi, I.: Emotional and non-emotional persuasion. Applied Artificial Intelligence 20 (2006)
4. Reed, C., Rowe, G.: Araucaria: Software for argument analysis, diagramming and representation. International Journal of AI Tools 14 (3-4), 961–980 (2004)
5. Reiter, E., Robertson, R., Osman, L.: Lesson from a failure: Generating tailored smoking cessation letters. Artificial Intelligence 144, 41–58 (2003)
6. Reiter, E., Sripada, S., Robertson, R.: Acquiring correct knowledge for natural language generation. Journal of Artificial Intelligence Research 18, 491–516 (2003)
7. Guerini, M., Stock, O., Zancanaro, M.: A taxonomy of strategies for multimodal persuasive message generation. Applied Artificial Intelligence Journal 21(2), 99–136 (2007)
8. Toulmin, S.: The Use of Arguments. Cambridge University Press, Cambridge (1958)
9. Walton, D.: Argumentation Schemes for Presumptive Reasoning. Lawrence Erlbaum Associates, Mahwah (1996)
10. Zukerman, I., McConachy, R., Korb, K.: Using argumentation strategies in automated argument generation. In: Proceedings of the 1st International Natural Language Generation Conference, pp. 55–62 (2000)
11. Reed, C., Long, D.: Ordering and focusing in an architecture for persuasive discourse planning. In: Proceedings of the 6th European Workshop on Natural Language Generation (EWNLG 1997), Duisburg, Germany (1997)
12. Mazzotta, I.: deRosis, F., Carofiglio, V.: Portia: A user-adapted persuasion system in the healthy eating domain. IEEE Intelligent Systems, Special Issue on Argumentation Technology (in press, 2007)
13. Restificar, A.C., Ali, S.S., McRoy, S.W.: Arguer: Using argument schemas for argument detection and rebuttal in dialogs. In: Proceedings of the Seventh International Conference on User Modelling (UM-1999), June 20-24, 1999, Banff, Canada (1999)
14. Piwek, P.: An annotated bibliography of affective natural language generation. ITRI ITRI-02-02, University of Brighton (2002)
15. deRosis, F., Grasso, F.: Affective natural language generation. In: Paiva, A. (ed.) IWAI 1999. LNCS, vol. 1814, Springer, Heidelberg (2000)

16. Carofiglio, V., deRosis, F.: Combining logical with emotional reasoning in natural argumentation. In: Brusilovsky, P., Corbett, A.T., de Rosis, F. (eds.) UM 2003. LNCS, vol. 2702, Springer, Heidelberg (2003)
17. Carenini, G., Ng, R., Zwart, E.: Extracting knowledge from evaluative text. In: Proceedings of the 3rd international conference on Knowledge Capture, pp. 11–18 (2005)
18. Wilson, T., Wiebe, J., Hwa, R.: Just how mad are you? finding strong and weak opinion clauses. In: Proceedings of AAAI, pp. 761–769 (2004)
19. Carenini, G., Ng, R., Pauls, A.: Multi-document summarization of evaluative text. In: Proceedings of EACL (2006)
20. Radev, D., McKeown, K.: Building a generation knowledge source using internet-accessible newswire. In: Proceedings of the 5th Conference on Applied Natural Language Processing (1997)
21. Bull, P., Noordhuizen, M.: The mistiming of applause in political speeches. Journal of Language and Social Psychology 19, 275–294 (2000)
22. Esuli, A., Sebastiani, F.: SentiWordNet: A publicly available lexical resource for opinion mining. In: Proceedings of the 5th Conference on Language Resources and Evaluation, Genova, IT, pp. 417–422 (2006)
23. Jing, H.: Usage of wordnet in natural language generation. In: Harabagiu, S. (ed.) Proceedings of the conference Use of WordNet in Natural Language Processing Systems, Somerset, New Jersey, Association for Computational Linguistics, pp. 128–134 (1998)
24. Pianta, E., Zanoli, R.: Tagpro: a system for italian pos tagging based on svm. Intelligenza Artificiale, Numero Speciale Strumenti di Elaborazione del Linguaggio Naturale per l'Italiano 4(2), 8–9 (2007)
25. Zanoli, R., Pianta, E.: Entitypro: exploiting svm for italian named entity recognition. Intelligenza Artificiale, Numero Speciale Strumenti di Elaborazione del Linguaggio Naturale per l'Italiano 4(2), 69–70 (2007)

Sense Annotation in the Penn Discourse Treebank

Eleni Miltsakaki[1], Livio Robaldo[2], Alan Lee[1], and Aravind Joshi[1]

[1] Institute for Research in Cognitive Science, University of Pennsylvania
{elenimi,aleewk,joshi}@linc.cis.upenn.edu
[2] Department of Computer Science, University of Turin
robaldo@di.unito.it

Abstract. An important aspect of discourse understanding and generation involves the recognition and processing of discourse relations. These are conveyed by discourse connectives, i.e., lexical items like *because* and *as a result* or implicit connectives expressing an inferred discourse relation. The Penn Discourse TreeBank (PDTB) provides annotations of the argument structure, attribution and semantics of discourse connectives. In this paper, we provide the rationale of the tagset, detailed descriptions of the senses with corpus examples, simple semantic definitions of each type of sense tags as well as informal descriptions of the inferences allowed at each level.

1 Introduction

Large scale annotated corpora have played and continue to play a critical role in natural language processing. The continuously growing demand for more powerful and sophisticated NLP applications is evident in recent efforts to produce corpora with richer annotations [6], including annotations at the discourse level[2], [8], [4]. The Penn Discourse Treebank is, to date, the largest annotation effort at the discourse level, providing annotations of explicit and implicit connectives. The design of this annotation effort is based on the view that discource connectives are predicates taking clausal arguments. In Spring 2006, the first version of the Penn Discourse Treebank was released, making availalble thousands annotations of discourse connectives and the textual spans that they relate.

Discourse connectives, however, like verbs, can have more than one meaning. Being able to correctly identify the intended sense of connectives is crucial for every natural language task which relies on understanding relationships between events or situations in the discourse. The accuracy of information retrieval from text can be significantly impaired if, for example, a temporal relation anchored on the connective *since* is interpreted as causal.

A well-known issue in sense annotations is identifying the appropriate level of granularity and meaning refinement as well as identifying consistent criteria for making sense distinctions. Even if an 'appropriate' level of granularity can be identified responding to the demands of a specific application, creating a flat set of sense tag is limiting in many ways. Our approach to the annotation of sense tag in PDTB is to define a small hierarchy of sense tags containing coarse sense distinctions at the top and finer at the bottom. This schema is flexible enough to allow the annotators to choose a tag from a level that is comfortable to them. In addition, it allows the user of the corpus to pick the level that is useful for his or her purposes or even add levels of annotation if finer distinctions are desirable.

In this paper, we present our work on adding sense annotations to all the explicit and implicit connectives in the Penn Discourse Treebank (approx. 35,000 tokens). In Section (2), we give a broad overview of the Penn Discourse Treebank, detailing the types of connectives that have been annotated. In Section (3), we present the tagset used for the annotation of senses of connectives in the Penn Discourse Treebank, its hierarchical organization, and simple formal semantic descriptions for each tag. In Section (4), we present a small set of pragmatic tags that we used to capture rhetorical uses of connectives.

2 The Penn Discourse Treebank

Following the views toward discourse structure in [12] and [3], the Penn Discourse Treebank treats discourse connectives as discourse-level predicates that take two abstract objects such as events, states, and propositions [1] as their arguments. It provides annotations of the argument structure, attribution and semantics of discourse connectives. The PDTB annotations are done on the Wall Street Journal (WSJ) articles in the Penn TreeBank (PTB) II corpus [7]. Each annotation relates a discourse connective with its two[1] arguments, labelled as `Arg2`, for the argument that appears in the clause that is syntactically bound to the connective, and `Arg1`, for the other argument.

Discourse connectives in the PDTB are distinguished primarily into `Explicit` discourse connectives, that include a set of lexical items drawn from well-defined syntactic classes, and `Implicit` discourse connectives, which are inserted between paragraph-internal adjacent sentence-pairs not related explicitly by any of the syntactically-defined set of `Explicit` connectives. In the latter case, the reader must attempt to infer a discourse relation between the adjacent sentences, and 'annotation' consists of *inserting* a connective expression that *best* conveys the inferred relation. Multiple discourse relations can also be inferred, and are annotated by inserting multiple `Implicit` connectives. In (1), we show three examples that respectively involve an `Explicit` connective (1.a), an `Implicit` connective (1.b), and multiple `Implicit` connectives (1.c). In all examples reported below, `Arg1` is shown in italics, `Arg2` in boldface, and the discourse connective(s) underlined.

(1) a. *She hasn't played any music* <u>since</u> **the earthquake hit**.
b. *They stopped delivering junk mail.* <u>[Implicit=so]</u> **Now thousands of mailers go straight into the trash**.

[1] The assumption of the arity constraint on a connective's arguments has been upheld in all the annotation done thus far. Discourse-level predicate-argument structures are therefore unlike the predicate-argument structures of verbs at the sentence-level, where verbs can take any number of arguments.

c. *The small, wiry Mr. Morishita comes across as an outspoken man of the world.* [Implicit=when, Implicit=for example] **He lectures a visitor about the way to sell American real estate and boasts about his friendship with Margaret Thatcher's son**.

Adjacent sentence-pairs between which `Implicit` connectives cannot be inserted are further distinguished and annotated as three types: `AltLex`, for when a discourse relation is inferred, but insertion of an `Implicit` connective leads to a redundancy in the expression of the relation due to the relation being alternatively lexicalized by some 'non-connective' expression (as in (2.a)); `EntRel`, for when no discourse relation can be inferred and where the second sentence only serves to provide some further description of an entity in the first sentence (as in (2.b)), and `NoRel`, for when no discourse relation or entity-based coherence relation can be inferred between the adjacent sentences (as in (2.c)).

(2) a. *So Seita has introduced blonde cigarettes under the Gauloises label, and intends to relaunch the unsuccessful Gitanes Blondes in new packaging.* [AltLex=the aim is] **The aim is to win market share from imported cigarettes, and to persuade smokers who are switching to blonde cigarettes to keep buying French.**.

b. *Proceeds from the offering are expected to be used for remodeling the company's Desert Inn resort in Las Vegas, refurbishing certain aircraft of the MGM Grand Air unit, and to acquire the property for the new resort.* EntRel **The company said it estimates the Desert Inn remodeling will cost about $32 million, and the refurbishment of the three DC-8-62 aircraft, made by McDonnell Douglas Corp., will cost around $24.5 million**.

c. *Jacobs is an international engineering and construction concern.* NoRel **Total capital investment at the site could be as much as $400 million, according to Intel**.

The PDTB has been used as a resource for Natural Language Generation [11], and for Sense Disambiguation [10]. This paper focuses on PDTB sense annotation, and describes the tagset used to annotate the discourse connectives. The reader interested in the overall annotation is addressed to [9] and [13].

3 Annotation of Senses in the PDTB

The Penn Discourse Treebank provides sense tags for the `Explicit`, `Implicit` and `AltLex` connectives. Depending on the context, the content of the arguments and possibly other factors, discourse connectives, just like verbs, can have more than one meaning. For example, *since* seems to have three different senses, one purely 'Temporal' (as in (3.a)), another purely 'Causal' (as in (3.b)) and a third both 'Causal' and 'Temporal' (as in (3.c)).

(3) a. *The Mountain View, Calif., company has been receiving 1,000 calls a day about the product* <u>since</u> **it was demonstrated at a computer publishing conference several weeks ago**.

b. *It was a far safer deal for lenders* <u>since</u> **NWA had a healthier cash flow and more collateral on hand**.

c. *Domestic car sales have plunged 19%* <u>since</u> **the Big Three ended many of their programs Sept. 30**.

Sense annotations in PDTB provide tags specifying the sense of the connective in cases of ambiguity, and in every case they provide a semantic description of the relation between the arguments of connectives. When annotators identify more than one simultaneous interpretations, multiple sense tags are provided. Sense annotations specify one or more, but not necessarily all the semantic relations that may hold between the arguments of the connectives.

The tagset of senses is organized hierarchically (shown in Figure 1) and comprises three levels: *class*, *type* and *subtype*. The top level, or *class level* of the hierarchy represents four major semantic classes: 'TEMPORAL', 'CONTINGENCY', 'COMPARISON' and 'EXPANSION'. For each class, a second level of *types* is defined to further refine the semantics of the class levels. For example, 'CONTINGENCY' has two types, 'Cause' (relating two situations via a direct cause-effect relation) and 'Condition' (relating a hypothetical scenario with its possible consequences). A third level of *subtype* specifies the semantic contribution of each argument. For 'CONTINGENCY', its 'Cause' type has two subtypes, 'reason' and 'result', which specify which argument is interpreted as the cause of the other. A typical connective labelled as 'reason' is *because* and a a typical connective labelled as 'result' is *as a result*.

For most types and subtypes, we also provide some hints about their possible semantics. In doing so, we do not attempt to represent the internal meaning of `Arg1` and `Arg2`, but simply refer to them as $\| Arg1 \|$ and $\| Arg2 \|$ respectively. We believe that roughing out the semantics of the sense tags provides a starting point for the definition of an integrated logical framework able to deal with the semantics of discourse connectives but it also helps the annotators in choosing the proper sense tag.

The hierarchical organization of the sense tags serves two purposes. First, it efficiently addresses well-known issues regarding inter-annotator reliability, by allowing the annotators to select a tag from a level that is comfortable to them. Sense annotators in PDTB are not forced to make fine semantic distinctions when they are not confident that their world knowledge or discourse context can support more specific interpretations. Secondly, the hierarchical organization of tags also allows useful inferences at all levels. For example, (1) illustrates a case where neither the text nor the annotators' world knowledge has been sufficient to enable them to provide a sense tag at the level of subtype. Instead, they have provided one at the level of type.

(1) *Besides, to a large extent, Mr. Jones may already be getting what he wants out of the team*, <u>even though</u> **it keeps losing**.

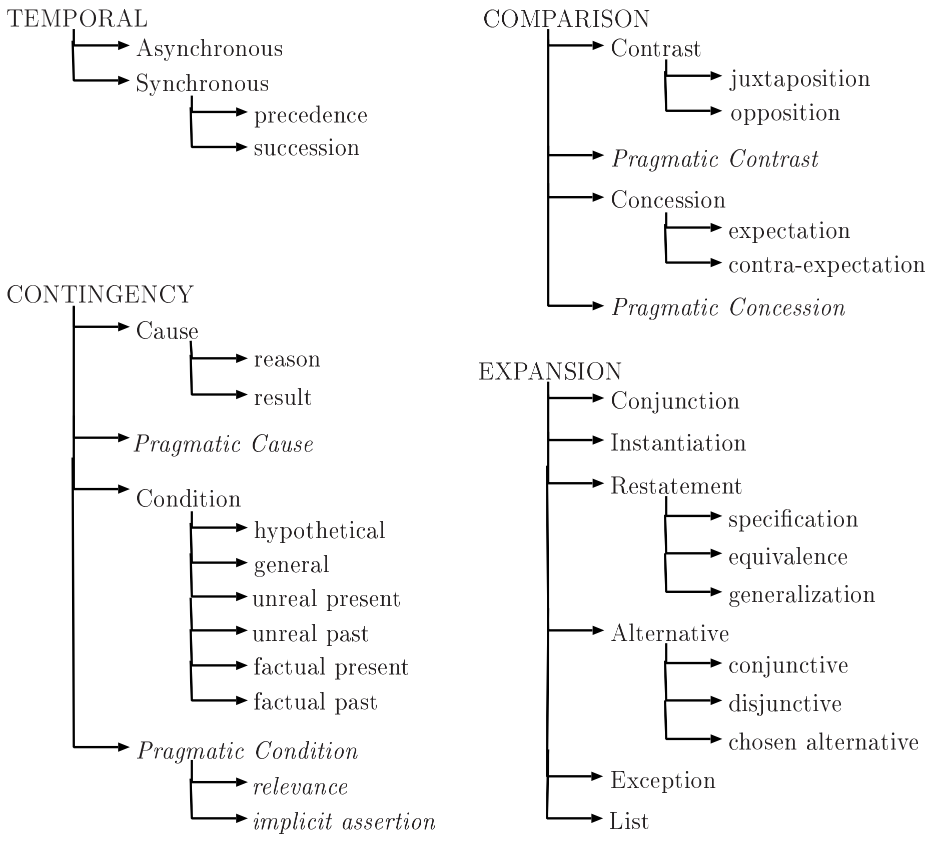


Fig. 1. Hierarchy of sense tags

Connectives can also be used to relate the *use* of the arguments of a connective to one another or the use of one argument with the sense of the other. For these *rhetorical* or *pragmatic* uses of connectives, we have defined *pragmatic* sense tags - specifically 'Pragmatic Cause', 'Pragmatic Condition', 'Pragmatic Contrast' and 'Pragmatic Concession'.

In what follows, we provide descriptions for all the semantic labels of the sense hierarchy.

3.1 Class 'TEMPORAL'

'TEMPORAL' is used when the situations described in the arguments are related temporally. The class level tag 'TEMPORAL' does not specify if the situations are temporally ordered or overlapping. Two types are defined for 'TEMPORAL': 'Asynchronous' (i.e., temporally ordered) and 'Synchronous' (i.e., temporally overlapping). 'Asynchronous' has two subtypes, 'precedence' and 'succession', which specify which situation takes place before the other one. The tag 'precedence' is used when the connective indicates that the situation in `Arg1` precedes the situation described in `Arg2`, as *before* does in (2). The tag 'succession' is used when the connective indicates that the situation described in `Arg1` follows the situation described in `Arg2`, as *after* does in (3).

(2) But a Soviet bank here would be crippled unless Moscow found a way to settle the $188 million debt, *which was lent to the country's short-lived democratic Kerensky government* <u>before</u> **the Communists seized power in 1917**.

(3) No matter who owns PS of New Hampshire, <u>after</u> **it emerges from bankruptcy proceedings** *its rates will be among the highest in the nation*, he said.

The tag 'Synchronous' applies when the connective indicates that the situations described in `Arg1` and `Arg2` overlap. The type 'Synchronous' does not specify the form of overlap, i.e., whether the two situations started and ended at the same time, whether one was temporally embedded in the other, or whether the two crossed. Typical connectives tagged as 'Synchronous' are *while* and *when*, the latter shown in (4).

(4) *Knowing a tasty – and free – meal* <u>when</u> **they eat one**, the executives gave the chefs a standing ovation. (TEMPORAL:Synchrony) (0010)

3.2 Class 'CONTINGENCY'

'CONTINGENCY' is used when the situations described in the arguments are causally influenced. It has two types, 'Cause' and 'Condition'. The main difference between the two is that in 'Cause' the connective expressing the relation does not have any impact on whether the arguments are taken to hold or not. For instance, in (4.a), the situations specified in `Arg1` and `Arg2` are taken to hold true independently of the connective. It is true that the use of dispersants was approved, that a test on the third day showed positive results, and that the latter caused the former. In this case, the directionality of causality, i.e., that $\|\texttt{Arg2}\|$ is the cause and $\|\texttt{Arg1}\|$ the effect, is specified with the subtype 'reason'. Formally, we represent the semantics of 'reason' as $\|\texttt{Arg1}\|{<}\|\texttt{Arg2}\| \wedge \|\texttt{Arg1}\| \wedge \|\texttt{Arg2}\|$, where $<$ is a logical operator taken from [5]. $\texttt{Arg1}{<}\texttt{Arg2}$ is intended to model the causal law[2] '`Arg1` causes `Arg2`'. The reverse case, i.e., when $\|\texttt{Arg1}\|$ is the cause and $\|\texttt{Arg2}\|$ the effect, is labelled with they subtype 'result'.

The type 'Condition' is used to describe all subtypes of conditional relations. In addition to causal influence, 'Condition' allows some basic inferences about the semantic contribution of the arguments. Specifically, the situation in `Arg2` is taken to be the condition and the situation described in `Arg1` is taken to be the consequence, i.e., the situation that holds when the condition is true. Unlike 'Cause', however, the truth value of the arguments of a 'Condition' relation cannot be determined independently of the connective. For this reason, we introduce

[2] As largerly discussed in the literature, causality cannot be modeled via the logical implication '$\rightarrow$'. '$\rightarrow$' will be used to handle the semantic of 'Restatement' (see below).

some branching-time logic operators into our rough description of the semantics of 'Condition' subtypes: A, F, and G. A universally quantifies over all possible futures; therefore, $A\beta$ is true iff β is true in all possible futures. F and G are respectively existential and universal quantifiers over instants in a single future: $F\alpha$ is true iff α is true in some instant in a possible future, while $G\alpha$ is true iff α is true in every instant in a possible future.

The sense hierarchy includes six basic subtypes of 'Condition'. Example (4.b) is marked with the subtype 'general' because the sentence-pair describes a generic truth about the world (or a statement that describes a regular outcome everytime the condition holds true). We formalize its semantics as $AG(\|\texttt{Arg2}\| < \|\text{Arg1}\|)$, i.e., when a sentence-pair is tagged as 'general', in all possible futures, it is always the case that $\|\texttt{Arg2}\|$ causes $\|\texttt{Arg1}\|$. Compare with 'hypothetical' which marks a causal relation that holds true only at the moment when the sentence is uttered. An example is shown in (4.c); this is a case of 'hypothetical' in that, in the future, even if the negotiators start to focus on those areas, the talks may be unsuccessful (i.e., in the future, there may be other factors that affect the performance of the talks). The formal semantics of 'hypothetical' is $\|\texttt{Arg2}\| < AF\ \|\texttt{Arg1}\|$: if $\|\texttt{Arg2}\|$ holds true, $\|\texttt{Arg1}\|$ is caused to hold too at some instant in all possible subsequent futures. Examples (4.d) and (4.e) are respectively marked as 'factual present' and 'unreal present'. The tag 'factual present' applies when `Arg2` denotes a situation that has either been presented as a fact in the prior discourse or is believed by somebody other than the speaker or writer. From a formal point of view, we add a conjunct stating that $|\text{Arg2}|$ is true or it is believed to hold true. The subtype 'unreal present' applies when `Arg2` describes a condition that either does not hold at present or considered unlikely to hold. In such a case, we assert the formula $\|\texttt{Arg2}\| < AF\ \|\texttt{Arg1}\| \wedge \sim\|\texttt{Arg2}\|$[3]. The other two subtypes are 'factual past' and 'unreal past', which are respectively similar to 'factual present' and 'unreal present' except that in this case the first argument refers to a situation that is assumed to have taken place at a time in the past.

(4) a. *Use of dispersants was approved* when **a test on the third day showed some positive results**.
    b. *They won't buy* if **the quality is not here**.
    c. Both sides have agreed *that the talks will be most successful* if **negotiators start by focusing on the areas that can be most easily changed**.
    d. If **that's true**, *Orange County has to be at least 10% of that.*
    e. If **the film contained dialogue**, *Mr.Lane's Artist would be called a homeless person.*

3.3 Class 'COMPARISON'

The class tag 'COMPARISON' applies when a discourse relation is established between `Arg1` and `Arg2` in order to highlight prominent differences between the

[3] $\sim\|\texttt{Arg2}\|$ means that $\|\texttt{Arg}\|$ does not hold or not expected to hold.

two situations. Semantically, the truth of both arguments is independent of the connective or the established relation. 'COMPARISON' has two types to further specify its semantics. In some cases, `Arg1` and `Arg2` share a predicate or a property and the difference is highlighted with respect to the values assigned to this property. This interpretation is tagged with the type 'Contrast'. There are also cases in which the highlighted differences are related to expectations raised by one argument which are then denied by the other. This intepretation is tagged with the Type 'Concession'. In 'Contrast' both arguments describe a situation that is not asserted on the basis of the other one. In this sense, there is no directionality in the interpretation of the arguments. This is an important difference between the interpretation of 'Contrast' and 'Concession'. Two subtypes of 'Contrast' are defined to further specify the type of values that are compared: 'juxtaposition' (weak contrast) and 'opposition' (strong contrast). The latter is applied when the values assigned to some shared property are taken from the extremes of a gradable scale (e.g., tall-short, accept-reject, etc.), the former otherwise. For example, (5.a) is tagged as 'opposition' because the two arguments describe opposite performances of the banks. Example (5.b) is tagged as 'juxtaposition' because the shared predicate *rose* or *jumped* takes two different values (69% and 85%) and the shared predicate *rose to X amount* takes two individual entities (the net operating venue and the net internet bill).

(5) a. *Its bank in Texas also reported a loss of $23.5 million for the quarter* <u>but</u> **that its consumer banks in Oregon, California, Nevada and Washington performed well during the quarter**.
b. *Operating revenue rose 69% to A$8.48 billion from A$5.01 billion* <u>but</u> **the net interest bill jumped 85% to A $686.7 million from A $371.1 million**.

'Concession' also has two subtypes: 'expectation' and 'contra-expectation'. The subtype 'expectation' applies when `Arg2` describes a situation A which causes another situation C, and `Arg1` asserts (or implies) the situation $\neg C$ (i.e $\|\texttt{Arg2}\| < C \wedge \|\texttt{Arg1}\| \rightarrow \neg C$), as in (6.a). The subtype 'contra-expectation' applies when `Arg1` causes C and `Arg1` denies it, as in (6.b).

(6) a. <u>Although</u> **the purchasing managers' index continues to indicate a slowing economy**, *it isn't signaling an imminent recession* .
b. *The Texas oilman has acquired a 26.2% stake valued at more than $1.2 billion in an automotive-lighting company, Koito Manufacturing Co.* <u>But</u> **he has failed to gain any influence at the company.**.

Some times in the discourse the intended 'juxtaposition' or 'opposition' is clear and sometimes it is not. When it is not, the sense of the connective is considered ambiguous and the higher level tag 'Contrast' applies. In fact, the gradable scale with respect to which we discriminate between 'juxtaposition' and 'opposition' strongly depends on the context where the sentence is uttered. For example, consider the pair *black-white*. These two concepts are usually taken to be antonyms.

Therefore, it seems that whenever `Arg1` assigns *black* and `Arg2` assigns *white* to a shared property (e.g. *Mary is black whereas John is white*), the discourse connective has to be labelled as 'opposition'. Nevertheless, in many contexts *black* and *white* are just two of the colors that may be assigned to the shared property (e.g., imagine *Mary bought a black hat whereas John bought a white one* uttered in a shop that sell red, yellow and blue hats as well). In such a case, they are not antonyms, and the discourse connective has to be labelled as 'juxtaposition'.

3.4 Class 'EXPANSION'

Under the class 'EXPANSION' we group all the relations which expand the discourse and move forward its narrative or exposition. 'EXPANSION' includes several types which refine its semantics. The type 'Conjunction' is used when the situation described in `Arg2` provides additional, discourse new, information that is related to the situation described in `Arg1`. It is inferred that the information described in `Arg2` is not related to `Arg1` in any of the ways described in the other types of 'EXPANSION' ($\|\texttt{Arg1}\| \wedge \|\texttt{Arg2}\|$). The tag 'Instantiation' is used when $\|\texttt{Arg1}\|$ evokes a set of events and $\|\texttt{Arg2}\|$ picks up one of these events and describes it in further detail; in this case, besides the logical conjuntion of the arguments, we assert $exemplify'(\|\texttt{Arg2}\|, \lambda x.x{\in}g(\|\texttt{Arg1}\|))$, where $exemplify'$ is a predicate taken from [3], g a function that 'extracts' the set of events from the semantics of `Arg1`, and x is a variable ranging over them. $exemplify'$ asserts that `Arg2` further describes one element in the extracted set. A connective is marked as 'Restatement' when the semantics of `Arg2` restates the semantics of `Arg1`. It is inferred that the situations described in `Arg1` and `Arg2` hold true at the same time. The subtypes 'specification', 'generalization', and 'equivalence' further specify the ways in which `Arg2` restates `Arg1`. In particular, besides the conjunction of the two arguments, we assert a logical implication ($\rightarrow$) between the arguments: $\|\texttt{Arg1}\|{\rightarrow}\|\texttt{Arg2}\|$ (generalization), $\|\texttt{Arg1}\|{\leftarrow}\|\texttt{Arg2}\|$ (specification), and $\|\texttt{Arg1}\|{\leftrightarrow}\|\texttt{Arg2}\|$ (equivalence). The type 'Alternative' applies when the two arguments of the connective evoke two alternative situations. The type 'Alternative' is further specified with the subtypes 'conjunctive', 'disjunctive' and 'chosen alternative'. The 'conjunctive' subtype is used when both alternatives are possible ($\|\texttt{Arg1}\| \vee \|\texttt{Arg2}\|$), 'disjunctive' when two situations are evoked in the discourse but only one of the two holds ($\|\texttt{Arg1}\|$ xor $\|\texttt{Arg2}\|$), and 'chosen alternative' when two alternatives are evoked in the discourse and the one denoted by `Arg2` is taken ($\|\texttt{Arg1}\|$ xor $\|\texttt{Arg2}\| \wedge \|\texttt{Arg2}\|$). The type 'Exception' applies when `Arg2` evokes a situation which makes `Arg1` not fully be true. In other words, in case of 'Exception', `Arg1` is false, `Arg2` is true and if `Arg2` were false, `Arg1` would be true. So, the formal semantics of 'Exception' is $\neg\ \|\texttt{Arg1}\| \wedge \|\texttt{Arg2}\| \wedge\neg\ \|\texttt{Arg2}\|{\rightarrow}\|\texttt{Arg1}\|$. Finally, the type 'List' applies when the events or states expressed in `Arg1` and `Arg2` are members of a list of events or states enumerated in the discourse. It is possible that semantically `Arg1` and `Arg2` are not related. For the appropriate interpretation of the discourse all the elements of the list must be retrieved. The predicate of the list (i.e., what `Arg1` and `Arg2` are elements of) must be retrieved from the prior discourse.

In (7) we provide examples of ‘Instantiation’ (7.a), ‘specification’ (7.b), ‘chosen alternative’ (7.c), and ‘Exception’ (7.c). Unfortunately, for space constraints we cannot provide an example of each type and subtype of ‘EXPANSION’.

(7) a. *Hypertext books are clearly superior to normal books.* For example, **they have database cross-referencing facilities ordinary volumes lack**.
b. *I never gamble too far.* [Implicit=In particular] **I quit after one try, whether I win or lose**.
c. It isnt allowed to *share in the continuing proceeds when the reruns are sold to local stations.* Instead **ABC will have to sell off the rights for a one-time fee**.
d. *Boston Co. officials declined to comment on the unit’s financial performance this year* except **to deny a published report that outside accountants had discovered evidence of significant accounting errors in the first three quarters’ results**.

4 Pragmatic Uses of Connectives

The PDTB contains instances of rhetorical or pragmatic uses of connectives. For these instances, we define a small set of sense tags called *pragmatic*. We have found instances of ‘Pragmatic cause’, ‘Pragmatic Condition’, ‘Pragmatic Concession’, and ‘Pragmatic Contrast’. For instance, the tag ‘Pragmatic Condition’ is used for instances of conditional constructions whose interpretation deviates from that of the semantics of ‘Condition’. Specifically, these are cases of explicit *if* tokens with `Arg1` and `Arg2` not being causally related. In all cases, `Arg1` holds true independently of `Arg2`. Two subtypes of ‘Pragmatic Condition’ have been defined: ‘relevance’ and ‘implicit assertion’. The former applies where `Arg2` provides the context in which the description of the situation in `Arg1` is relevant. An example of ‘relevance’ is shown in (8.a); note that there is no causal relation between the two arguments. The pragmatic tag ‘implicit assertion’ applies in special rhetorical uses of if-constructions when the intepretation of the conditional construction is an implicit assertion. In (8.b), for example, `Arg1`, *O’Connor is your man* is not a consequent state that will result if the condition expressed in `Arg2` holds true. Instead, the conditional construction in this case implicitly asserts that O’Connor will keep the crime rates high.

(8) a. If **anyone has difficulty imagining a world in which history went merrily on without us**, *Mr. Gould sketches several.*
b. If **you want to keep the crime rates high**, *O’Connor is your man.*
c. *Mrs Yeargin is lying.* [Implicit=because] **They found students in an advanced class a year earlier who said she gave them similar help**.

‘Pragmatic Cause’ is used when `Arg1` expresses a claim and `Arg2` provides justification for this claim, as shown in (8.c). The situations described in `Arg1` and `Arg2` are not causally influenced. Epistemic uses fall under this category.

5 Inter-annotator Agreement and Adjudication

The PDTB corpus was sense annotated by two annotators. Class level inter-annotator agreement was 92% and subtype level agreement (the most refined level) was 77%. Class level disagreement was adjudicated by a team of three experts. Disagreement at lower levels was resolved by providing a sense tag from the immediately higher level. For example, if one annotator tagged a token with the type 'Concession' and the other, with the type 'Contrast', we resolved the disagreement by providing the the class level tag 'Comparison' based on the assumption that both a concessive and contrastive interpretation could be construed.

6 Summary and Future Work

The Penn Discourse Treebank provides annotations of discourse connectives and their arguments. Discourse connectives, like verbs, can have more than one sense. Here, we presented the tagset that we used to annotate the senses of connectives. The tagset is organized hierarchically in three levels, with coarse sense distinctions made at the top level and finer distinctions provided in lower levels. The PDTB corpus contains instances of rhetorical uses of connectives. These instances are annotated with a small set of pragmatic tags. For each sense tag we provided a simple formal description of its semantics. We are currently studying the distribution of senses per connective and looking more closely at semantic features of the arguments in order to develop empirically motivated descriptions of the semantic roles of the arguments and also in order to identify useful features for models of automatic sense disambiguation.

Acknowledgments

We are grateful to Nikhil Dinesh, Rashmi Prasad and Bonnie Webber for their insightful comments on this work. The PDTB project is partially supported by NSF Grant: Research Resources, EIA 02-24417 to the University of Pennsylvania (PI: Aravind Joshi). The first release (PDTB-1.0) is distributed free of charge under terms of the GNU General Public License. The corpus and associated software is available for download from http://www.seas.upenn.edu/ pdtb. The second release, including sense annotations, is scheduled for January 2008.

References

1. Asher, N.: Reference to Abstract Objects. Kluwer, Dordrecht (1993)
2. Carlson, L., Marcu, D., Okurowski, M.: Current Directions in Discourse and Dialogue, Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory. Kluwer Academic Publishers, Dordrecht (2003)
3. Forbes-Riley, K., Webber, B., Joshi, A.: Computing discourse semantics: The predicate-argument semantics of discourse connectives in D-LTAG. Journal of Semantics 23, 55–106 (2006)

4. Gaizauskas, R., et al.: The timebank corpus. In: Corpus Linguistics 2003. Lancaster, U.K (2003)
5. Giordano, L., Schwind, C.: Conditional logic of actions and causation. Artificial Intelligence 157, 239–279 (2004)
6. Kingsbury, P., Palmer, M.: From Treebank to Propbank. In: Third International Conference on Language Resources and Evaluation, LREC-2002, Las Palmas, Canary Islands, Spain (2002)
7. Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of English: The Penn Treebank. Computational Linguistics 19(2), 313–330 (1993)
8. Mitkov, R., et al.: Coreference and anaphora: Developing annotating tools, annotated resources and annotation strategies. In: Proc. of the Discourse Anaphora and Anaphora Resolution Colloquium (DAARC 2000), Lancaster, U.K. (2000)
9. Miltsakaki, E., et al.: The Penn Discourse Treebank. In: Proc. of the 4th International Conference on Language Rescourses and Evaluation (LREC 2004) (2004)
10. Miltsakaki, E., et al.: Experiments on sense annotation and sense disambiguation of discourse connectives. In: Proc. of the 4th Workshop on Treebanks and Linguistic Theories (TLT2005) (2005)
11. Prasad, R., et al.: Discourse TreeBank as a resource for natural language generation. In: Proc. of the Corpus Linguistics Workshop on Using Corpora for NLG (2005)
12. Webber, B., et al.: Anaphora and discourse structure. Computational Linguistics 29(4), 545–587 (2003)
13. Webber, B., et al.: A short introduction to the PDTB. In: Copenhagen Working Papers in Language and Speech Processing (2005)

A Semantics-Enhanced Language Model for Unsupervised Word Sense Disambiguation

Shou-de Lin[1] and Karin Verspoor[2]

[1] National Taiwan University
sdlin@csie.ntu.edu.tw
[2] Los Alamos National Laboratory
verspoor@lanl.gov

Abstract. An N-gram language model aims at capturing statistical word order dependency information from corpora. Although the concept of language models has been applied extensively to handle a variety of NLP problems with reasonable success, the standard model does not incorporate semantic information, and consequently limits its applicability to semantic problems such as word sense disambiguation. We propose a framework that integrates semantic information into the language model schema, allowing a system to exploit both syntactic and semantic information to address NLP problems. Furthermore, acknowledging the limited availability of semantically annotated data, we discuss how the proposed model can be learned without annotated training examples. Finally, we report on a case study showing how the semantics-enhanced language model can be applied to unsupervised word sense disambiguation with promising results.

1 Introduction

Syntax and semantics both play an important role in language use. Syntax refers to the grammatical structure of a language whereas semantics refers to the meaning of the symbols arranged with that structure. To fully comprehend a language, a human must understand its syntactic structure, the meaning each symbol represents, and the interaction between the two. In most languages, syntactic structure conveys something about the semantics of the symbols, and the semantics of symbols may constrain valid syntactic realizations. As a simple example: when we see a noun following a number in English (e.g. "one book"), we can infer that the noun is countable. Conversely, if it is known that a noun is countable, a speaker of English knows that it can plausibly be preceded by a numeral. It is therefore reasonable to assume that for a computer system to successfully process natural language, it has to be equipped with capabilities to represent and utilize both the syntactic and semantic information of the language simultaneously.

The n-gram language model (LM) is a powerful and popular framework for capturing the word order information of language, or fundamentally syntactic information. It has been applied successfully to a variety of NLP problems such as machine translation, speech recognition, and optical character recognition.

As described in equation (1), an n-gram language model utilizes conditional probabilities to capture word order information, and the validity of a sentence can be approximated by the accumulated probability of the successive n-gram probabilities of its constituent words $W_1 ... W_k$.

$$Validity(W_1 W_2 ... W_k) = \prod_{i=1}^{k} P(W_i | W_{i-n+1} ... W_{i-1}) \quad (1)$$

As powerful as a traditional n-gram LM can be, it does not capture the semantic information of a language. Therefore it has seldom been applied to semantic problems such as word sense disambiguation (WSD). To address this limitation, in this paper we propose to expand the formulation of a LM to include not only the words in the sentences but also their semantic labels (e.g. word senses). By incorporating semantic information into a LM, the framework is applicable to problems such as WSD, semantic role labeling, and even more generally machine translation and information extraction – tasks that require both semantic and syntactic information for an effective solution.

The major advantage of our algorithm compared to conventional unsupervised WSD is that it can perform WSD without need for any sense glosses, sense-similarity measures, or other linguistic information as has been required in many other unsupervised WSD systems. We need only an unannotated corpus plus a sense dictionary for which some senses of different words have been "pooled" together into something like a WordNet synset, as we exploit the redundancy of sense sequences even where the words may differ. Therefore, our approach can be applied in the early stages of sense invention for a language or domain, where only limited lexical semantic resources are available.

2 Incorporating and Learning Semantics in a LM

The first part of this section proposes a semantics-enhanced language model framework while the second part discusses how its parameters can be learned without annotated data.

2.1 A Semantics-Enhanced Language Model

Figure 1(a) is a general finite state representation of a sentence of four words ($W_1 ... W_4$) connected through a bigram LM. Each word can be regarded as a state node and the transition probabilities between states can be modeled as the n-gram conditional probabilities of the involved states (here we assume the transition probabilities are bigrams). In fact each word in a sentence has a certain lexical meaning (sense or semantic label, S_i) as represented in Figure 1(c). Conceptually, for each word-based finite state representation there is a dual representation in the semantics (or sense) domain, as shown in 1(b). A Semantic Language Model (or SLM) like 1(b) records the order relations between senses. Alternatively, one can combine both representations into a hybrid language model that captures both the word order information and the word meaning, as demonstrated in 1(d). 1(d) represents a Word-Sense Language Model

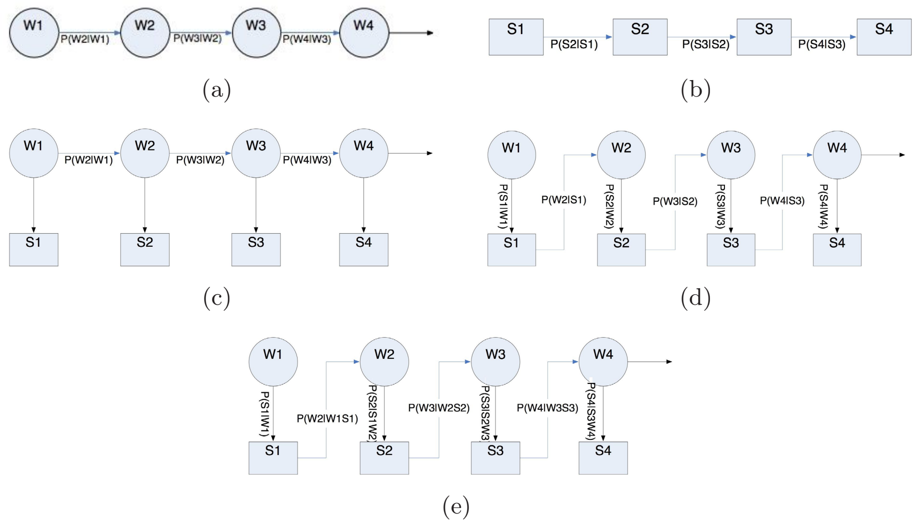


Fig. 1. (a) A standard finite-state, bigram LM representation of a sentence. (b) a Semantic Language Model. (c) each word in the sentence has a certain meaning (or semantic label). (d) a hybrid LM integrating word and sense information (WSLM). (e) like (d) except that a trigram model is used.

(or WSLM), a semantics-enhanced LM incorporating two types of states: word symbols and their semantic labels. The intuition behind WSLM is that when processing a word, people first try to recognize its meaning (i.e. $P(S_n|W_n)$), and based on that predict the next word (i.e. $P(W_{n+1}|S_n)$). Figure 1(e) is the same as 1(d) except that the bigram probabilities are replaced by trigrams. It embodies the concept that the next word to be revealed depends on the previous word together with its semantic label, and the meaning of the current word depends on not only the word itself but the meaning of the previous word.

The major reason for the success of a LM approach to NLP problems is its capability of predicting the validity of a sentence. In 1(a), we can say that a sentence $W_1W_2W_3W_4$ is valid because $P(W_2|W_1) * P(W_3|W_2) * P(W_4|W_3)$ is relatively high. Similarly, given that the semantic labels of each word in the sentence are known, the probabilities $P(S_2|S_1) * P(S_3|S_2) * P(S_4|S_3)$ can be applied to assess the semantic validity of this sentence as well. Furthermore, we can say that a word sequence together with its semantic assignment (interpretation) is valid based on a WSLM if the probability of $P(S_1|W_1) * P(W_2|S_1) * \ldots * P(W_4|S_3) * P(S_4|W_4)$ is high. We can therefore use a semantics-enhanced LM to rank possible interpretations of a word sequence.

2.2 Unsupervised Parameter Learning

The n-gram probabilities of a word-based LM such as the transition probabilities in Figure 1(a) can be easily learned through counting term frequencies and

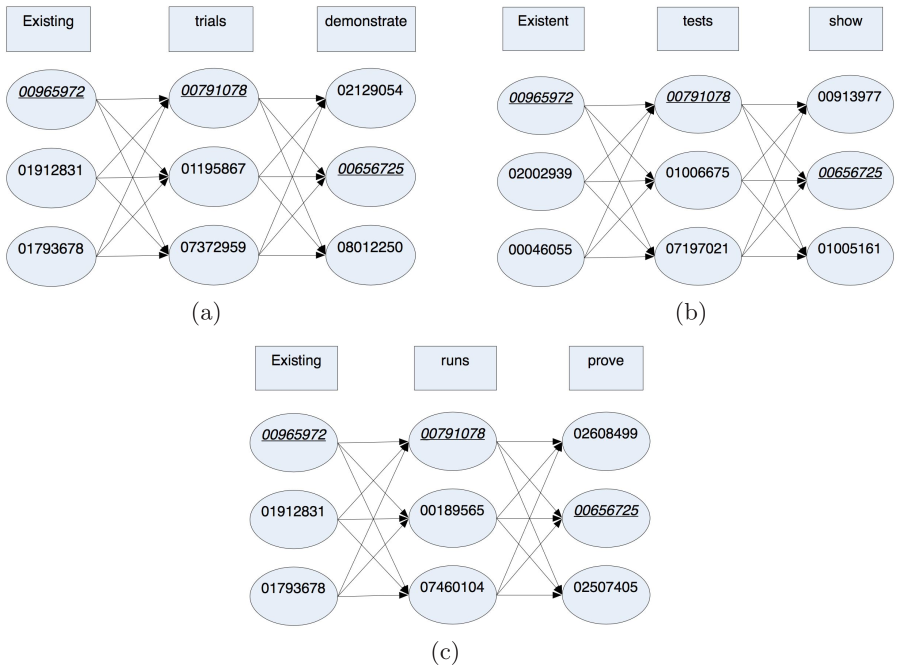


Fig. 2. Sense-based graph of word sequences (a) "Existing trials demonstrate..." (b) "Existent tests show..." (c) "Existing runs prove..."

co-occurrences from large corpora. If there were some large corpora with semantically annotated words and sentences, we could learn the semantics-enhanced LM such as 1(b) and 1(d)-1(e) directly through frequency counting as well. Unfortunately, there is no corpus containing a significant amount of semantically annotated data available. To address this problem, we discuss below an approach that allows the system to approximate the n-gram probabilities of the semantics-enhanced language models. Without loss of generality, in the following discussion we assume the transition probabilities to be learned are all bigrams.

The problem setup is as follows: the system is given a plain text, unannotated corpus together with a dictionary (assuming WordNet 2.1) that contains a list of possible semantic labels for each word. Using these resources alone, the system must learn the n-gram dependencies between semantic labels. Note that every word in the WordNet dictionary has at least one sense (or synset label), and each sense has a unique 8-digit id representing its database location. Different words can share synsets, indicating they have meanings in common. For example, the word *trial* has six senses in the dictionary and one of these (id=00791078) is shared by the word *test* and *run*. The word *demonstrate* has four meanings where one of them (id=00656725) is associated with the words *prove* and *show*. To learn a SLM, one has to learn the conditional probabilities of one sense following the other such as $P(S_k = 00656725|S_{k-1} = 00791078)$.

The first step of learning is to construct a sense-based graph representation for the plain text corpus by connecting all the senses of each word to the senses of the subsequent word. For example, Figure 2(a) is the sense-graph of the phrase "Existing trials demonstrate". For illustration purposes we display only three senses per word in the figure, though there may be more senses for the word defined in WordNet. The weights of the links in the graph, based on the concept of a LM, can be modeled by the n-gram (e.g. bigram) probabilities. If all the bigrams between senses in the graph are known, then for each path of senses (where a path contains one sense per word) we can generate its associated probability, as in equation (2). Note that if a word has no known senses in WordNet (e.g. for closed class words or proper nouns) we assign it a single "dummy" sense.

$$\begin{aligned} &Validity(existing = 00965972, trial = 00791078, demonstrate = 02129054) \\ &= Pr(00965972|start) * Pr(00791078|00965872) * Pr(02129054|00791078) \end{aligned} \tag{2}$$

This probability reflects the cumulative validity of each sense assignment for the sequence of words. One can rank all the sense paths based on their probabilities to find the optimal assignment of senses to words. If the associated probability for each path in the graph is given, we can apply a technique called *fractional counting* to determine bigram probabilities. Fractional counting counts the occurrence of each bigram in all possible paths, where the count is weighted by the associated probability of the path.

Unfortunately, without a sense-annotated corpus neither the sense bigrams nor the path probabilities can be known directly. However, since computing the likelihood for each path and generating the bigram probabilities are dual problems (i.e. one can be generated if the other is known), it is possible to apply the expectation-maximization (EM) algorithm to approximate both numbers [8]. EM is an efficient iterative procedure for computing the Maximum Likelihood (ML) estimate in the presence of missing data. It estimates the model parameters for which the observed data are most likely, using an iteration of two processes: the E-step, in which the missing data are estimated using conditional expectation given the observed data and the current estimate of the model parameters, and the M-step, in which the likelihood function is maximized under the assumption that the missing data are known (using the estimate of the missing data from the E-step).

To perform the EM learning, the first step is to initialize the probabilities of the bigrams. As will be shown in our case study, the initialization can be uniformly distributed or use certain preexisting knowledge. In the Expectation stage (E-step) of the EM algorithm, the system uses the existing bigram probabilities to generate the associated probability of each path, such as the one shown in equation 2. In the maximization stage (M-step) the system applies fractional counting to refine the bigram probabilities. It is guaranteed that the refined bigram can produce a higher probability for the observed data. The E-step and M-step continue to iterate until a local optimum is reached.

One potential problem for this approach is efficiency. The total number of paths in the graph grows exponentially with the number of words (i.e. $O(b^n)$, where n is the number of words and b is the average branching factor of nodes, i.e. the average number of senses per word). Therefore it is computationally prohibitive for the system to enumerate all paths and produce their associated probabilities one by one to perform fractional counting. Fortunately in this situation one can apply a polynomial algorithm called Baum-Welch (or *forward-backward*) algorithm for fractional counting [2]. Rather than generating all paths with their probabilities in the graph, we need to know only the total probability of all the paths that a link (bigram) occurs in. This can be generated by recording dynamically for each link the accumulated probabilities from the beginning of the graph (the alpha value) to the link and the accumulated probabilities from the link to the end (the beta value). Since in our case the alpha and beta values are independent, it is possible to generate all n-grams with polynomial time $O(nb^2)$ and space $O(nb)$. A similar approach has been applied successfully to unsupervised NLP problems such as tagging, decipherment, and machine translation ([7], [12], [13], [14]).

The simple example shown in Figure (2) describes the intuition behind the method. Imagine the system encounters the phrases "Existing trials demonstrate", "Existent tests show", "Existing runs prove" in the corpus. According to Figure (2) there is one common sense 00965972 for the words *existing* and *existent*, a single common sense 00791078 for *trial*, *test*, and *run*, and a common sense 00656725 for the words *demonstrate*, *show*, and *prove*. Based on the minimum description length principle (or Occams Razor), a reasonable hypothesis is that these three senses should have higher chance to be the right assignments (and thus should appear successively more often) compared with the other candidates, since one can then use only three senses to "explain" all the sentences.

The proposed learning algorithm captures the spirit of this idea. Assuming equal probabilities are used to initialize the bigrams and assuming all senses listed in Figure (2) do not appear elsewhere in the corpus, then after the first iteration of EM, 00791078 will have a higher chance to follow 00965972 compared with others (e.g. equation (3)). This is because the system sees 00791078 following 00965972 more times than others in the fractional counting stage.

$$Pr(00791078|00965972) > Pr(00189565|00965972) \tag{3}$$

This approach works because there are situations in which multiple words can be used to express a given meaning, and people tend not to choose the same word repeatedly. The system can take advantage of this to learn information about senses that tend to go together from the shared senses of these varied words, as formalized in the semantics-enhanced LM.

The same approach can be applied to learn the parameters in a WSLM. The only difference is that the words are included in the graph as single-sense nodes. Figure 3 is the graph presentation of a WSLM.

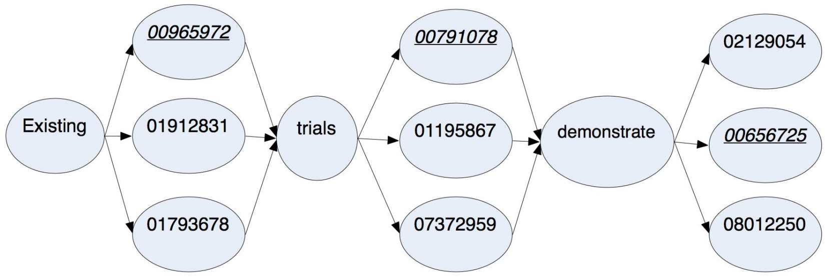


Fig. 3. The graph generated for the WSLM. Such a network has the format word1→sense1→ word2→sense2→...

3 Unsupervised WSD Using SLM and WSLM

We describe a case study on applying the SLM and WSLM to perform an all-words word sense disambiguation task. Since both language models are trained without sense-annotated data, this task is an unsupervised WSD task.

3.1 Background

Unsupervised WSD aims at determining the senses of words in a text without using a sense-annotated corpus for training. The methods employed generally fall into two categories, one for all-words, token-based WSD (i.e. assign each token a sense in its individual sentential context) and the other to find the most frequent sense of each unique token in the text as a whole (following a one sense per discourse assumption). The motivation to focus on the second type of task is that assigning the most frequent sense to every word turns out to be a simple heuristic that outperforms most approaches [11]. The following paragraphs describe the existing unsupervised WSD methods.

Banerjee and Pedersen proposed a method that exploits the concept of gloss overlap for WSD [1], where a gloss is a sentence in WordNet that characterizes the meaning of a sense (synset). It assumes the sense whose gloss definition looks most similar (overlaps strongly) with the glosses of surrounding content words is the correct one. Mihalcea's graph-based algorithm [16] first constructs a weighted sense-based graph , where weights are the similarity between senses (e.g. gloss overlap). Then it applies PageRank to identify prestigious senses as the correct interpretation. Galley and McKeown also propose a graph-based approach called lexical chains that regards a sense to be dominant if it has more strong connections with its context words [9]. The strength of connection is determined by the type of relation as well as the distance between the words in the text. Navigli and Velardi propose a conceptually similar but more knowledge-intensive approach called structural semantic interconnections (SSI) [17]. For each sense, the method first constructs semantic graphs consisting of collocation information (extracted from annotated corpora), WordNet relation information, and domain labels. Using these graphs, the algorithm iteratively chooses senses with strong connectivity to the relevant senses in the semantic graph. McCarthy *et al* propose a method to determine the most frequent senses for words [15]. In their framework, the distributionally similar neighbors of each word are determined, and a sense of a word is regarded as dominant if it is the most similar to the senses of its distributionally similar neighbors.

Although the above methods approach the unsupervised WSD problem from different angles, they do each take advantage of semantic similarity measures derived from an existing knowledge resource (WordNet). While we are not arguing the legitimacy of this strategy, we believe there is another type of information that a system can benefit from to determine the sense of words, specifically word and sense order information. Furthermore, the strategy we propose allows the system to be deployed in environments where semantic similarity among senses cannot be determined *a priori.* The only requirement in our approach is that there exist multiple words mapped to a single concept in a sense inventory.

Based on this alternative strategy even the non-content words such as stop words (ignored in existing approaches) can be helpful. Considering the sentence "He went into the bank beside the river", most of the above approaches will likely choose the *river bank* ($bank\#2$) sense for bank instead of the correct *financial institute* ($bank\#1$) sense, because the former sense is semantically closer to the only other content word *river*. However, even without other context information, it is not hard for an English speaker to realize the financial bank is more likely to be the correct one, since people do not usually go into a river bank. A somewhat accurate SLM can guide the system to make this decision since it shows $P(bank\#1|into\#1the\#1) \gg P(bank\#2|into\#1the\#1)$.

Such information can be learned in an unsupervised manner if the system sees similar sentences such as "He went into a banking-company" (where *banking-company* has $bank\#1$ sense in WordNet 2.1). Also consider the sentence "The tank has an empty tank". Again it would not be trivial for the previously described algorithms to realize these two *tank*s have different meanings since their frameworks (explicitly or implicitly) imply or result in one sense per sentence. However, an accurate semantics-enhanced language model can tell us that the *tank as container* sense has higher chance to follow the word *empty* while the *tank as the army tank* sense has higher chance to be followed by *has.*

3.2 System Design and Experiment Setup

We applied both bigram SLM and WSLM to perform unsupervised WSD. Our WSD system can be divided into three stages. The first stage is the initialization stage. In SLM, we need to initialize $P(S_{k+1}|S_k)$ and in WSLM there are two types of probabilities to be initialized: $P(S_k|W_k)$ and $P(W_{k+1}|S_k)$. We explore here two different ways to initialize the LMs without any *a priori* knowledge of the probability distribution of senses. The second stage is the learning stage, using the EM algorithm together with forward-backward training to learn the bigrams. The final stage is the decoding stage, in which the learned bigrams are utilized to identify the senses of words in their sentential context that optimize the total probability. Using the dynamic programming method, the overall time

Table 1. The results for all-words unsupervised WSD on SemCor using SLM and WSLM based on uniform and node-frequency initialization

Initialization	Corpus	Baseline (%)	SLM (%)	WSLM (%)
Uniform	SC	17.1	31.8	27.7
Uniform	SC+BNC	17.1	32.3	28.8
Graph Freq	SC	17.1	25.1	34.0
Graph Freq	SC+BNC	17.1	36.0	34.6

complexity for the system is only linear to the number of words and quadratic to the average number of senses per word.

We tested our system on SemCor (SC) data, which is a sense-annotated corpus that contains a total of 778K words (where 234K have sense annotations). We use SemCor and British National Corpus (BNC) sampler data (1.1 million words) for training. In the EM algorithm initializations reported on below, no external knowledge other than the unannotated corpus and the sense dictionary is exploited. The experimental setup is as follows: we first determine the baseline performance on the WSD task using only the initial knowledge (i.e. without applying language models). Then we train a semantics-enhanced LM based on the initialization and use it to perform decoding. Our model is evaluated by checking how much the learned LM can improve the accuracy over the baseline.

Initialization: Uniform N-Gram Probabilities. The baseline for this case is a random sense assignment for all-words WSD (i.e. disambiguation of all word tokens) in SemCor, resulting in 17% accuracy on the test set. The initialization simply assigns equal probability to all bigrams. As shown in Table 1, the results improve to 32.3% for SLM and 28.8% for WSLM after training on a corpus consisting of the SemCor texts plus texts from the BNC Sampler.

Initialization: Graph Frequency. The second initialization is based on the node occurrence frequency in the sense graph. That is, $Pr(S_1|S_2) = gf(S_1)$ for SLM and $Pr(S_1|W_1) = gf(S_1)$ for WSLM , where $gf(S_1)$ represents the frequency of a node S1 in the sense graph, or its graph frequency (for example, in Figure 2 00965972 appears three times). The intuition behind this initialization is that a sense should have a higher chance to appear if it occurs in multiple words that frequently occur in the text. Again, to count the node frequency we do not need any extra knowledge since the graph itself can be generated based on only the corpus and the dictionary. This initialization improves the accuracy to 36.0% for SLM and 34.6% for WSLM.

These experiments show that learned syntactic order structure can tell us much about the sense of a word in context, in the absence of external knowledge.

3.3 Discussion

The case study on applying semantics-enhanced LM to WSD reveals two important facts. The first is that syntactic order information for words and senses can

Table 2. Comparison between LM-based approaches, semantic approaches and semantics-enhanced LM approaches for all-nouns Unsupervised WSD

Initialization	Corpus	SLM (%)	WSLM (%)
Uniform	SC+BNC	35.6	32.3
gloss overlap		36.5	
Graph Freq	SC+BNC	29.6	38.0
SSI		42.7	

benefit WSD. This conclusion to some extent echoes the concept of syntactic semantics [18], which claims that semantics are embedded inside syntax. The second conclusion is that the unsupervised learning method proposed in this paper does learn a sufficient amount of meaningful semantic order information to allow the system to improve disambiguation quality. It follows from this that the framework is flexible enough to be trained on a domain-specific corpus to obtain a SLM or WSLM specifically for that domain. This has important potential applications in domains with senses not represented in resources such as WordNet.

Table 2 shows how different types of knowledge perform in WSD. We compare our system with two existing WSD systems on the all-nouns WSD task (that is, evaluating disambiguation performance only on nouns in the corpus): Banerjee and Pedersen's gloss overlap system and the SSI system (we limit ourselves to the all-nouns task as these are the results as reported in [4]). The LM-based approach without preliminary knowledge performs right in between gloss overlap and SSI approaches in predicting the nouns in SemCor. This is interesting and informative since the results demonstrate that by using only word order information and no lexical semantic information (e.g. sense similarity), we still generate competitive WSD results. Comparing Table 2 with Table 1, one can also infer that WSD on nouns is an easier task than on other parts of speech.

One advantage of our model is that it could incorporate any amount of supervised information in the initialization step. A small amount of annotated data can be used to generate the initial n-grams to be refined through EM. This would certainly result in significant improvements over the knowledge-poor experiments presented here. Given the performance of our system relative to the more knowledge-intensive approaches, that approach would also be likely to result in an overall improvement over those results since it incorporates an additional source of linguistic information.

4 Related Work

There have been previous efforts in incorporating semantics into a language model. Brown *et al* proposed a class-based language model that includes semantic classes in a LM [5]. Bellegarda proposes to exploit latent semantic analysis to map words and their relationships with documents into a vector space [3]. Chueh *et al* propose to combine semantic topic information with n-gram LM using the maximum entropy principle [6]. Griffiths *et al* also propose to integrate topic semantic information [10] into syntax based on a Markov chain Monte Carlo method.

The major difference between our model and these is that we propose to learn semantics at the word level rather than at the document or topic level. Consequently the models are different in the parameters to be learned (in the other models, the topic usually determines words to be used while in our model the words can determine senses), preliminary knowledge incorporation (e.g. [5] used a fully connected word-class mapping during initialization) and most importantly, the applications. Other systems were evaluated on word clustering or document classification while we have made the first attempt to apply a semantics-enhanced LM to a fine-grained semantic analysis task, namely word sense disambiguation.

5 Conclusion and Future Work

There are three major contributions in this paper. First we propose a framework that enables us to incorporate semantics into a language model. Second we show how such a model can be learned efficiently ($O(nb^2)$ time) in an unsupervised manner. Third we demonstrate how this model can be used to perform the WSD task in knowledge-poor environments. Our experiments also suggest that WSD can be a suitable platform to evaluate the semantic language models, and that using only syntactic information one can still perform WSD as well as using conventional semantic (e.g. gloss) information.

There are two main future directions for this work. In terms of the model itself, we would like to integrate additional knowledge into the initialization, to take advantage of existing *a priori* knowledge, specifically sense frequency information derived from WordNet (which orders senses by frequency), as well as using the semantic hierarchy in WordNet to smooth probabilities in the language model. We would also like to investigate how much the results can be improved based on higher n-gram models (e.g. trigram). In terms of applications we would like to investigate whether the model can be applied to other natural language processing tasks that generally require both syntactic and semantic information such as information extraction, summarization, and machine translation.

References

1. Banerjee, S., Pedersen, T.: Extended gloss overlaps as a measure of semantic relatedness. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pp. 805–810 (2003)
2. Baum, L.E.: An Inequality and Associated Maximization in Statistical Estimation for Probabilistic Functions of Markov Processes. Inequalities 627(3), 1–8 (1972)
3. Bellegarda, J.: Exploiting latent semantic information in statistical language modeling. Proceedings of IEEE 88(8), 1279–1296 (2000)
4. Brody, S., Navigli, R., Lapata, M.: Ensemble Methods for Unsupervised WSD. In: Proceedings of the ACL/COLING, pp. 97–104 (2006)

5. Brown, P.F., et al.: Class-based n-gram models of natural language. Computational Linguistics 18(4), 467–479 (1992)
6. Chueh, C.H., Wang, H.M., Chien, J.T.: A Maximum Entropy Approach for Semantic Language Modeling. Computational Linguistics and Chinese Language Processing 11(1), 37–56 (2006)
7. Cutting, D., et al.: A practical part-of-speech tagger. In: Proceedings of ANLP-1992, Trento, Italy, pp. 133–140 (1992)
8. Dempster, A.D., Laird, N.M., Rubin, D.B.: Maximum likelihood for incomplete data via the EM algorithm. Journal of Royal Statistical Society Series B 39, 1–38 (1977)
9. Galley, M., McKeown, K.: Improving Word Sense Disambiguation in Lexical Chaining. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pp. 1486–1488 (2003)
10. Griffiths, T., et al.: Integrating Topics and Syntax. In: Proceedings of the Advances in Neural Information Processing Systems (2004)
11. Hoste, V., et al.: Parameter optimization for machine-learning of word sense disambiguation. Language Engineering 8(4), 311–325 (2002)
12. Knight, K., et al.: Unsupervised Analysis for Decipherment Problems. In: Proceedings of the ACL-COLING (2006)
13. Koehn, P., Knight, K.: Estimating word translation probabilities from unrelated monolingual corpora using the EM algorithm. In: Proceedings of the AAAI, pp. 711–715 (2000)
14. Lin, S.d., Knight, K.: Discovering the linear writing order of a two-dimensional ancient hieroglyphic script. Artificial Intelligence 170(4-5) (2006)
15. McCarthy, D., et al.: Finding predominant word senses in untagged text. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain (2004)
16. Mihalcea, R.: Unsupervised Large-Vocabulary Word Sense Disambiguation with Graph-based Algorithms for Sequence Data Labeling. In: Proceedings of the Joint Conference on Human Language Technology / Empirical Methods in Natural Language Processing (HLT/EMNLP) (2005)
17. Navigli, R., Velardi, P.: Structural Semantic Interconnections: a Knowledge-Based Approach to Word Sense Disambiguation. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 27(7), 1063–1074 (2005)
18. Rapaport, W.J.: Holism, Conceptual-Role Semantics, and Syntactic Semantics. Minds and Machines 12(1), 3–59 (2002)

Discovering Word Senses from Text Using Random Indexing

Niladri Chatterjee[1] and Shiwali Mohan[2]

[1] Department of Mathematics, Indian Institute of Technology Delhi, New Delhi, India 110016
niladri@maths.iitd.ac.in
[2] Yahoo! Research and Development India, Bangalore, India 560 071
shiwali@yahoo-inc.com

Abstract. Random Indexing is a novel technique for dimensionality reduction while creating Word Space model from a given text. This paper explores the possible application of Random Indexing in discovering word senses from the text. The words appearing in the text are plotted onto a multi-dimensional Word Space using Random Indexing. The geometric distance between words is used as an indicative of their semantic similarity. Soft Clustering by Committee algorithm (CBC) has been used to constellate similar words. The present work shows that the Word Space model can be used effectively to determine the similarity index required for clustering. The approach does not require parsers, lexicons or any other resources which are traditionally used in sense disambiguation of words. The proposed approach has been applied to TASA corpus and encouraging results have been obtained.

1 Introduction

Automatic disambiguation of word senses has been an interesting challenge since the very beginning of computational linguistics in 1950s [1]. Various clustering techniques, such as bisecting K-means [2], Buckshot [3], UNICON [4], Chameleon [5], are being used to discover different senses of words. These techniques constellate words that have been used in similar contexts in the text. For example, when the word 'plant' is used in the living sense, it is clustered with words like 'tree', 'shrub', 'grass' etc. But when it is used in the non-living sense, it is clustered with 'factory', 'refinery' etc. Similarity between words is generally defined with the help of existing lexicons, such as WordNet [6], or parsers (e.g. Minipar [7]).

Word Space model [8] has long been in use for semantic indexing of text. The key idea of Word Space model is to assign vectors to the words in high dimensional vector spaces, whose relative directions are assumed to indicate semantic similarity. The Word Space model has several disadvantages: *sparseness* of the data and *high dimensionality* of the semantic space when dealing with real world applications and large size data sets. Random Indexing [9] is an approach developed to deal with the problem of high dimensionality in Word Space model.

In this paper we attempt to show that the Word Space model constructed using Random Indexing can be used efficiently to cluster words, which in turn can be used for disambiguating the sense of the word. In a Word Space model the geometric distance between the words is indicative of the semantic similarity of the words. We use soft Clustering by Committee (CBC) [7] algorithm to congregate similar words. It can discover the less frequent senses of a word, and thereby avoids discovering duplicate senses. The typical CBC algorithm uses resources, such as MiniPar, to determine similarity between words**.** However, the advantage of Random Indexing is that it uses minimal resources. Here similarity between the words is determined based on their usage in the text, and this eliminates the need of using any lexicon or parser. Further, Random Indexing helps in dimensionality reduction as it involves simple computations e.g. vector addition and thus is less expensive than other techniques such as Latent Semantic Indexing [10].

2 The Word Space Model and Random Indexing

The meaning of a word is interpreted by the context it is used in. Word Space model [8] is a spatial representation of word meaning. In the following subsections we describe these models.

2.1 Word Space Model

In this model the complete vocabulary of any text (containing *n* words) can be represented in an *n*-dimensional space in which each word occupies a specific point in the space, and has a vector associated with it defining its meaning. The words are placed on the Word Space model according to their distributional properties in the text, such that:

1. The words that are used within similar group of words (i.e. in similar *context*) should be placed nearer to each other.
2. The words that lie closer to each other in the word space have similar meanings, while the words distant in the word space are dissimilar in their meanings.

2.1.1 Vectors and Co-occurrence Matrix – Geometric Representation of Distributional Information

A *context* of a word is understood as the linguistic surrounding of the word. As an illustration, consider the following sentence: *A friend in need is a friend indeed.*

Table 1. Co-occurence matrix for the sentence *A friend in need is a friend indeed*

	Co-occurrents					
Word	a	friend	in	need	is	indeed
a	0	2	0	0	1	0
friend	2	0	1	0	0	1
in	0	1	0	1	0	0
need	0	0	1	0	1	0
is	1	0	0	1	0	0
indeed	0	1	0	0	0	0

If we define the context of a word as one preceding and one succeeding word, then the context of '*need*' is '*in*' and '*is*', and the context of '*a*' is '*is*' and '*friend*'. To tabulate this context information a co-occurrence matrix of the following form is created, in which the $(i, j)^{th}$ element denotes the number of times word i occurs in the context of word j within the text. Here, the context vector for 'a' is [0 2 0 0 1 0] and for 'need' is [0 0 1 0 1 0], determined by the corresponding rows of the matrix.

A context vector thus obtained can be used to represent the distributional information of the word into the geometrical space. This is similar to each word being assigned a unique unary vector of dimension six (here), called *index vector*. The context vector for a word can be obtained by summing up the index vectors of the words on the either side of it. An index vector [0 0 0 0 1 0] can be assigned to the word *'is'* ; and the word *'in'* can be assigned an index vector [0 0 1 0 0 0]. These two in turn can be summed up to get the context vector for *'need'* [0 0 1 0 1 0].

2.1.2 Similarity in Mathematical Terms

Context vectors give the location of the word in the word space. In order to determine how similar the words are in their meaning, a similarity measure has to be defined. Various schemes, e.g. scalar product of vectors, Euclidean distance, Minkowski metrics [9], are used to compute similarity between vectors corresponding to the words. We have used cosine of the angles between the two vectors x and y to compute normalized vector similarity. The cosine angle between vectors x and y is defined as:

$$sim_{\cos}(x, y) = \frac{x.y}{|x||y|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \tag{1}$$

The cosine measure is the most frequently utilized similarity metric in word-space research. The advantage of using cosine metric over other metrics is that it provides a fixed measure of similarity, which ranges from 1 (for identical vectors), to 0 (for orthogonal vectors) and -1 (for vectors pointing in the opposite directions). Moreover, it is also comparatively efficient to compute.

2.1.3 Problems Associated with Implementing Word Spaces

The dimension n used to define the word space corresponding to a text document is equal to the number of unique words in the document. Therefore, the number of dimensions increases as the size of text increases. Thus computational overhead increases rapidly with the size of the text. The other problem is of data sparseness. The majority of cells in the co-occurrence matrix constructed corresponding to a document will be zero. The reason is that most of the words in any language appear in limited contexts only, i.e. the words they co-occur with are very limited. While dimensionality reduction does make the resulting lower-dimensional context vectors easier to compute, it does not solve the problem of initially having to collect a potentially huge co-occurrence matrix. Even implementations, such as Latent Semantic Analysis [11], that use powerful dimensionality reduction, need to initially collect the words-by-documents or words-by-words co-occurrence matrix. Random

Indexing (RI) described below removes the need for the huge co-occurrence matrix. Instead of first collecting co-occurrences in a matrix and then extracting context vectors from it, RI incrementally accumulates context vectors, which can then be assembled into a co-occurrence matrix.

2.2 Random Indexing

Random Indexing [9] is based on Pentti Kanerva's [11] work on sparse distributed memory. Random Indexing accumulates context vectors in a two step process:

1. Each word in the text is assigned a unique and randomly generated vector called the *index vector*. The index vectors are sparse, high dimensional and ternary (i.e. 1, -1, 0). Each word is also assigned an initially empty *context vector* which has the same dimensionality (r) as the index vector.
2. The context vectors are then accumulated by advancing through the text one word taken at a time, and then adding the context's index vector to the focus word's context vector. When the entire data is processed, the r-dimensional context vectors are effectively the sum of the words' contexts.

For illustration we again take the example of the sentence *'A friend in need is a friend indeed'*. Let the dimension r of the index vector be 10, and the context be defined as one preceding and one succeeding word.

Let *'friend'* be assigned a random index vector: [0 0 0 1 0 0 0 0 -1 0], and *'need'* be assigned a random index vector: [0 1 0 0 -1 0 0 0 0 0]. Then to compute the context vector of *'in'* we need to sum up the index vectors of its context. Since the context is defined as one preceding and one succeeding word, the context vector of *'in'* is the sum of index vectors of *'friend'* and *'need'* , and is equal to [0 1 0 1 -1 0 0 0 -1 0].

If a co-occurrence matrix has to be constructed, r-dimensional context vectors can be collected into a matrix of order [w, r], where w is the number of unique words, and r is the chosen dimensionality of each word. Note that this is similar to constructing an n-dimensional unary context vector that has a single 1 in different positions for different words, and n is the number of distinct words. These n dimensional unary vectors are orthogonal, whereas the r-dimensional random index vectors are nearly orthogonal [13]. Choosing RI is an advantageous tradeoff between the number of dimensions and orthogonality, as the r-dimensional random index vectors can be seen as approximations of the n-dimensional unary vectors. The context vectors computed on the language data are used in mapping the words onto the word space.

Compared to other Word Space methodologies, Random Indexing approach gives us the following advantages:

First, it is an incremental method, i.e. the context vectors can be used for similarity computations even when only a small number of examples have been encountered. By contrast, most other word space methods require the entire data to be sampled before similarity computations can be performed.

Second, it uses fixed dimensionality, which means that new data does not increase the dimensionality of the vectors. Increasing dimensionality can lead to significant scalability problems in other word space methods.

Third, it uses implicit dimension reduction, since the fixed dimensionality is much lower than the number of words in the data. This leads to a significant gain in processing time and reduction in memory consumption. Typically, the dimension r is about 10% of n, the number of unknown words.

3 Clustering by Committee

Clustering by Committee [7] has been specially designed for Natural Language Processing purposes. The hard version of CBC, in which a word is assigned to exactly one cluster, is typically used for document retrieval. We use a soft version of the above in word sense disambiguation. The advantage thereby is that, it allows fuzzy clustering, and consequently, the words are assigned to more than one cluster, perhaps with varying degree of membership. It consists of three phases:

Phase I - Generation of a similarity matrix
Here, a similarity index [7] between words is defined, and a similarity matrix is generated from the words appearing in the text, in which each cell contains the numerical value of the similarity between any pair of words.

Phase II - Formation of committees
The second phase of the clustering algorithm takes in as input the list of words to be clustered and the similarity matrix. It recursively finds tight clusters called *committees,* scattered in the similarity space. Each committee can be thought of as the representation of a context or a sense. Each committee is assigned a centroid vector which is the average of the vectors of the words contained by them. The algorithm tries to form as many committees as possible on the condition that each newly formed committee is not very similar to any existing committee (i.e. the similarity is not more than a given threshold θ_1). If the condition is violated, the committee is discarded. The similarity between two committees is computed by determining the cosine metric between the centroid vectors of the respective committees. Next, it identifies the residue words that are not covered by any committee. A committee is said to *cover* a word if the word's similarity to the centroid of the committee exceeds some high similarity threshold (i.e. greater than another given threshold θ_2). The algorithm then attempts to find recursively more committees among the residue words. The output of the algorithm is the union of all committees found in each recursive step. Committees are the cores of the clusters to which words are successively added in Phase III as explained below. The committees do not change after their formation.

Phase III – Assigning of word to its most similar committee
In the final phase of the algorithm each word is assigned to its most similar clusters. The word is assigned to a cluster if its similarity to the committee that forms the core of the cluster exceeds a given threshold σ. The cluster now represents the context the word has been used in. Once a word has been assigned to a cluster, the centroid of the committee is subtracted from the context vector of the word. This enables the

algorithm to find the less frequent context of the word. This phase is similar to K-means clustering. Like K-means the words are assigned to clusters whose centroids are closest to the word. However, unlike K-means clustering, the word is not added to the committee itself, but to the cluster surrounding it, so the centroids remain constant. This differentiates it from K-means clustering algorithm.

Once an element is assigned to its most similar cluster, the centroid of the committee is subtracted from the context vector of the concerned word. The context vector of the word, as discussed in Section 3.2, is the sum of all the contexts the word may have appeared in, in the text. If one of the contexts is removed from the context vector of the word, the similarity of the word with other committees increases thus allowing the algorithm to discover other less frequent senses of the word.

4 Experimental Setup

We conducted our experiments on the TASA corpus [8]. The corpus is 56 Mb in size and contains 10,802,187 unique words after stemming. It consists of high school level English text and is divided into paragraphs on various subjects, such as science, social studies, language and arts, health and business. The paragraphs are of 150-200 words each. To use them in our experiments we did the following preprocessing: The words appearing in the text were stemmed using Porter's stemming algorithm [13] to reduce the word to their base forms. This reduced the number of word forms considerably. Stop words [14] such as *is, are, has, have* etc. were removed from the text as these words do not contribute to the context.

4.1 Mapping of Words onto Word Space Model

Each word in the document was initially assigned a unique randomly generated index vector of the dimension r = 1500, with ternary values (1, -1, 0). The index vectors were so constructed that each vector of 1500 units contained eight randomly placed 1s and eight randomly placed -1s, rest of the units were assigned 0. Each word was also assigned an initially empty context vector of dimension 1500. We conducted experiments with different context window sizes. The results are presented in Section 5. The context of a given word was restricted in one sentence, i.e. windows across sentences were not considered. In case where the window is extended in the preceding or the succeeding sentence, a unidirectional window was used. Once the context vectors for the words were obtained, similarity between words was determined by computing the cosine metric between the corresponding context vectors.

4.2 Implementation of Clustering Algorithm

Once we acquired the semantic similarities of the words in the text, we ran soft Clustering by Committee algorithm to find the committees present in the text and to assign words to their most similar committees. Fig. 1 gives a description of the algorithm. The experiments were conducted for varying values of parameters θ_1, θ_2, σ (defined in Section 3), and results are presented in Section 5.

```
Phase 1
let E be the list of unique words (n) in the text
S be the similarity matrix (n X n)
  assign values to S(i,j), by computing the cosine metric between
 E(i) and E(j)

Phase 2
let S be the similarity matrix generated from phase 1
let E be the list of words to be clustered
let C be the list of committees
discover_committees (S, E,$\theta_1$,$\theta_2$)
{
for each element e Є E {
             cluster elements for S using average link clustering}
for each discovered cluster c{
               compute avgsim(c) //average pairwise similarity between
             elements in c
               compute score: |c| × avgsim(c) //|c| is the number of
             elements in c}
store the highest-scoring cluster in a list L.
sort the clusters in L in descending order of their scores.
let C be a list of committees.
for each cluster c Є L{
               compute the centroid of c
               if c's similarity to the centroid of each committee
             previously added to C is below a threshold $\theta_1$, add c to
             C.}
      If L is empty, return C.
else{
       For each element e Є E{
                      If e's similarity to every committee in C is
                    below threshold $\theta_2$, add e to a list of residues R
                    }
       If R is empty, return C
       else discover_committees (S, R, $\theta_1$,$\theta_2$)}
}

Phase 3
let X be a list of clusters initially empty
let C be the list of committees from phase 3
while S is not empty {
let cЄ S be the most similar committee to e
if the similarity(e, c) < $\sigma$, exit the loop
if c is not similar to any cluster in C {
               assign e to c
               remove from e the centroid vector of c;}
remove c from S}
```

Fig. 1. Soft Clustering by Committee algorithm

5 Results and Discussion

As mentioned in Section 4, the experiments presented in this paper were conducted using the untagged TASA corpus. We selected around 50 paragraphs each from the categories: Science, Social Science and Language and Arts. The paragraphs were selected randomly from the corpus. They contained 1349 unique words of which 104 were polysemous words. The results of the different stages of the algorithm are summarized below.

Phase 1

This phase determined the similarity index between words. Words such as *chlorine* and *sodium* were found to have similarities as high as 0.802 with a context window of size two. We realized that the training data for most of the sample words was small. Only two paragraphs of about 150 words each contained the words *chlorine* and *sodium*, still the similarity index was high.

Phase II

Here, the committees present in the text are identified. In total 834 committees in all were discovered. Table 2 shows some the committees with formed with the highest scores.

Table 2. The committees with highest scores

Type of Text	Cluster Type	Cluster formed
Science	Common elements	{chlorine, sodium, calcium, iron, wood, steel}
	Sources of light	{sun, bulb, lamp, candle, star}
Social Science	Places to live	{country, continent, island, state, city}
	Water resources	{nile, river, pacific, lake, spring}
Language and Arts	Body movements	{dance, walk, run, slide, clap}
	Colours	{red, yellow, blue, green, black}

Phase III

This phase assigned the words to different clusters. Table 3 shows some of the words and their discovered senses/committees. Note that some of the words belong to more than one cluster, suggesting that they are used in more than one sense.

Table 3. Some of the words and their discovered senses/committees

Polysemous Word	Clusters
Capital	*Sense 1- money, donation, funding* *Sense 2- camp, township, slum*
Water	*Sense 1- liquid, blood, moisture* *Sense 2- ocean, pond, lagoon* *Sense 3-tide, surf, wave, swell*
Plank	*Sense 1- bookcase, dining table, paneling* *Sense 2- concrete, brick, marble, tile*

We assigned the contextual senses to these words manually, and compared our results to the senses assigned to these words by the algorithm. The data consisted of 104 polysemous words and 157 senses.

We report the rest of results using precision, recall and the F measure. For our evaluation we have used modified definitions of the above terms as given in [7].

Precision of a word *w* is defined as the ratio of the correct clusters to the total number of clusters it is assigned to. The *precision* (P) of the algorithm is the average precision of all the words considered by it. *Recall* of a word *w* is defined as the ratio of the correct clusters of the word and the total number of senses the word is used in the corpus. The *recall* (R) of the algorithm is the average recall of the words considered by it. The *F-measure* (F) combines precision and recall into a single quantity and is given by

$$F = \frac{2RP}{R+P}$$

Table 4. Comparison of Precision, Recall and F-measure obtained while using different window sizes for values $\theta_1 = 0.35$, $\theta_2 = 0.55$, $\sigma = 0.48$

Window Size→ Category ↓	One word			Two words			Three words		
	P	R	F	P	R	F	P	R	F
Science	0.612	0.312	0.405	0.841	0.642	0.627	0.723	0.502	0.596
Social Science	0.567	0.277	0.372	0.822	0.588	0.685	0.709	0.480	0.572
Language and Arts	0.245	0.156	0.212	0.390	0.195	0.360	0.278	0.185	0.232

As apparent from Table 4, the best results are obtained when using a window size of two words on either side of the focus word.

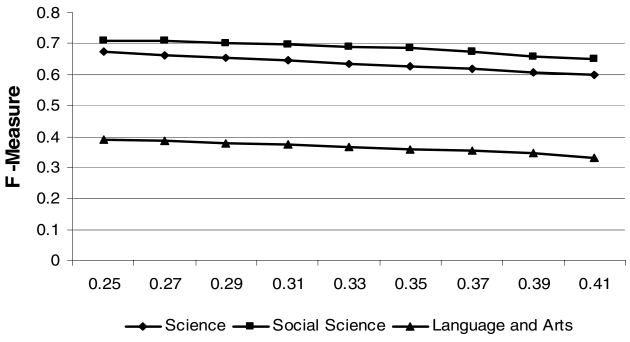


Fig. 2. F-Measure for different paragraphs with $\sigma = 0.48$ and varying values of θ_1

As the value of θ_1 increases, the clustering becomes more strict and only words with very high similarity index are clustered, causing the *F*-measures to decrease when θ_1 increases.

For all sense discoveries an element is assigned to a cluster if its similarity to the cluster exceeds a threshold σ. The value of σ does not affect the first sense returned by the algorithms for each word because each word is always assigned to its most similar cluster. We experimented with different values of σ and present the results in Fig. 3.

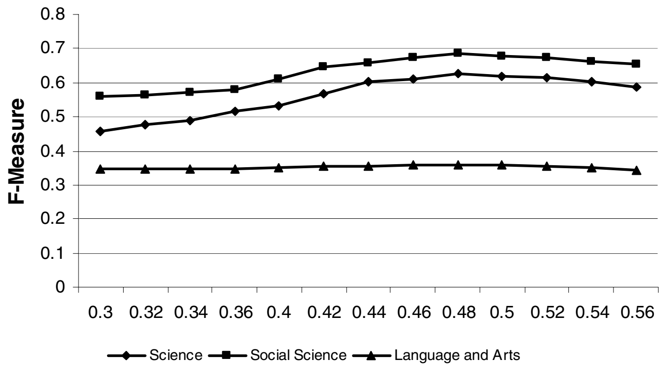


Fig. 3. F-Measure for different paragraphs with θ_1= 0.35 and varying values of σ

With a lower σ value words are assigned to more clusters causing a decrease in precision and hence in the F-measure. At higher values of σ the recall reduces as the algorithm misses a few senses of words and thus a decrease in F-measure is observed.

It can be observed from the results that the algorithm gives very good precision and recall values for paragraphs related to Science and Social Science, but performs poorly on paragraphs related to Language and Arts. This forms an interesting observation. A further study into the paragraphs reveals certain characteristic feature of the paragraphs related to different fields and the words used in them.

Ploysemous words, such as *'water'*, *'plank'* used in Science and Social Science paragraphs are used in fewer senses, like *'water'* is used in three senses, namely *element*, *water body*, and *motion of water* whereas *'plank'* is used in two senses, *things made of wood* and *material used in construction*. The different senses in these paragraphs can be clearly defined. However, a word like *'dance'*, which was very commonly observed in paragraphs on Language and Arts was used in more than five senses and some of the senses, such as '*a form of expression*' or '*motion of the body*'. These senses are more abstract, and are hard to define. This causes the word '*dance*' to be related to many words, such as *'smile'* , for the sense '*a form of expression*' and to *'run'* for the sense *'motion of the body'* and with a very small similarity index. This causes a poor clustering of words in Language and Arts paragraphs.

Moreover, the words appearing in Science paragraphs, such as *'chlorine'* , *'sodium', 'wood'*, when used in similar sense, occurred with a fixed set of co-occurents, therefore the similarity index was very high. This caused formation of very strong clusters. However, words such as *'run'*, *'walk'*, *'jump'* etc. when used in similar sense in Language and Arts paragraphs, were used with a many, different co-occurents. Therefore the similarity index was low, and weak clusters were formed.

Decreasing the values of the various cut-off scores (θ_1= 0.30, θ_2 =0.50, σ = 0.35) seems to improve the results for Language and Arts paragraphs, with precision of 0.683, recall of 0.305 and F-measure of 0.47. However, the results are still not comparable with those of Science and Social Science paragraphs. Also, the decreased values give poorer results for the Science and Social Science paragraphs, because certain dissimilar words are added to the same clusters.

A larger training data for Language and Arts paragraph containing similar instances of words and their senses should solve this problem of poor cluster formation.

6 Conclusions and Future Scope

In this work we have used Random Indexing based Word Space model, in conjunction with Clustering by Committee (CBC) algorithm to form an effective technique for word sense disambiguation. The Word Space model implementation of CBC is much simpler as compared to original CBC as it does not involve any resources, such as a parser or a machine readable dictionary. The approach works efficiently on corpora, such as TASA that contains simple English sentences.

The proposed approach works efficiently on paragraphs related to Science and Social Science. The best F-values that could achieved for these paragraphs are 0.627 and 0.685, respectively, by considering a context of 2 x 2 windows. If we compare these results with other reported WSD results a clear improvement can be noticed. For example, Graph ranking algorithm based WSD on an average report F-measure of 0.455 on SensEval corpora [15]. Reported performance of other clustering based algorithms [8], such as UNICON (F- Measure = 0.498), BUCKSHOT (F-Measure = 0.486), K-Means clustering (F-Measure = 0.460), are also much less than the approach proposed in this work.

However, for paragraphs related to Language and Arts the proposed approach at present fails to provide good results. We ascribe the cause to a wider range of uses of words that is typically found in literature etc. We intend to improve our algorithm to take care of these cases efficiently. This apart from experimenting with different values of cutoffs and including more instances in training data, will also include modifications in the Word Space model itself.

Presently, we have not focused on various computational complexity (e.g. time, space) issues. In future we intend to compare our scheme with other clustering based word sense disambiguation techniques in terms of time and space considerations. We will test the scheme on other available corpora, such as British News Corpus, which contain long, complex sentences, in order to measure the efficiency of the proposed scheme on a wider spectrum.

Acknowledgements

We wish to thank Praful Mangalath of University of Colorado at Boulder for providing us with TASA Corpus.

References

1. Ide, N., Veronis, J.: Word Disambiguation Ambiguation - State Of Art. Computational Linguistic (1998)
2. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. Technical Report #00-034. Department of Computer Science and Engineering, University of Minnesota (2000)

3. Cutting, D.R., et al.: Scatter/Gather: A cluster-based approach to browsing large document collections. In: Proceedings of SIGIR-1992, Copenhagen, Denmark (1992)
4. Pantel, P., Lin, D.: Discovering word senses from text. In: Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Edmonton, Canada (2002)
5. Karypis, G., Han, E.H., Kumar, V.: Chameleon: A hierarchical clustering algorithm using dynamic modeling. IEEE Computer: Special Issue on Data Analysis and Mining (1999)
6. Miller, G.: WordNet: An online lexical database. International Journal of Lexicography (1990)
7. Pantel, P.: Clustering by Committee. Ph.D. dissertation. Department of Computing Science. University of Alberta (2003)
8. Sahlgren, M.: The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensionalvector spaces.Ph.D. dissertation. Department of Linguistics. Stockholm University (2006)
9. Sahlgren, M.: An Introduction to Random Indexing. In: Proceedings of the Methods and Applications of Semantic Indexing. Workshop at the 7th International Conference on Terminology and Knowledge Engineering. TKE, Copenhagen, Denmark (2005)
10. Landauer, T.K., Foltz, P.W., Laham, D.: An Introduction to Latent Semantic Analysis. In: 45th Annual Computer Personnel Research Conference – ACM (2004)
11. Kanerva, P.: Sparse distributed memory. MIT Press, Cambridge (1968)
12. Kaski, S.: Dimensionality reduction by random mapping - Fast similarity computation for clustering. In: Proceedings of the International Joint Conference on Neural Networks, IJCNN 1998. IEEE Service Center (1998)
13. Porter, M.: An algorithm for suffix stripping. New models in probabilistic information retrieval. London (1980)
14. http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words
15. http://www.senseval.org/

Domain Information for Fine-Grained Person Name Categorization

Zornitsa Kozareva, Sonia Vazquez, and Andres Montoyo

Departamento de Lenguajes y Sistemas Informaticos
Universidad de Alicante
{zkozareva,svazquez,montoyo}@dlsi.ua.es

Abstract. Named Entity Recognition became the basis of many Natural Language Processing applications. However, the existing coarse-grained named entity recognizers are insufficient for complex applications such as Question Answering, Internet Search engines or Ontology population. In this paper, we propose a domain distribution approach according to which names which occur in the same domains belong to the same fine-grained category. For our study, we generate a relevant domain resource by mapping and ranking the words from the WordNet glosses to their WordNet-Domains. This approach allows us to capture the semantic information of the context around the named entity and thus to discover the corresponding fine-grained name category. The presented approach is evaluated with six different person names and it reaches 73% f-score. The obtained results are encouraging and perform significantly better than a majority baseline.

1 Introduction

The Named Entity Recognition (NER) task was first introduced in the Message Understanding Conference (MUC) as it was discovered that most of the elements needed for the template filling processes in Information Extraction systems are related to names of people, organizations, locations, monetary, date, time and percentage expressions.

There are two main paradigms for NER. In the first one, NEs are recognized on the basis of a set of rules and gazetteer lists [6], [1]. The coverage of these systems is very high, however they depend on the knowledge of their human creator, the number of hand-crafted rules and the kind of entries in the gazetteer lists. In addition, NER rule-based systems are domain and language dependent, therefore they need lots of time in order to be developed and to be adapted.

The other paradigm is machine learning (ML) based NER. Given a set of feature vectors characterizing a named entity, a machine learning algorithm learns these properties and then assigns automatically NE categories to unseen entities. These systems are easily adaptable to different domains, they can function with language-independent characteristics [16], [17], however, their main drawback is related to the number of hand-labeled examples from which the ML system learns. For languages with limited resources and funding, such annotated corpora are not available. This directed researchers towards the development of semi-supervised NE recognizers which turn automatically unlabeled data into labeled [4].

So far the presented and the developed NE approaches focus only on the resolution of the seven main NE categories as defined in the MUC challenge. However, to be able to answer the question "Who is the president of USA in 1994?", a Question Answering (QA) system needs to identify that a given person name belongs to the semantic category PRESIDENT. For web queries such as " MTV winner", the search engine should return relevant documents containing person names from the SINGER category. Unfortunately, current NE recognizers do not have the potential to provide such type of classification. This motivated us to conduct a study for fine-grained NE categorization.

In this paper, we propose a new approach for fine-grained name categorization which is based on the information of relevant domains. Our hypothesis is that names occurring in the same domain are highly probable to belong to the same semantic category e.g. named entity class. The experiments are carried out with person names, because according to [5], the person name classification is more challenging and needs deeper semantic knowledge derived from the surrounding text. For each sentence having a person name we have to classify, our approach calculates the domain probability distribution of the words around the person name. Then, the most representative domain is selected and it is assigned as the fine-grained category of the person name. Our approach is capable to handle the time-consistency property according to which a person name changes its semantic category across time. For instance, the movie star Arnold Schwarzenegger became the Governor of California in 2003. This event is captured with our approach because the word distribution changes from the domain entertainment to the domain politics.

The rest of the paper is organized as follows. Section 2 discusses related work to fine-grained NER and name discrimination. In Section 3 we present our approach. Section 4 and 5 show the experimental setup and the evaluation we have performed. Finally we conclude in Section 6.

2 Related Work

While much research focuses on the coarse-grained NE categorization, there are not many approaches for the fine-grained NE categorization. The main reason is related to the need of certain amount of hand-annotated examples per fine-grained category. A couple of important issues arise such as: how do we understand how many examples do we need per category, how do we guarantee what a representative example for given class is, a gold standard data which is not available so far. In addition, we need an explicit definition for the granularity of the NE categories. For instance is the classification person-president sufficient or are we interested in person-president-of-country. Not on a last place, we have to consider the sources of information from which we can gather the instances that represent given class. For example, if we look for presidents we can gather many different contexts about them from the news papers or the web, while for the classification of bacterias we have to use specialized corpus.

Given the above mentioned circumstances, [18] proposed a NE hierarchy that consists of 150 different NE types. The hierarchy is designed from corpus information and facts gathered from existing QA and IE systems. Afterwords, [5] developed fine-grained NE classification approach for the classification into eight specialized classes. They used the context surrounding the named entity in a combination with the semantic information derived from the topic signatures and WordNet. Other approaches like [9] and [19] used syntactic features derived from word co-occurrences, while [3] performed a comparative study between the effect of contextual and syntactic features.

Some fine-grained NE approaches focused on the automatic acquisition of instances for a given class and later they were used to populate an ontology. For instance [14] proposed a lightly-supervised method for the automatic acquisition of NEs from arbitrary categories. The approach is based on lexico–syntactic patterns which are applied to unstructured text of web documents. With a similar pattern-based approach, [11] built a proper name ontology from news wire texts and later on populated some of the WordNet nodes.

To surmount the problem of labeled data needed for the evaluation of the developed categorization approach, [2] used Wikipedia to assign different semantic categories to people who share one and the same name. [15] proposed a new strategy called "pair conflation" which allows the automatic evaluation of ambiguous names or fine-grained categories by conflating two non-ambiguous names under the same label. In order to evaluate the performance of the city category, they gather separately context for the cities New York and Boston, then they conflate the examples by hiding the non-ambiguous names of New York and Boston with the NY-B label. The system has to separate automatically the examples which refer to New York from those that refer to Boston. To do that, they developed a second order co-occurrence method which was evaluated with the location, organization and person categories.

The presented approaches suffer from the lack of global contextual representation of the sentence in which the NEs appear. For this reason, we propose the domain distribution approach which captures the contextual and the semantic meaning of the text and associates it to the relevant domains of the words surrounding the person name. Later the domains are considered as the fine-grained category of the named entity.

3 Fine-Grained Person Name Categorization with Relevant Domains

An inherent property of natural language is that the words appearing in a discourse are semantically related. We take advantage of this property, and we use the context of the words in order to obtain the semantic information they are referring to. This semantic information is used to discover the underlying meaning of the whole sentence. Our fine-grained person name categorization approach is based on the hypothesis that the words appearing in a similar context are semantically related. To obtain the semantic evidence of the words, we use the relevant domains resource [13] in which the words are semantically associated with domain information.

3.1 Extraction of Relevant Domains

The first step in our fine-grained name categorization is related to the generation of the relevant domain (RD) resource. The RDs are obtained through the mapping of the WordNetDomains (WND) [10] and the words of WordNet. Each WordNet word is associated to various semantic categories. An important characteristic of the WND is that they relate semantically words that belong to different syntactic categories. For instance, the noun *patient* and the verb *operate* are mapped to the domain `MEDICINE`.

To obtain the word-domain mapping, we take all words that appear in the glosses of WordNet, and we assign to these words the different domain labels that represent the synset of the gloss. Once all words from the WordNet glosses are related to the domains, we create a word-domain list. The "word" can be noun, verb, adverb or adjective, and the "domain" corresponds to the label of the synset. However, our purpose is not only to generate a word-domain list, but alto to rank it according to some relevancy score. In order to do that, we use the Mutual Information (MI) (1)

$$MI(w,D) = \log_2 \frac{Pr(w|D)}{Pr(w)} \tag{1}$$

and Association Ratio (AR) (2)

$$AR(w,D) = Pr(w|D) \log_2 \frac{Pr(w|D)}{Pr(w)} \tag{2}$$

formulae, where w is the word and D is the domain. The two measures are selected, because MI arranges the word-domain pairs according to the most representative domain that corresponds to a word. While by representativeness we mean a word that tends to appear very often in the context of a given domain. However, MI cannot establish the importance of the word-domain relation, therefore in a continuation we apply AR. This measure provides a significance score information of the most relevant and common domain of a word. AR is able to capture the words that appear many times in several domains and associates them as non common words. Finally, the word-domain pairs are arranged by their AR values and thus the relevant domain recourse is obtained.

3.2 Adaptation of Relevant Domains to Fine-Grained Person Name Categorization

In the scenario of our fine-grained approach, the usage of the relevant domain resource is very pertinent and appropriate, because we can establish the semantic relations among the words around the NE. Thus we can determine the global domain of the context in which the NE appears. For each sentence for which we have to find the fine-grained NE category, we apply the following algorithm:

Given:

- w word
- S sentence with named entity
- D domain
- AR association ratio value

Loop for $\forall w_i \in S$:

1. generate all possible $w : D : AR$ triplets
2. arrange $w : D : AR$ by D
3. rank all D by AR score and sort D in a descending order
4. determine for $\forall w_i$ the most representative D
5. assign D as a category to the named entity

3.3 A Walk-Through Example

In a continuation, we show a walk-through example, where the algorithmic structure shown in subsection 3.2 is applied to determine the category of the person name Madonna.

The text snippets we work with have a length of around 50 words around the named entity we want to classify. The text snippets look like the following one: "*nearly three years, but Radio Venceremos continues to broadcast though not from the hills, and not in the name of a guerrilla movement. Instead, the station has settled into comfortable studios on a quiet residential street here and has gone commercial, replacing its battle reports with the music of* $<$ *name_of_interest* $>$ *Madonna* $<$ */name_of_interest* $>$*, Phil Collins, Air Supply, the Eagles, and Boyz II Men.*"
The purpose of our fine-grained system is to determine the semantic category for the person name Madonna using the context in which the name appears. In our example, the underlined surrounding words form the context for the named entity classification. It can be seen that we consider as context baring words the nouns, the verbs, the adverbs and the adjectives. For each underlined word, we obtain the word:domain:AR triplets from the relevant domain resource:

year= {year ethnology 0.034834, year astronomy 0.011396, year archaeology 0.008253, year exchange 0.004034, year economy 0.003140, year dance 0.002007, year banking 0.001252, year color 0.001091, year anthropology 0.000700, year fashion 0.000648}

radio= {radio telecommunication 0.179210, radio electricity 0.076610, radio electronics 0.055499, radio telephony 0.051643, radio engineering 0.025319, radio acoustics 0.014942, radio electrotechnics 0.009850, radio physics 0.006277, radio furniture 0.005360}

...

music = {music music 0.313137, music dance 0.074631, music free_time 0.067948, music radio 0.037999, music acoustics 0.036253, music art 0.014448, music telecommunication 0.004600, music theater 0.002844, music school 0.001713, music pedagogy 0.001426}

Once we have obtained the triplets for all words in the sentence, we order the triplets by their domains, and we finally rank them according to their AR value.

For the Madonna text snippet, the most common WordNetDomains are shown in Table 1.

Table 1. List of the nine most relevant domains for the Madonna snippet

Domain	**AR**
music	0.770537
acoustic	0.740445
dance	0.717400
radio	0.716576
theatre	0.714463
art	0.705165
free_time	0.700763
color	0.658827
fashion	0.627876

According to the obtained results in Table 1, the domain with the highest AR value is *music*. This domain determines that the context in which the person name Madonna appears is related to music e.g. person-music which we map into the category singer because singers are associated with music.

4 Experimental Setup

For the evaluation of our fine-grained NE categorization approach, we followed the name-conflation strategy proposed in [15] . In their approach the NE examples are extracted from large news corpora where different unambiguous names are conflated together in order to create the ambiguous pairs. Although the selected names are individually unambiguous, they are still related in some way according to the hypothesis of [12]. For each sentence, [15], do not reveal or utilize the underlying meaning of the names until evaluation.

4.1 Data Description and Name Conflation

Since there is no fine-grained NE corpus, we decided to compile our own corpus. We used the 900 million word New York Times news portion, from which we have extracted different NEs and their name variants. One of the challenges of the person names is related to the fact that the same individual (e.g. Bill Clinton) is often represented differently in the same text, therefore named entity variants such as "president Bill Clinton", "president Clinton", "the president of U.S.A. Clinton" are also considered in the example extraction process.

The person names we have selected for our experimental study are the presidents Bill Clinton, George Bush and Fidel Castro, and the singers Madonna, Michael Jackson and Gloria Estefan. These names are identified with regular expressions and for each name a context of fifty words[1] around the named entity is gathered. With the termination of the name extraction process, we obtain our fine-grained named entity corpus.

As the distribution of the president and the singer names is different in the New York Times corpus, this can bring imbalance in the experimental evaluation. For this reason we decided to balance the data by selecting randomly 200 examples per NE or 600 examples per NE category. From this data we created two data sets. One is used during the development stage and the other is used during the test stage of our approach.

The sentences with the president names Bill Clinton, George Bush and Fidel Castro are mingled together and the president names are obfuscated with the `PRESIDENT` label. The same procedure is repeated for the singer names which are obfuscated with the `SINGER` label. The purpose of our domain distribution approach is first to determine for each hidden name entitiy its semantic category e.g. president or singer, and then to discover the exact named entity which is hidden behind the fine-grained category e.g. one of the president or singer names.

4.2 Evaluation

We have carried out two experimental evaluations. In the first one, each word in the sentence is associated with several domains. To determine the global domain of the sentence, we sum up the domain probabilities of the words and rank the most representative domain. The probability distribution of the domains determines the person category.

In the second experiment, for each word we rank the domains according to their association ratio values. The word whose domain has the highest weight determines the person name category. For each experiment, we measure precision, recall, f-score and accuracy of the correctly classified named entities. Our 50% baseline is obtained from a system that randomly assigns to one half of the example the `SINGER` category and to the other half the `PRESIDENT` category.

Although we have evaluated the performance of our approach only with the `PRESIDENT` and `SINGER` categories, the conducted experiments are sufficient to demonstrate the performance of our approach, its drawbacks and the encountered obstacles. Moreover, we have conducted a study where three human annotators select randomly 100 examples from the development corpus and assign independently the domains that correspond to the NE examples. This study is carried out in order to verify the purity level of the compiled data, to evaluate

[1] We study a context of twenty-five, fifty and hundred words. According to the experimental results, the context of fifty words is the most representative for the person name categorization.

Table 2. Results for the relevant domain ranking per person name category

Set	Category	Prec.	Rrec.	F.	Acc.
Dev	PRESIDENT	63.31	96.66	**76.51**	**98.33**
Dev	SINGER	92.95	44.00	59.72	72.00
Test	PRESIDENT	65.23	96.33	**77.79**	**98.16**
Test	SINGER	92.99	48.66	63.89	74.33

Table 3. Results for the relevant domain probability distribution per person name category

Set	Category	Prec.	Rec.	F.	Acc.
Dev	PRESIDENT	71.46	82.66	**76.66**	91.33
Dev	SINGER	79.44	67.00	**72.69**	**83.50**
Test	PRESIDENT	71.88	82.66	76.89	91.33
Test	SINGER	79.60	67.66	**73.15**	**83.88**

the results of the relevant domains and also to explain some of the occurred classification errors.

5 Results and Discussion

In this section we show the results for the development and test data sets from the first and the second experiment we have carried out. The results for the correct resolution of the PRESIDENT and SINGER categories are shown in Table 2 and 3, while the coverage of the individual person names is shown in Table 4 and 5. All results are compared to and outperform the 50% baseline.

During the comparative study of the probability distribution and the domain ranking experiment, we observed that the probability approach is more reliable with the SINGER category, while the domain ranking functions better with the president names. This is related to the context in which the NEs appear.

Table 4. Results for the relevant domain ranking per individual

PRESIDENT			
	Bill Clinton	George Bush	Fidel Castro
Dev	**98.30**	**99.48**	**97.24**
Test	**98.47**	**97.95**	**97.95**
SINGER			
	Madonna	Gloria Estefan	Michael Jackson
Dev	50.00	**88.32**	45.90
Test	43.75	**83.72**	63.01

Table 5. Results for the relevant domain probability distribution per individual

PRESIDENT			
	Bill Clinton	George Bush	Fidel Castro
Dev	91.56	91.71	88.55
Test	89.50	93.04	88.88
SINGER			
	Madonna	Gloria Estefan	Michael Jackson
Dev	**79.24**	79.12	**70.34**
Test	**69.2**8	82.00	**80.95**

The president names are mostly surrounded by words whose density is the political domain. Therefore, the ranking of the most representative word domain determines easily the NE category. However, the examples in which the singer names appear have verbs, nouns, adjectives and adverbs related to domains other than the singer. This causes dispersion and scatterness, hence the probability distribution is more efficient with the examples in this category.

The semantic categorization of the person names obtains very good coverage for the PRESIDENT category. The highest result of 98% is obtained with the examples of president Bill Clinton. In comparison, the SINGER category is more problematic. For instance, Madonna is identified as singer in 69% of the cases with the probability distribution experiment and in 49% of the cases with the domain ranking approximation. This low coverage is related to the high ambiguity of the name. The same categorization problem is observed with Michael Jackson, who appeared to be ambiguous as we saw that in many examples this name appears with the domain war and the name corresponds to a general of the American troops and not to the pop singer. These two singer names do not indicate that the president names are more easier to be recognized. This experiment shows that the ambiguity problem poses a significant difficulty during the fine-grained person name categorization. Our observation is proven by the resolution of Gloria Estefan which is a low ambiguous name. For 83% of the examples, this name is correctly identified as a singer.

In addition, we provide the results of the annotation of the domains of the experimental data. The Kappa agreement score measured for the 100 examples on which three individual human annotators assigned the relevant domains is 93%. According to the human annotators, 90% of the president examples are related to the president and political domains. However, a significant variability is observed across the other category, where only 80% of the examples are related to singer and music. For instance, the singer Madonna occurs in 71% of the examples in the context of a singer, 9% of the examples as a movie star that shoot the movie Evita, 17% in the context of churches and 3% in the context of painting. As can be seen, the purity level of the compiled experimental corpus is not 100% for the president and singer domain. This analysis also explains why the SINGER category performs a bit lower compared to the PRESIDENT one. Our approach is very promising as it determines correctly the fine-grained person category with 73% from a 90-80% data purity level. The method is reliable for a fine-grained NE categorization and can be easily adapted and expanded to large scale NE categories or languages as we show in [8] and [7].

6 Conclusions and Future Work

In this paper, we presented a new approach for person name categorization which is based on domain distribution. Experimental results are very promising reaching 73% coverage for the `PRESIDENT` and `SINGER` categories we have selected, and from 69% to 98% for the individual person names. Compared to other fine-grained NE approaches that rely on lexical or syntactic information, we can claim that our approach is also robust and reliable because it considers the semantic information of the context in which the NEs appear. In addition, the relevant domains maintain the property of time-consistency according to which people change their semantic categories across time. Therefore, our approach identified how Madonna's name evolves in a given time period as a singer and a movie star.

An advantage of the proposed approach consists in the possibility to be expanded to other NE categories as we show in [8] and [7]. This makes our methodology applicable to large scale, because we only have to calculate the domain distribution of the context surrounding the NE candidate. In the future, we want to use our approach as a basis for the creation of fine-grained NER training data, whose representativeness can be guaranteed with active learning.

The major drawbacks of the relevant domain approach are related to the generation procedure of the experimental data and to the ambiguity level of the selected conflated names. In this approximation, we used the relevant domains found in the context words surrounding the NEs. To improve our fine-grained approach, we will apply term weighting strategies by giving more weight to the words that determine the global context of the snippet rather than considering all possible noun, verb, adverb and adjective candidates. Additionally, we have to handle the words which appear with unknown domains or the general domain FACTOTUM, because they hamper the correct fine-grained categorization. In order to create a more reliable relevant domain recourse, we plan to use extended WordNet where the words in the gloss are already disambiguated. This will augment the precision of the generated relevant domains resource.

In the future we want to apply the domain distribution approach for the person name disambiguation as well as to evaluate the presented approach for product and organization classes.

Acknowledgements

This research has been partially funded by the European Union under the project QALLME number FP6 IST-033860 and by the Spanish Ministry of Science and Technology under the project TEXT-MESS number TIN2006-15265-C06-01.

References

1. Black, W., Rinaldi, F., Mowatt, D.: Facile: Description of the ne system used for muc. In: Proceedings of the Message Understanding Conference (1998)
2. Bunescu, R., Pasca, M.: Using encyclopedic knowledge for named entity disambiguation. In: Proceeding of ACL, pp. 9–16 (2006)
3. Cimiano, P., Volker, J.: Towards large-scale, open-domain and ontology-based named entity classification. In: Proceeding of RANLP 2005, pp. 166–172 (2005)
4. Collins, M., Singer, Y.: Unsupervised models for named entity classification. In: Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (1999)
5. Fleischman, M., Hovy, E.: Fine grained classification of named entities. In: Proceedings of the 19th international conference on Computational linguistics, pp. 1–7, Association for Computational Linguistics, Morristown (2002)
6. Gaizauskas, R., et al.: University of sheffield: Description of the lasie system as used for muc. In: Proceedings of the Sixth Message Understanding Conference (MUC-6), Morgan Kaufmann, San Francisco (1995)
7. Kozareva, Z., Vazquez, S., Montoyo, A.: Discovering the underlying meanings and categories of a name through domain and semantic information. In: Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP) (2007)
8. Kozareva, Z., Vazquez, S., Montoyo, A.: A language independent approach for name categorization and discrimination. In: Proceedings of the ACL 2007 Workshop on Balto-Slavonic Natural Language Processing (2007)
9. Lin, D.: Automatic retrieval and clustering of similar words. In: Proceeding of COLING-ACL (1998)
10. Magnini, B., Cavaglia, G.: Integrating subject field codes into wordnet. In: Proceedings of LREC, pp. 1413–1418 (2000)
11. Mann, G.S.: Fine-grained proper noun ontology for question answering. In: Proceeding of COLING-2002 on SEMANET, pp. 1–7 (2002)
12. Nakov, P., Hearst, M.: Category-based pseudowords. In: NAACL 2003: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 67–69 (2003)
13. Navarro, B., et al.: Improving interaction with the user in cross-language question answering through relevant domains and syntactic semantic patterns. In: Proceedings of CLEF-2005, pp. 334–342 (2005)
14. Pasca, M.: Acquisition of categorized named entities for web search. In: Proceedings of CIKM, pp. 137–145 (2004)
15. Pedersen, T., et al.: An unsupervised language independent method of name discrimination using second order co-occurrence features. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 208–222. Springer, Heidelberg (2006)
16. Sang, E.F.T.K.: Introduction to the conll-2002 shared task: Language-independent named entity recognition. In: Proceedings of CoNLL-2002, pp. 155–158. Taipei, Taiwan (2002)
17. Sang, E.F.T.K., De Meulder, F.: Introduction to the conll-2003 shared task: Language-independent named entity recognition. In: Proceedings of HLT-NAACL, pp. 142–147 (2003)
18. Sekine, S., Sudo, K., Nobata, C.: Extended named entity hierarchy. In: Proceeding of LREC (2002)
19. Tanev, H., Magnini, B.: Weakly supervised approaches for ontology population. In: Proceedings of ACL, pp. 17–24 (2006)

Language Independent First and Last Name Identification in Person Names

Octavian Popescu and Bernardo Magnini

FBK-Trento, Italy
{popescu,magnini}@fbk.eu

Abstract. In this paper we address the problem of first name and last name identification in a news collection. The approach presented is based on corpus investigation and is language independent. At the core of the system there is a name classifier based on the values of different parameters. In its most general form, the name category identification is not an easy task. The hardest problems are raised by ambiguous tokens – those that can be either a first or a last name and/or by tokens with just one occurrence. However, the system is able to predict the name category with high accuracy. The experiments have been run on an Italian newspaper and the evaluation has been carried on I-CAB.

1 Introduction

Knowing whether a token composing the name of a person refers either to her/his first or last name is an important task in several respects. It is probably one of the first things a reader would like to know about a person, especially if she/he has no native intuitions. The name category (first vs. last) is also important for further processing of textual information. It plays an important role in enhancing the overall accuracy of a cross document coreference system (Popescu&Magnini, 2007). Many name mentions consist of only one token, but they definitely stand for two-token names; knowing the category of the token that is missing may give important clues about the person that carries that name.

The task consists in determining for each name occurrence in a large corpus the name category of each individual token composing it. For example, "*George W. Bush*" should be analyzed like "*George*<first name> *W.*<first name> *Bush*<last name>". In its most general form the name category identification is not an easy task. The hardest problems are raised by ambiguous tokens – those that can be either first or last names and/or by tokens with just one occurrence.

In this paper we address the problem of first, last name identification in a news collection. While relying on gazetteers or name dictionaries seems to be an easy way out, we show that this is not enough. The approach we are going to present is based on corpus investigation and is language independent. At the core of our system there is a name classifier which, for each token within a name mention, makes decisions combining different types of information. In particular, we rely on the token's distribution computed on the corpus/web, the token's lexical characteristics and the name usage in the corpus. All these parameters may have values which are particular to the language itself, but, nevertheless, determinable from the corpus.

The problem of name category has been mentioned in papers dealing with ontology population and cross-coreference systems. In Mann (2003) a method of building an English proper noun ontology is discussed using repositories like WordNet. In Krstev et al. (2005) the multilingual generalization is considered. Recently, Driscoll & Yarowky (2007) have undertaken the problem of standardized personal name variants. However, to our knowledge, the identification of name categories, which may be a very useful piece of information for all these tasks, has not been undertaken so far.

For the experiments reported here we have used a news corpus coming from an Italian local newspaper containing approximately 550 000 different names (about 6 millions mentions in total). We have taken into account nothing else but the names and their occurrences, abbreviations included. For Italian, it seems that there is a high prior probability that a first name is also a family name. However, we will show that for the great majority of cases, the system we have built is able to predict the name category with a high accuracy.

We have relied on the output of a Named Entities Recogniser (NER), based on a SVM, which obtains state of the art results. The evaluation was carried on the test set I-CAB (Magnini et al., 2006), a four day news documents coming from the same corpus.

We present the parameters of the models in Section 2 and the general architecture of the system in Section 3. Section 4 is devoted to the description of three approaches that can fulfil the central role of name classifier. In Section 5 we present the data, the experiments we have conducted and their evaluation. The paper ends with Section 6 – Conclusion and Further Research.

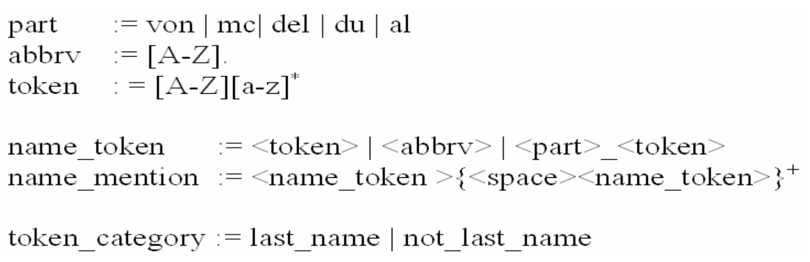
2 The Model's Parameters

At first sight, name mentions form a very heterogeneous class. Firstly, apparently there are no explicit differences between first and last names. Consider the Italian name "*Padoa*". It is hard to predict, base on its form, its name category. Secondly, we may find names containing up to 15 tokens, the order of tokens is variable from mention to mention, and, also, which tokens are chosen to mention the respective person(s) carrying that name may vary from instance to instance. However, a native speaker finds it easy to differentiate first vs. last names in almost all possible cases. Also, usually, she has strong intuitions on both how frequent a name is (carried by different persons) and on whether a particular usage corresponds to an established norm (compare "*George W. Bush*" to "*Bush George W.*", "*W. Bush George*"). Yet, if the names come from a different language, there are no such intuitions. From our point of view, all the names in the corpus are like foreign names.

The underlying assumption in this paper is that a token present in the name of a person is either a first or a last name, that is, we work with only two categories. Generally, these two categories are widely recognized[1]. Therefore, if in a particular case

[1] In some languages, what is called "middle name" in English is either not recognized as an independent category, or this category may look a lot different (see Russian, for example).

the category of a token is assumed not to be one of the two, then it is implied that it is the other one. In the subsequent paragraphs, for brevity sake, we refer to last or ‘not last’ name. In Figure 1 we give the BNF description of these terms.



```
part         := von | mc| del | du | al
abbrv        := [A-Z].
token        : = [A-Z][a-z]*

name_token       := <token> | <abbrv> | <part>_<token>
name_mention  := <name_token >{<space><name_token>}+

token_category := last_name | not_last_name
```

Fig. 1. BNF description of the task terminology

We introduce five parameters that we consider useful in determining the name category. Three out of the five parameters express probabilities for the individual tokens, while the other two express probabilities concerning names as a whole.

The first parameter, P_1, estimates the prior probability of a token to function as a last name. It is the probability of a token to be a last name regardless of the categories of the other tokens it combines with.

The second parameter, P_2, measures the collocational affinity of a token. We distinguish two distinct parameters, P_{2L} and P_{2R}, which are the probabilities for the respective token to collocate on the left and on the right of a last name respectively (the probability to collocate with a first name is the difference to 1).

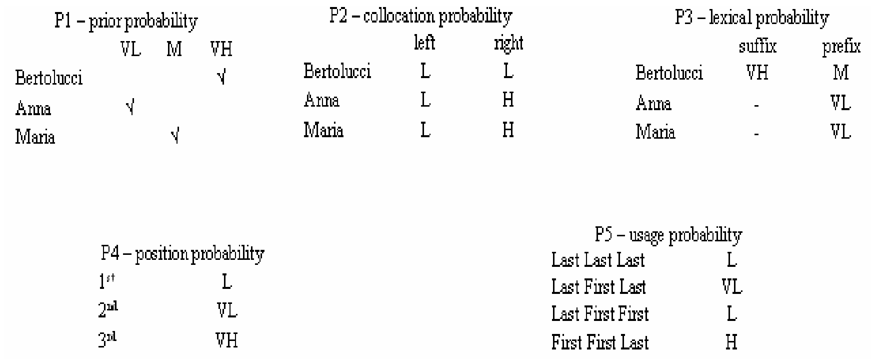
The third parameter, P_3, computes a lexical probability. Very often the lexical characteristics of a token are a powerful indicator of its category. We compute a list of suffixes/prefixes which, with a certain probability, are specific to last names. Therefore, this parameter is made up of two distinct parameters, let’s call them P_{3suf} and P_{3pref}.

The fourth parameter, P_4, represents the probability of a position within a name to be the position of a last name. This parameter is related to the names usage in language. In many languages, either the first or the last position in a name is the last name preferred position.

The fifth parameter, P_5, is the probability that a name has a certain composition. In many languages the most frequent form is “First name Last name”. However, the frequencies of a name may be highly variable from language to language. For example “First name Last name Last name” could be quite common in Spanish, but in Romanian it is almost inexistent.

Let’s take an example: consider an Italian name “*Bertolucci Anna Maria*”. Probably no native speaker doubts that “*Bertolucci*” is the last name, and “*Anna*”, “*Maria*” are the first names for this name. However, this is not an obvious conclusion. In order to reach the same conclusion, a system must be able to weigh different, sometimes contradictory, pieces of information. Firstly, “*Anna*” and “*Maria*” are ambiguous, they can be both first and last names. Secondly, in Italian, the norm is “{First name}$^+$Last name". Relying on "the most frequent assignment" leads to an incorrect name category assignment.

However, the correct category assignment can be obtained if we know that (1) the prior probability of last_name for "*Anna*" is very low ('VL'), for "*Maria*" is medium('M'), that (2) "*Anna*" could be followed by a not_last_name easily, that (3) "-*ucci*" in "*Bertolucci*"is a suffix which marks very confidently only last names, therefore the last_name probability is very high ('VH'), and that (4), in spite of the fact that the last name occupies the last position in a name mention with predilection, (5) the form "last name first name last name" is highly improbable in Italian usage. In Figure 2 this information is presented schematically.



P1 – prior probability

	VL	M	VH
Bertolucci			√
Anna	√		
Maria		√	

P2 – collocation probability

	left	right
Bertolucci	L	L
Anna	L	H
Maria	L	H

P3 – lexical probability

	suffix	prefix
Bertolucci	VH	M
Anna	-	VL
Maria	-	VL

P4 – position probability

1st	L
2nd	VL
3rd	VH

P5 – usage probability

Last Last Last	L
Last First Last	VL
Last First First	L
First First Last	H

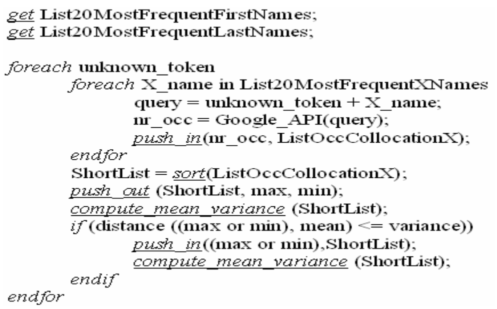
Fig. 2. Token/Name parameters and their normalized values

2.1 Compute Parameters Values

The easiest and also the safest way to compute the parameters values is to have access to manually built name resources (dictionaries, census lists etc.). Unfortunately, such resources are hard to find and their coverage is low. The technique we present further is a bootstrapping approach. We have started with two name dictionaries, one for first and one for last names respectively. Their coverage is only 48% of the corpus data. Using these two dictionaries and corpus statistics, the rest of 62% of cases are resolved, indicating the certainty of each prediction. For our approach the dimension of name dictionaries is not important, but the correctness of at least one of them is crucial. Our first_name dictionary has only about 3000 entries, a reasonable number for doing manual checking.

For each one of the above parameters we describe further a procedure that computes the respective values from the web and/or corpus.

The prior probability of a token to be a last name, P_1, can be determined using the web. The working hypothesis is that we can estimate the frequency of the category "last_name" of a token by counting how many Web occurrences of the string "token first name" there are, where "first name" comes from a closed list. From a public web page we have extracted the top 20 most frequent last names and first names (in total 40 names). Google allows the Web users to make automatic queries and to obtain the number of occurrences. We have used this option to compute the frequency class for about 15 000 unknown tokens. For each token, we launch 40 queries formed by the token itself with one of the most frequent first/last names and we record the number of occurrences. We obtain two groups of 20 numbers each, representing the number of times the token is collocated with a first name, and, respectively, with a last name. We compute the average for each group leaving out the extremes unless they are within a standard deviation of the mean. We take this precaution in order to avoid the fact that a bigram is frequent on the web just becomes it is the name of a famous person. For example, the distribution of the token "*Arzeglio*", which is a rare Italian first name, is skewed by the fact that "*Arzeglio Ciampi*" is a famous person. In the end we obtain an estimate of the frequency of a token functioning as a first and last name respectively. In Figure 3 we reproduce the relevant pseudo code of this procedure.



```
get List20MostFrequentFirstNames;
get List20MostFrequentLastNames;

foreach unknown_token
        foreach X_name in List20MostFrequentXNames
                query = unknown_token + X_name;
                nr_occ = Google_API(query);
                push_in(nr_occ, ListOccCollocationX);
        endfor
        ShortList = sort(ListOccCollocationX);
        push_out (ShortList, max, min);
        compute_mean_variance (ShortList);
        if (distance ((max or min), mean) <= variance))
                push_in((max or min),ShortList);
                compute_mean_variance (ShortList);
        endif
endfor
```

Fig. 3. Compute token prior probability using Web estimates

We can also use the corpus itself to determine the name category for unknown tokens. Let us assume that we know the name usage norm in the corpus and that we also know the norm frequency, let's call it *pnorm*. The Italian norm is "First name Last name" and it is known from a previous experiment (Magnini et al. 2006) that in the Italian bigram names, the "First Name Last Name" form is observed in roughly 88% of cases. We can evaluate the token's prior probability of being a last name by assuming a normal distribution for the different names it collocates with in bigrams. The probability of being a last name can be computed using a Poisson distribution having the mean $\mu = pnorm$. For example, if a token is seen in three different name mentions on the whole corpus and it collocates on the right of three different first names, then the probability of it belonging to last_name is $1 - (12/100)^3 \approx 1$.

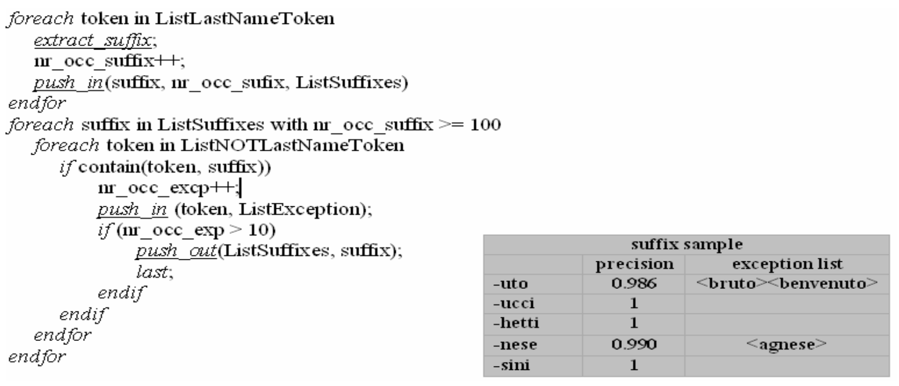
For every token we compute five distributional values out of the corpus: (D_0) the number of occurrences in different names each having more than two tokens, (D_1) the number of occurrences on the rightmost position, (D_2) the number of occurrences on the leftmost position, (D_3) the number of occurrences on the right of a known not_last_name, (D_4) the number of occurrences on the left of a known last_name. We estimate the last_name category for an unknown token by computing a joint conditional probability. The method is described in Section 4.1 "min max Estimates".

The P_2 parameter, the collocation affinity, can be calculated as the frequency ratio of the bigrams where the category of the co-bigram partner is known. We count the number of occurrences of type "token last_name" (P_{2L}) and "last_name token" (P_{2R}) and the total number of bigrams containing the token.

The P_3 parameter, the lexical probability, measures the similarity between two tokens in terms of their lexical composition. In many languages, some of the last names can be grouped according to their prefix/suffix. As we have already seen in the example "*Bertolucci Anna Maria*", "*-ucci*" is a prefix which marks, almost for sure, only the last names. It is not necessary that every possible discriminative lexical trait is 100% sure. It is probably more useful if we can compute the list of exceptions for each trait. We have chosen a prior threshold of 90%, which means that a minimum of 100 occurrences and a maximum of 10 possible exceptions must be observed. For example, we consider as a reliable indicator of last_name category a suffix that is present in at least 100 sure last names and there are less than 10 first names that carry it, too.

The fact that P_3 is a relevant parameter in evaluating the name category can be seen from the fact that out of 270 suffixes that score over the 90% threshold, the great majority also scores over 98%. There are 154 suffixes that score 100% and 227 which score over 98%.

In Figure 4 we reproduce the relevant code and a sample of the results obtained.



```
foreach token in ListLastNameToken
   extract_suffix;
   nr_occ_suffix++;
   push_in(suffix, nr_occ_sufix, ListSuffixes)
endfor
foreach suffix in ListSuffixes with nr_occ_suffix >= 100
   foreach token in ListNOTLastNameToken
      if contain(token, suffix))
         nr_occ_excp++;
         push_in (token, ListException);
         if (nr_occ_exp > 10)
            push_out(ListSuffixes, suffix);
            last;
         endif
      endif
   endfor
endfor
```

suffix sample		
	precision	exception list
-uto	0.986	<bruto><benvenuto>
-ucci	1	
-hetti	1	
-nese	0.990	<agnese>
-sini	1	

Fig. 4. Extract lexical name category distinct traits

The P_4 and P_5 parameters are computed in the same way the P_2 is computed. We count the number of each possible case on the known cases. We assign a probability of a certain position, or certain form by computing the ratio between the number of occurrences of a case and the total number of cases. We have not rigorously checked if the estimates are correct. As the corpus we have worked with has offered thousands of examples, by assuming a normal distribution, we may be confident in those estimates. We will return to this matter in Section 6 "Conclusion and Further Work".

3 System Architecture

In this section we present the system architecture and briefly address the main points of each one of the building blocks. In Figure 5 the double line bordered rectangle represents procedure names. The arrows point to the input and output of these procedures which may be regular intermediate data files (simple bordered rectangles) or resources (cylinders).

For best results, the system requires the presence of two external resources: a name "dictionary" and the "list of most frequent token names". Their dimension and contribution has been discussed in the previous section. The central role in the system is played by the procedure "last_name_classifier". Section 4 is entirely dedicated to it.

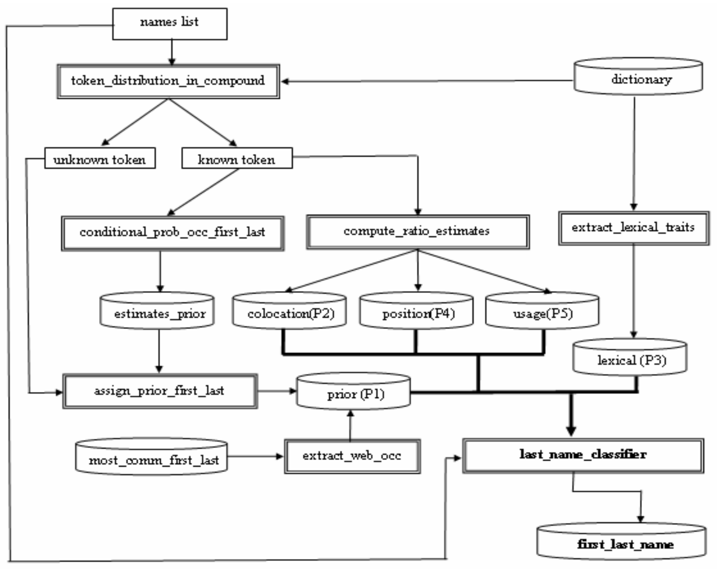


Fig. 5. System Architecture

- "token_distribution_in_compound" receives the list of name mentions, and, for those with the length greater than two tokens, it computes fives distributional values for each token, D_0, D_1, D_2, D_3, D_4 (see section 2.1). It marks the tokens existing in the "dictionary", which can be "*not_last*", "*last*" or "*ambiguous*".
- "conditional_prob_occ_first_last" finds estimates of the prior probability, P1, according to the values D_i. Suppose we have *n* tokens in "known_token". While estimates could be computed considering the whole *n* tokens, we have preferred to compute estimates on groups of 1500 tokens, and to see whether the within variation leads to the acceptance of the null hypothesis, H_0 – equal means. Fortunately this is the case.
- "compute_ratio_estimates" computes the values for P_2,P_4, P_5 as frequency ratios.
- "assign_prior_first_last" assigns prior probabilities to the unknown tokens. We used a min max approach that is presented in the next section.
- "extract_web_occurence" implements the pseudo-code presented in Figure 3.
- "extract_lexical_traits" implements the pseudo-code presented in Figure 4.
- "last_name_classifier" combines the information coming from the values of the five parameters in order to choose the actual category of each of the tokens compounding a name.

The output of the system is the resource “first_last_name” where each token within a name is labeled with each category. Not all the cases are resolved by this approach. The cases which are left out cannot be solved by the distributional properties of names. Consider for example the names formed by two equally ambiguous tokens having just one occurrence, or names where all tokens are abbreviations. The best guess is that the real categorization is given according to the *pnorm*[2]. The system has the capacity to identify those cases with no distributional resolution.

4 last_name Classifier

We present three different algorithms for the last_name classifier: one based on conditional probabilities, one which implements a neuronal network (NN), and one which implements a SVM. The training set is made considering three quarters of the known cases. The comparison among these three approaches is carried in the next section using two baselines: one that assigns the last_name for the rightmost token, “baseline_right” and a SVM with just one feature – the token’s category – “baseline_SVM”.

4.1 Min Max Estimates

The output of “conditional_prob_occ_first_last” is (D_0, D_1, D_2, D_3, D_4), a 5-tuple for each known token: the number of occurrences in the rightmost position, the number of occurrences in the leftmost position, the number of occurrences on the right of a known not_last name, the number of occurrences on the left of a known last_name. Firstly, each value is divided by the total number of occurrences of the token. Secondly, each ratio is rounded to its closest first decimal, such as that the values of each D_i are normalized in ten intervals .We obtain individual estimates of the probability of last_name for each normalized value of D_i. The estimates are computed as the mean of six groups of known last_name and not_last_name, consisting of 1 500 elements each.

To assign the prior probabilities to the unknown token we used a min max approach with the formula presented in Figure 6.

$$\hat{p}(\text{last\_name\_}D_1\text{\_}D_3) = \max(D_1, D_3)$$
$$\hat{p}(\text{not\_last\_name\_}D_2\text{\_}D_4) = \max(D_2, D_4)$$

$$\hat{p}(\text{last\_name}) = \min(\text{last\_name\_}D_1\text{\_}D_3,\ 1 - \text{not\_last\_name\_}D_2\text{\_}D_4)$$
$$\hat{p}(\text{not\_last\_name}) = \min(\text{not\_last\_name\_}D_1\text{\_}D_3,\ 1 - \text{last\_name\_}D_1\text{\_}D_3)$$

```
foreach name in name_list
  foreach token in name_list
    if (contain(token, lexical_traits))
      assign category;
    endif
    else
    assign min max estimates category;
    endelse
  endfor
endfor
```

Fig. 6. Min max estimates and category assignment using them

[2] The probability of being right in cases with one occurrence is given by the exponential distribution, the waiting time to the first success for a Poisson distribution, with the mean $\mu = 1 - pnorm$.

The estimates are real numbers. The category assignments can be done using thresholds computed on known cases. The last_name classifier uses the lexical traits as the most informative element. In the cases in which this information is missing, the assignment is done by considering the estimates computed as above.

4.2 Feed Forward Neuronal Network

A feed forward neuronal network can handle the relationship between the first four parameters, P*i*. We have implemented a feed forward Neural Network of the type "winner-take-it-all" with the pocket algorithm. The NN computes for each token its weighted score. The fifth parameter, the usage form, is used as a checking test. The test is a perceptron that has input the weighted scores of each token and the probabilities of the usage form (the number of input cells is *nr_tokens* + *nr_possible_forms*).

Figure 7 shows the NN we have used. Node H_2 and node H_3, which are hidden nodes, can be computed separately, with the consequence of having a uni-layer perceptron. We have preferred the multi-layer version, as the computational weight is low.

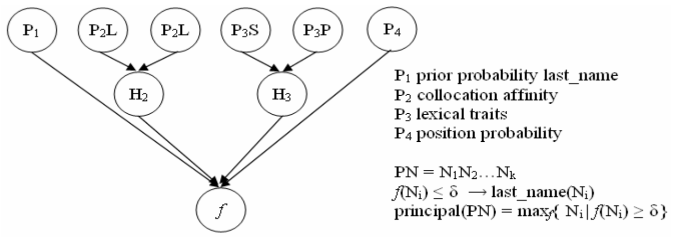
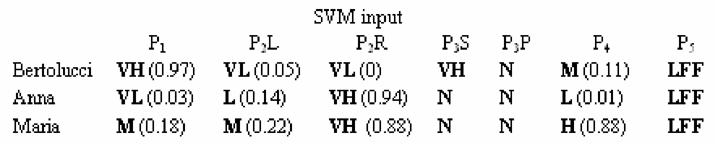

P_1
P_2L
P_2L
P_3S
P_3P
P_4
H_2
H_3
f
P_1 prior probability last_name
P_2 collocation affinity
P_3 lexical traits
P_4 position probability
$PN = N_1N_2\ldots N_k$
$f(N_i) \leq \delta \longrightarrow \text{last\_name}(N_i)$
$\text{principal}(PN) = \max_f\{ N_i \mid f(N_i) \geq \delta \}$

Fig. 7. Feed Forward NN for the last_name classifier

A side effect is to identify the "principal" last_name token. If a name has more than one last_name the "principal" last_name is the one that is used by itself to refer to the person carrying that name.

4.3 SVM

We have used a public available SVM implementation, known as Yamcha. Among various scenarios that could be conceived we have chosen the one in which the features are exactly the five parameters, P*i*. The values of P_4, P_5 are fixed and there is no problem to treat them as strings. The values of P_1, P_2, P_3 are variable, thus they must



SVM input

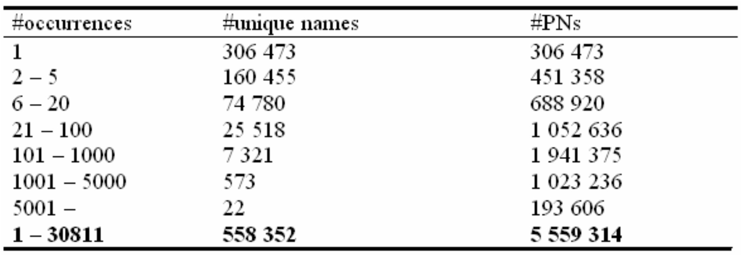
	P_1	P_2L	P_2R	P_3S	P_3P	P_4	P_5
Bertolucci	**VH** (0.97)	**VL** (0.05)	**VL** (0)	**VH**	**N**	**M** (0.11)	**LFF**
Anna	**VL** (0.03)	**L** (0.14)	**VH** (0.94)	**N**	**N**	**L** (0.01)	**LFF**
Maria	**M** (0.18)	**M** (0.22)	**VH** (0.88)	**N**	**N**	**H** (0.88)	**LFF**

Fig. 8. SVM input example

be normalized. We have used five categories for normalization: very low (VL), low (L), medium (M), high (H), very high (VH); P_3 (N) stands for "no clue" when the value of a particular lexical trait is unknown. In Figure 8, the example "*Bertolucci Anna Maria*" in the input format for SVM is shown.

5 Data and Evaluation

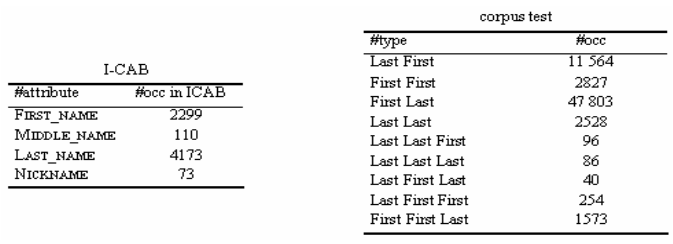
The experiments have been carried on the news corpus from a local Italian newspaper spanning a period of 7 years. The names are extracted using a NER model based on Yamcha (Zanolli&Pianta 2006) (see Figure 9).



#occurrences	#unique names	#PNs
1	306 473	306 473
2 – 5	160 455	451 358
6 – 20	74 780	688 920
21 – 100	25 518	1 052 636
101 – 1000	7 321	1 941 375
1001 – 5000	573	1 023 236
5001 –	22	193 606
1 – 30811	**558 352**	**5 559 314**

Fig. 9. Name Mentions and their distribution in the Adige corpus

We evaluate the performances of the algorithms presented in Section 4 on I-CAB. I-CAB is a gold standard four day news corpus manually annotated for cross document coreference. We are interested only in the names occurring in I-CAB (see Figure 10). For our purposes the middle names have been considered first names and nicknames. A second evaluation has been carried on a test corpus built from corpus. The surest cases – the known unambiguous items – have been divided in two parts such that the number of different cases in one group is a quarter of the total number. The "quarter" has been used for the testing.



I-CAB

#attribute	#occ in ICAB
FIRST_NAME	2299
MIDDLE_NAME	110
LAST_NAME	4173
NICKNAME	73

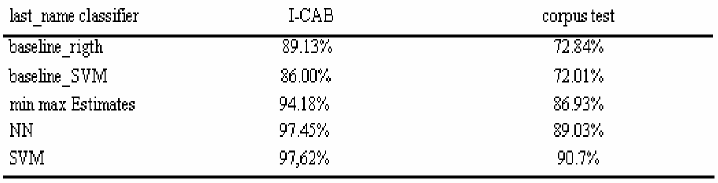
corpus test

#type	#occ
Last First	11 564
First First	2827
First Last	47 803
Last Last	2528
Last Last First	96
Last Last Last	86
Last First Last	40
Last First First	254
First First Last	1573

Fig. 10. I-CAB and test extracted from corpus

The dictionary covers 87.3% of the first names and 91.8% of the last names occurring in I-CAB. There are 245 possibly ambiguous names, out of which 9 are actually ambiguously used in I-CAB, and there are 13 ambiguous abbreviations. In I-CAB there are also 248 occurrences of free first names and 2119 occurrences of free last names (a free name is a one token name).

The baseline_right assigns the last_name category to the rightmost element and the most frequent one to all the other tokens. The baseline_SVM is a model constructed on ¾ of the all known names using just one feature: the token category. In Figure 11 we present the results obtained on the I-CAB and the corpus test respectively.



last_name classifier	I-CAB	corpus test
baseline_rigth	89.13%	72.84%
baseline_SVM	86.00%	72.01%
min max Estimates	94.18%	86.93%
NN	97.45%	89.03%
SVM	97,62%	90.7%

Fig. 11. Evaluation Results

As I-CAB is a four days news corpus, the difficult cases are not well represented. Therefore, a baseline may score well. The last column in Figure 11 shows that this is not what happens in the whole corpus. A safe estimation will be that at least 25% of all different names may need a special treatment.

The parameters introduced in Section 2 are indeed relevant for the task. We can see that "min max Estimates" is well beyond the baseline with almost 15%.

We conclude this section with further remarks on the issue raised at the end of Section 3, which is relevant for the way the evaluation is carried on. There is a drawback in the evaluation procedure. The evaluation has been carried on different names, independently of how many times the name is mentioned in the corpus. Therefore, it is assumed that all the mentions behave in the same way. And this assumption is not valid. The percentage of cases where the same name refers to two or more persons is probably low. The cases where all the tokens of a name are ambiguous, with no strong preference for just one category, are rare. They probably represent less than 1% of the whole corpus. However, the correct treatment of such cases goes beyond the scope of this paper. The relevant information may come in some of the cases from a local coreference module, a module which establishes the coreferences among the name mentions in the same piece of news (Popescu&Magnini 2007).

6 Conclusion and Further Work

In this paper we have presented a system for a first vs. last name identification task. Our results show that this is not a trivial case in at least 25% of the cases. However, it seems that good results can be obtained by using the distributional properties of the tokens occurring in names. The best combination scored 90.7%, approximately 20% better than the baseline. We consider that this is possible due to the use of the five parameters which we introduced in section 2.

There are few directions we would like to focus on in the near future. The first one is related to the fact that even in a local newspaper, many of the names are foreign.

We will try to cluster the names according to nationality and to evaluate the values of the parameters within each cluster. We have already undertaken the first experiments in this direction. Secondly, we would like to also include middle names, for languages where this category exists. Thirdly, there is a category of names which is problematic: the nicknames. From a distributional point of view, they behave like last_name (number of free occurrences). Yet they do not combine with other names (collocation affinity is very low). In same cases it is possible to use this peculiarity to identify them.

References

1. Driscoll, P., Yarowsky, D.: Disambiguation of Standardized Personal Name Variants. In: Proc. IWMMIES, Borovets, Bulgaria, pp. 1–7 (2007)
2. Krstev, C., Dusko, V., Maurel, D.: Multilingual ontology of proper names. Language and Technology Conference (2005)
3. Magnini, B., et al.: Ontology Population from Textual Mentions: Task Definition and Benchmark. In: Proc. OLP2 workshop on Ontology Population and Learning, Sidney, Australia, Joint with ACL/Coling (2006)
4. Mann, G.S., Yarowsky, D.: Unsupervised personal name disambiguation. In: Proc. Conference on natural Languag Learning, pp. 33–40 (2003)
5. Magnini, B., et al.: Ontology Population from Textual Mentions: Task Definition and Benchmark. In: Proc. OLP2 workshop on Ontology Population and Learning, Sidney, Australia, Joint with ACL/Coling (2006)
6. Popescu, O., Magnini, B.: Iterative Person Coreference Using Name Frequency Estimates. In: Proc. 3rd Language & Technology Conference, Poznan, Poland (2007)
7. Zanoli, R., Pianta, E.: SVM based NER, Technical Report, Trento, Italy (2006)

Mixing Statistical and Symbolic Approaches for Chemical Names Recognition

Florian Boudin[1], Juan Manuel Torres-Moreno[1,2], and Marc El-Bèze[1]

[1] Laboratoire Informatique d'Avignon
339 chemin des Meinajaries, BP1228
84911 Avignon Cedex 9, France
[2] École Polytechnique de Montréal - Département de génie informatique
CP 6079 Succ. Centre Ville H3C 3A7
Montréal (Québec), Canada
{florian.boudin,juan-manuel.torres,marc.elbeze}@univ-avignon.fr
http://www.lia.univ-avignon.fr

Abstract. This paper investigates the problem of automatic chemical Term Recognition (TR) and proposes to tackle the problem by fusing Symbolic and statistical techniques. Unlike other solutions described in the literature, which only use complex and costly human made ruled-based matching algorithms, we show that the combination of a seven rules matching algorithm and a naïve Bayes classifier achieves high performances. Through experiments performed on different kind of available Organic Chemistry texts, we show that our hybrid approach is also consistent across different data sets.

Keywords: Term Recognition, Text Mining, Chemical Informatics.

1 Introduction

Over one million new chemical compounds are discovered and published annually. As in many scientific domains, the Organic Chemistry (OC) data are not published coherently but scattered through thousands of different journal articles. Identifiying and extracting chemical compounds is a critical task for chemical information retrieval. Information extraction technology arose in response to the need for efficient processing of documents in specialized domains. Classical Natural Language Processing (NLP) tools such as parsers, taggers or chunkers achieve very poor on OC documents. This is due to the specificity of the domain, a very wide vocabulary, long sentences containing a high quantity of "hapax legomen"[1]. Scientists, especially chemists, want to be able to search for articles related to particular chemical compounds. Nowadays, search engines mainly depend on the "classical" title, author(s) and keywords scheme searching. Extracting chemicals from texts and using them to classify, organize and accelerate the information access fit to a wide range of possible applications. Chemical compounds are, in articles, identified by verbal depictions (i.e. name, identifiers, formulae) but also pictorial depictions (chemical

[1] Terms which only appears once in a text.

structure representations). From the analysis of several articles we have found that most chemical compounds can be automatically extracted by examining chemical texts and verifying the presence of specific patterns. In this work, we propose an hybrid approach combining pattern matching and probabilistic classification. This paper is organized as follows. Section 2 overviews the related work, section 3 defines what we consider as a chemical compound. The two approaches and their combination are described in section 4. Experimental settings are presented in section 5 followed by the results while the section 7 concludes this paper.

2 Related Work

Nowadays, the majority of information extraction approaches in the life sciences have focused on molecular biology and genomics information so far [1]. Only a very limited number of named entity recognition approaches are described in the literature for the recognition of chemical compounds. A rule-based method was introduced by [2]. This approach was tested only on a very small benchmark set (158 chemical terms to be identified, f-measure between 0.7619 and 0.8169, see section 5.3 for details on performance measures). Other systems used simple dictionary matching without any evaluation of the performance [3]. Chemical Formulae extraction using Support Vector Machines (SVM) classification [4] and reconstruction of molecular structure by analyzing chemical terminology [5] have also been tried. These approaches tackle the issue of a different problem. As far as we know, there are no current published works on the adaptation of such statistical text mining techniques to process organic chemical papers.

3 What Is a Chemical Compound?

One of the most difficult part is to define what is a chemical compound and what it is not. We have to cope with a large variety of syntactical and semantically different compound description. The International Union of Pure and Applied Chemistry (IUPAC)[2] is mostly well-known as the recognized authority in developing standards for the naming of the chemical elements and their compounds, through its Interdivisional Committee on Terminology, Nomenclature and Symbols (ICTNS). The IUPAC nomenclature is a useful resource for naming chemical compounds and for describing the science of chemistry in general. Chemicals can be described in literature by trivial names (e.g. brand or trade names), by registry numbers (e.g. database identifiers), by systematic naming schemes (e.g. nomenclature such as IUPAC [6] or formal descriptions like SMILES [7]) and by chemical structure depictions. Rules for naming organic compounds are contained in one publication, known as the Blue Book [8]. Compounds are named by using a number of prefixes, suffixes and infixes that support very precise information about them (i.e. type and position of functional groups, priority, etc...). For example, the compound `2-methylpropane` is composed by the root names `prop-` and `meth-` corresponding to the number of carbons in the main chain and

[2] http://www.iupac.org

the attached chain respectively. The main chain is a propane chain and a methyl group is bonded (attached) to the middle (2) carbon, these specifications give the systematic name: `2-methylpropane`. In articles, `2-methylpropane` is commonly called as `isobutane` but can also be `(CH3)2CHCH3`. To illustrate the large variety of synonyms, the chemical `2-methylpropane` has officially 12 synonyms (in which `Trimethylmethane`; `1,1-Dimethylethane`; `iso-C4H10`; `i-Butane`; `Isobutane mixtures`; `tert-Butane`; Methyl`propane`; `2-methyl-isobutane` `Propane`) but the number of variants can be as high as several hundred. All these variants correspond to the same compound and have to be identified. This example gives a flavour of the tremendous difficulty of the task.

4 An Hybrid Approach

In this section, we describe two different approaches we used for chemical names identification and we explain why we choose to combine them.

4.1 Pattern Matching

The first approach consists in manually writing a small pool of patterns based on the Blue Book nomenclature. The system skims through the document verbatim and tries to capture the chemical compounds. The presence of specific prefixes, suffixes, infixes, numbers and special characters (such as brackets or Greek letters) in a term allows our system to identify facile terms (e.g. high probability to be a chemical name). We consider a term T as a token separated by two spaces. The score S_{pm} of a term T to be a chemical compound is calculated as:

$$S_{pm}(T) = \sum_{j=0}^{N} Match_j(T)$$

$$Match_j(T) = \begin{cases} \omega_j & \text{if the pattern } j \text{ match the term } T \\ 0 & \text{else} \end{cases}$$

N is the total number of rules/patterns, $\sum_j \omega_j = 1$ and $\omega_j \in [0, 1]$. Assuming a uniform weights distribution (i.e. weights ω_j are equally spread according to the number of rules), a term is considered to be a chemical compound if at least one rule is matching (i.e. if $S_{pm}(T) \geq 0$). The higher is $S_{pm}(T)$, the higher is the number of patterns matching with the term T and as a result the higher is the likelihood to be a chemical compound. The seven rules given below compose the pool of patterns implemented in our system.

1. Presence of a morpheme indicating the number of carbon atoms (40 patterns): (`*meth*`, `*eth*`, `*propa*`, `*buta*` ...)
2. Presence of a specific suffix (58 patterns): (`*ane`, `*yne`, `*thiol`, `*oate`, `*amine` ...)
3. Presence of a numbering prefix/infix (locant): (`1,3-*`, `2,3,5-*`, `*-2-*`, `[4,5-b]*` ...)

4. Presence of a multiplying prefix (10 patterns):
   (`tri*`, `tetra*`, `penta*` ...)
5. Presence of a ambiguity prefix (3 patterns):
   (`iso*`, `sec*`, `tert*` ...)
6. Presence of a specific infix (46 patterns):
   (`*chlor*`, `*phosphor*`, `*amin*` ...)
7. Presence of specific Caps and Numbers patterns:
   (`AcOH`, `NH4OAc`, `DMFDMA` ...)

4.2 The Bayes Classifier

The second approach uses a probabilistic classifier based on applying Bayes' theorem with strong independence assumptions [9]. The instances to be classified are described by attribute vectors $\overrightarrow{a} = (a_1, a_2..., a_n)$. The overlaping n-grams of letters ($n = 3$) are used to train the classifier. For example, the term `2-methylpentane` will be splitted in thirteen 3-grams (e.g. `2-m`, `-me`, `met`, `eth`, `thy`, `hyl`, `ylp`, `lpe`, `pen`, `ent`, `nta`, `tan` and `ane`). The use of 3-grams representing the first/last two characters of a term (respectively `**2`, `**2-`, `ne*` and `e**` for the example above) have been experimented but finally not retained[3]. The Bayes classifier assigns to an instance the most probable –or maximum *a posteriori*– classification from a finite set C of classes:

$$C_{map} \equiv \underset{c \in C}{argmax}\ P(c|\overrightarrow{a}) \tag{1}$$

Which after applying Bayes' theorem can be written

$$C_{map} = \underset{c \in C}{argmax}\ P(c)P(\overrightarrow{a}|c) \tag{2}$$

We choose to define each attribute a_i as one of the 3-grams that compose the term T. The finite set C is composed by two classes: c and ¬c (e.g. chemical and not chemical). We need to estimate the probability of a certain 3-gram $a_i = (w_{i-2}, w_{i-1}, w_i)$ occurring in a class c.

$$P(w_i|w_{i-2}w_{i-1}c) \tag{3}$$

The posterior probabilities could be estimated directly from the training data using Laplace smoothing to avoid zero probabilities. With this assumption, Equation (2) becomes the Bayes classifer.

$$C_{map} = \underset{c \in C}{argmax}\ P(c)\prod_i P(w_i|w_{i-2}w_{i-1}c) \tag{4}$$

4.3 Combination of the Approaches

Although successful, the first approach (c.f section 4.1) is limited by the tremendous variety of chemical names in literature. As a consequence, the overall performance is below the Bayes classifier. The classification approach is more

[3] These 3-grams being not discriminant introduce misclassifications.

accurate and achieves good results (see section 6). Since our main goal is to produce a system with a very high precision, the choice was hence made to try a combination of the two approaches. The basic idea implemented by the hybrid method is that of "voting" or "recommendation". When one term is classified as chemical compound and at the same time is matched by at least one rule then the term is validated as chemical compound. The combination is hoping to increase the precision to a very high score by removing misclassification errors. The price to paid for an increase of precision will be a fall of recall, only the intersection of the two term classes is considered.

5 Experimental Settings

The method described in the previous section has been implemented and evaluated on a testing corpus. In the following subsections, details of the experimental settings are described.

5.1 Classifier Training

Training the parameters requires (i) creating two vocabulary sets, e.g., a chemical coumpound name set V_c and a non-chemical set $V_{\neg c}$, (ii) estimating the n-gram probabilities by calculating the n-gram occurrences. The chemical compound name vocabulary V_c was created from a CAS[4] database of about 10K compounds. For each chemical compound a query has been sent to the online database: http://webbook.nist.gov and by parsing web pages all differents names (synonyms) have been obtained. The resulting V_c is composed by nearly 65K compound names. The non-chemical vocabulary $V_{\neg c}$ was created using the SCOWL (Spell Checker Oriented Word Lists) corpus [5]. The reasons of using the SCOWL corpus are (1) to avoid the non-chemical vocabulary to contain any chemical compound names or errors, and (2) to easily gather a large quantity of n-grams. The n-gram probabilities were estimated from the occurrence frequencies inside the vocabulary sets. Two training data sets for chemical names (called `Small Voc` for 10K and `Large Voc` for 65K) and ten of increasing size for non-chemical words have been experimented.

5.2 Test Data

In order to evaluate our approach across real-life data sets, we have constructed a test data set composed by abstracts and plain articles. The test corpus is composed by 12 annotated abstracts extracted from the Beilstein Journal of

[4] Chemical Abstracts Service (CAS), a division of the American Chemical Society, assigns these identifiers to every chemical that has been described in the literature. CAS registry numbers are unique numerical identifiers for chemical compounds, polymers, biological sequences, mixtures and alloys.

[5] http://wordlist.sourceforge.net/

Organic Chemistry[6] RSS feed and 8 plain articles coming from different journals (Organic Letters and Accounts of Chemical Research[7]) of different years (respectively 2000-2002 and 2005-2007), different authors and topics. The corpus has been annotated by two different annotators and validated by a domain specialist. Corpus size is approximately 20,000 terms in which 850 chemical compounds were manually identified. For the abstracts, there are 2,700 words in which 170 chemical compounds and for the plain articles there are 17,300 words in which 680 chemical compounds.

5.3 Performance Measures

The following performance measures are considered relevant.

Precision. It is the proportion of retrieved and relevant chemical compounds to all the compounds retrieved.

Recall. It is the proportion of retrieved and relevant chemical compounds, out of all relevant compounds.

***f*-measure.** It is the weighted harmonic mean of precision and recall. The traditional *f*-measure or balanced *f*-score is:

$$f\text{-measure} = \frac{2 \cdot (Precision \cdot Recall)}{(Precision + Recall)}$$

6 Experimental Results

Figure 1 shows the results of Precision, Recall and *f*-measure for the expression based pattern matching (c.f section 4.1) according to the rule used and their incremental combination (i.e. the combination in rule 3 means using rules 1,2 and 3). The observed results confirm the limitations of the approach. Indeed, the huge variety of different writing schemes used for chemical compounds makes impossible to obtain a full recall. We observe that each rule allow an increase of the f-measure, this means that all rules are “useful” (allow to increase the classification performance). Rules 6, 2 and 1 are the best-score rules. This is interesting because it is not the logical order according to the number of patterns contained in each rule (58 for rule 2, 48 for rule 6 and 40 for rule 1). It indicates that the presence of a specific infix (rule 6) is more discriminant than the presence of a morpheme indicating the number of carbon (rule 1) or the presence of specific suffixes (rule 2).

[6] The Beilstein Journal of Organic Chemistry is an Open Access, peer-reviewed online journal that will encompass all aspects of organic chemistry. The journal covers organic chemistry in its broadest sense, including: organic synthesis, organic reactions, natural products chemistry, supramolecular chemistry and chemical biology. http://bjoc.beilstein-journals.org/home/

[7] http://pubs.acs.org

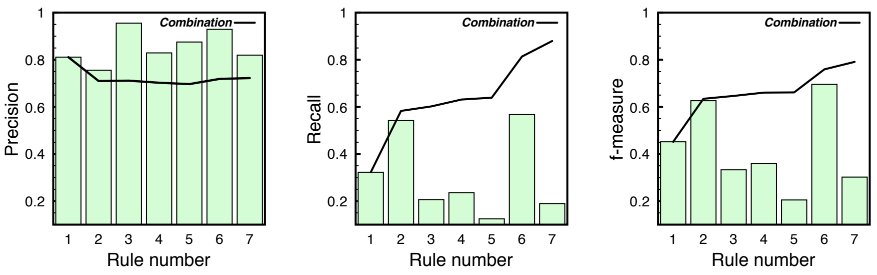


Fig. 1. Performance of the pattern matching approach in relation to the rule used. The performance of the incremental combination is also shown (black line).

One might expect that the performance of the classifier would improve as the size of the training corpus increases, because a larger training corpus usually leads to a better estimation of the n-gram probabilities. In fact, Figure 2 shows that once the corpus size reaches 40% (5850 different 3-grams), f-measures of both Small and Large chemical training sets (respectively `Small Voc` and `Large Voc`) remain obviously at the same values. This is due to the fact that the terms containing in `Large Voc` have been obtained automatically (c.f see section 5.1) and so non-chemical terms have been introduced in the chemical training set.

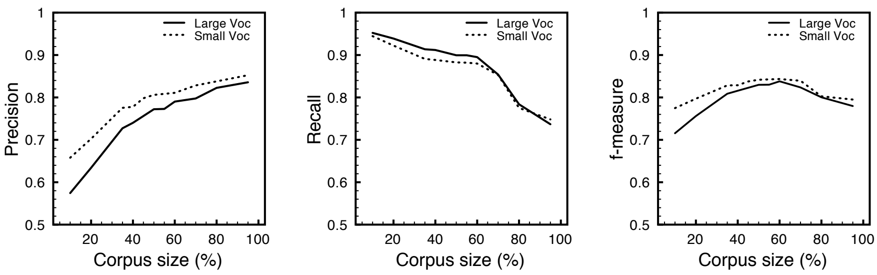


Fig. 2. Performance of the classifier vs. the size of the training corpora

Figure 3 shows the results of Precision, Recall and f-measure of the rule-based and classifier approaches compared to their combination. We can observe that the combination significantly increase the precision (0.92839 against 0.72224 for the rule-based and 0.79015 for the classifier) and what ensued logically outperforms the best approach alone in f-measure (0.87701 against 0.79099 for the rule-based and 0.83801 for the classifier). This is a very interesting result because we can infer that approaches are complementary and can be combined without any consequent decrease of recall. We can extrapolate and suppose that the Entity Recognition in chemical texts may be broken up into sub-tasks solvable by slightly different but complementary approaches.

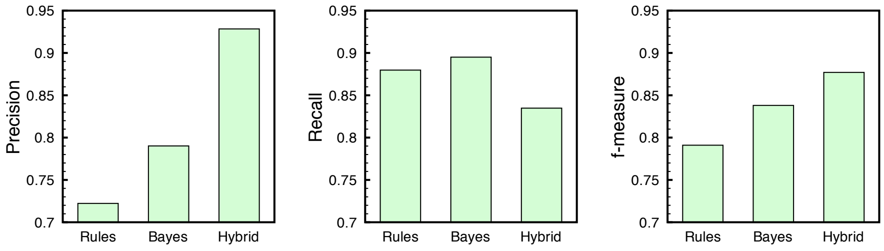


Fig. 3. Performance of the rule-based and classifier approaches compared to their combination

We have performed experiments on the two different kind of available corpus, i.e. abstracts and articles. Table 1 compares the performance of the hybrid method on the abstracts and on the articles. Our hybrid approach is consistent across the different data sets, the precision being in both case very high. The lack of recall in articles can be explained by the high proportion of trivial names (e.g. brand or trade names) that are not well recognized by our rule-based approach and also by the difference of chemicals proportion in data sets (6.29% for abstracts and 3.93% for articles). An *a priori* adaptation of the Bayes classifier's training corpora by tuning the ratio between the probabilities of the two vocabulary sets ($P(c)$ and $P(\neg c)$) has shown to increase the scores, this technique will be developed in further works.

Table 1. Performance score of the hybrid method on the two kinds of corpus, i.e abstracts and articles

	Precision	Recall	f-measure
Abstracts	**0.88333**	**0.93529**	**0.90857**
Articles	*0.93402*	*0.82221*	*0.87306*

We have made an *a posteriori* error analysis and have observed that the terms not detected by our systems are essentially historical/common/brand names such as alumina, salt or pipecolate. These names are very difficult to be recognized because of their belonging in the two classes (c and $\neg c$) and because of their structures (not containing discriminant patterns/structures).

7 Conclusion and Future Work

We have described an hybrid method for chemical entity recognition that combines a simple rule-based pattern matching (seven rules) and a naïve Bayes classifier. Through experiments performed on different kind of available Organic Chemistry texts, we have showed that our hybrid approach is also consistent across different data sets. These results are promising, and represent a good starting point for future research but do show a critical point: the unstoppable growth of the number of different chemical compounds in the literature. As a consequence, Information Extraction (IE) approaches are more than ever required by life scientists to ensure an optimal sharing of the information. Among the others, there are several points that would be worthy of further investigation:

– Improve the estimation of probabilities by using smoothing techniques for unseen n-grams [10].
– Run experiments on different kinds of non-chemical corpora or different n-grams sizes and measure their impacts.
– Explore the usage of alternative combinations: combining the approaches in another way.
– Fuse chemical entity recognition with a domain-specialized automatic summarization system [11] as a domain-specialized weighted metric (i.e. the number of chemical compounds within a sentence is used as a parameter by the sentence scoring algorithm).

Acknowledgment

We are grateful to Patricia Velázquez-Morales for making available the test collection and for her help with this data set and to Pr. Alain Krief and Julie Henry for our useful talks. This work was partially supported by the *Laboratoire de chimie organique de synthèse*, FUNDP (*Facultés Universitaires Notre-Dame de la Paix*), Namur, Belgium.

References

1. Fluck, J., et al.: Information Extraction Technologies for the Life Science Industry. Drug Discovery Today–Technologies 2(3), 217–224 (2005)
2. Narayanaswamy, M., Ravikumar, K.E., Vijay-Shanker, K.: A biological named entity recognizer. Pac. Symp. Biocomput. 427, 38 (2003)
3. Singh, S.B., Hull, R.D., Fluder, E.M.: Text Influenced Molecular Indexing (TIMI): A Literature Database Mining Approach that Handles Text and Chemistry. J. Chem. Inf. Comput. Sci 43(3), 743–752 (2003)
4. Sun, B., et al.: Extraction and search of chemical formulae in text documents on the web. In: WWW 2007: Proceedings of the 16th international conference on World Wide Web, pp. 251–260. ACM Press, New York (2007)
5. Reyle, U.: Understanding chemical terminology. Terminology (Amsterdam) 12(1), 111–136 (2006)
6. Panico, R., Powell, W.H., Richer, J.C.: A guide to IUPAC nomenclature of organic compounds (recommendations 1993). Blackwell Science, Malden (1993)
7. Weininger, D.: SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. Journal of Chemical Information and Computer Sciences 28(1), 31–36 (1988)
8. Rigaudy, J., Klesney, S.P.: Nomenclature of organic chemistry(sections A, B, C, D, E, F, and H). Pergamon Press, Oxford (1979)

9. Rish, I.: An empirical study of the naive Bayes classifier. In: Proceedings of IJCAI-2001 Workshop on Empirical Methods in Artificial Intelligence, vol. 335 (2001)
10. Manning, C.D., Schütze, H.: Foundations of statistical natural language processing. MIT Press, Cambridge (1999)
11. Boudin, F., Torres-Moreno, J.M.: NEO-CORTEX: A Performant User-Oriented Multi-Document Summarization System. Computational Linguistics and Intelligent Text Processing, 551–562 (2007)

Portuguese Pronoun Resolution: Resources and Evaluation

Ramon Ré Moya Cuevas, Willian Yukio Honda, Diego Jesus de Lucena, Ivandré Paraboni, and Patrícia Rufino Oliveira[*]

[*] Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (EACH / USP)
Av.Arlindo Bettio, 1000 - 03828-000, São Paulo, Brazil
{fusion,kio,diego.si,ivandre,proliveira}@usp.br

Abstract. Despite being one of the most widely-spoken languages in the world, Portuguese remains a relatively resource-poor language, for which only in recently years NLP tools such as parsers, taggers and (fairly) large corpora have become available. In this work we describe the task of pronominal co-reference annotation and resolution in Portuguese texts, in which we take advantage of information provided by a tagged corpus and a simple annotation tool that has been developed for this purpose. Besides developing some of these basic resources from scratch, our ultimate goal is to investigate the multilingual resolution of Portuguese personal pronouns to improve the accuracy of their translations to both Spanish and English in an underlying MT project.

1 Introduction

Pronoun resolution – the task of identifying the antecedent of a pronoun in discourse - is often crucial to a variety of NLP applications, ranging from Text Summarization to Machine Translation (MT), and it has long been recognised as a challenging computational problem for which existing approaches - either making use of learning techniques or otherwise - often need to resort to large amounts of knowledge produced by standard NLP tools such as parsers and POS taggers applied on annotated corpora [1].

The challenge is however considerably increased if we speak of languages for which some of these basic resources are still under development, or may have only recently become available. This is the case, for instance, of Portuguese, one of the most widely-spoken languages in the world, and which still lacks somewhat behind as a relatively resource-poor language in NLP.

Our own work focuses on pronoun resolution as required by a Portuguese-Spanish-English MT project under development. Our present choice - Portuguese third person plural pronouns ("Eles/Elas") - is based on the assumption that these (as well as their Spanish counterparts) are less prone to ambiguity, and arguably easier to resolve than the English equivalent ("They"), which may suggest an interesting multilingual approach to anaphora resolution not unlike [7].

As a first step to boost translation performance in these languages, we describe the construction of some basic resources for Portuguese (namely, a co-reference annotation tool, an annotated corpus and training data derived from tagged text) as reusable components to be made available to the research community. Secondly, we evaluate the usefulness of our preliminary data in two standard machine learning approaches to pronoun resolution (statistical / unsupervised and symbolic / supervised).

The rest of the paper is structured as follows. Section 2 describes our corpus annotation work and the annotation tool that has been developed. Sections 3 describes the co-reference resolution task as a classification problem and the generation of training instances. Section 4 presents two experiments on the usefulness of the training data and Section 5 discusses our efforts so far and future work.

2 Corpus Annotation and Tools

We selected the Portuguese portion of an English-Portuguese-Spanish parallel corpus comprising 646 articles (440,690 words in total) from the Environment, Science, Humanities, Politics and Technology supplements of the on-line edition of the "Revista Pesquisa FAPESP", a Brazilian journal on scientific news. The resulting corpus was tagged using the PALAVRAS tool [2].

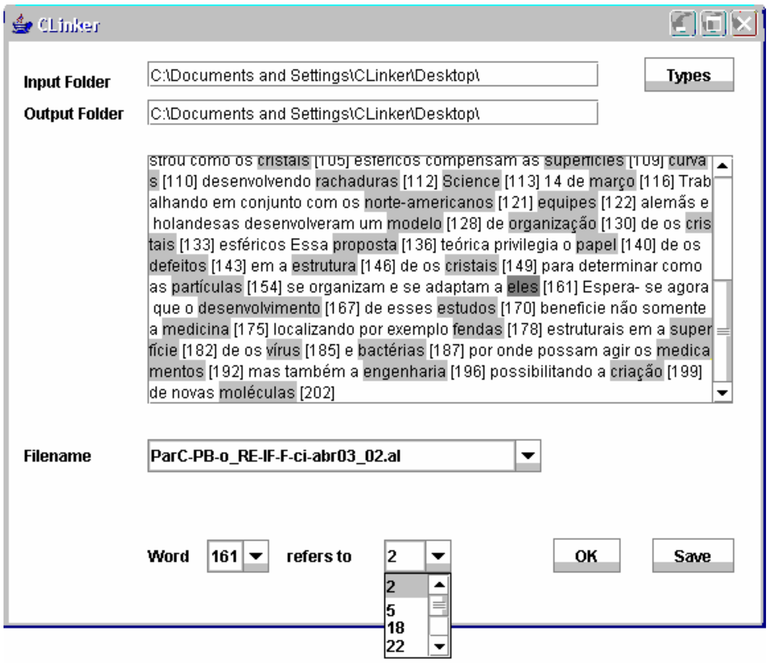
Given our long-term goal of investigating multilingual anaphora resolution as a means to improve the performance of a statistical machine translation system, we focused on third person plural pronouns (male) "Eles" and (female) "Elas", which are both translated as (no gender-specific) "They" in English. 813 instances of such pronouns (584 male and 229 female) were found in our corpus.

For the co-reference annotation task proper, we first considered using an existing annotation tool such as MMAX [3]. However, in order to take advantage of the (Portuguese) information made available by PALAVRAS, we developed a simple co-reference annotation tool from scratch. Besides providing the basis for our training data described in the next section, the use of the existing tags allowed us to automatically constrain the choices to be made by the human annotator regarding both referring expressions (which are user-defined) and potential antecedents (taken to be the existing NPs etc.). A screenshot of the annotation tool GUI is shown below:

The tool allows the annotator to specify an input folder in which the corpus is to be found, and an output folder to which the annotation will be saved in stand-off format. As in many existing annotation tools, each text from the input folder is displayed in turn, and both referring expressions and potential antecedents are highlighted by a particular colour scheme (rendered in black and white above) and unique indexes to guide the annotation. The user specifies a list of target referring expressions (types) and then browses each input file by selecting the antecedent for each expression. The tool computes the total number of references in each text and the number of references already annotated, allowing the user to pause and resume the current work at all times, switch to another text etc.

Two independent annotators used the tool to link each of the selected instances of reference to their antecedents in the text, except for the cases of reference to compound antecedents (e.g., "John and Mary") which are not presently addressed.

Following the annotation task, the annotators compared their data and excluded all instances of reference on which they could not immediately reach agreement. This was mainly the case of errors introduced by the tagger itself (e.g., unidentified NPs) and ill-formed or ungrammatical sentences. As a result, we arrived at a set of 483



Input Folder C:\Documents and Settings\CLinker\Desktop\

Output Folder C:\Documents and Settings\CLinker\Desktop\

strou como os cristais [105] esféricos compensam as superfícies [109] curvas [110] desenvolvendo rachaduras [112] Science [113] 14 de março [116] Trabalhando em conjunto com os norte-americanos [121] equipes [122] alemãs e holandesas desenvolveram um modelo [128] de organização [130] de os cristais [133] esféricos Essa proposta [136] teórica privilegia o papel [140] de os defeitos [143] em a estrutura [146] de os cristais [149] para determinar como as partículas [154] se organizam e se adaptam a eles [161] Espera- se agora que o desenvolvimento [167] de esses estudos [170] beneficie não somente a medicina [175] localizando por exemplo fendas [178] estruturais em a superfície [182] de os vírus [185] e bactérias [187] por onde possam agir os medicamentos [192] mas também a engenharia [196] possibilitando a criação [199] de novas moléculas [202]

Filename ParC-PB-o_RE-IF-F-ci-abr03_02.al

Word 161 refers to 2

2
5
18
22

Fig. 1. A co-reference annotation tool based on Portuguese tagged corpora. The antecedent term of pronoun 161 is selected from a list of possible candidates, which is automatically constrained by the information provided by the tagger.

revised instances of reference to single terms in the text. This data set is the basis of the training data described in the next section.

3 Training Data

Likewise [4], we will regard the present pronoun resolution task as a classification problem in which a pronoun *p* and a potential antecedent *a* may co-refer or not. To this end, we consider positive instances of co-reference the pairs (*p*, *a*) explicitly defined as co-referential by the annotators, and we consider negative instances all pairs (*p*, *a*) in which *a* is an intermediate NP between *p* and its actual antecedent. For example, the pronoun *p* in the following text gives rise to one positive (*p*, *a1*) and three negative ((*p*, *a2*), (*p*, *a3*) and (*p*, *a4*)) instances of co-reference[1]:

Example 1. A pronoun (*p*) with its actual antecedent (*a1*) and three intermediate NPs (*a2*-*a4*).

A researcher from Embrapa Genetic Resources and Biotechnology in Brasilia, Alessandra Pereira Fávero, has managed to select [wild species]$_{a1}$ that carry [genes]$_{a2}$ that provide [resistance]$_{a3}$ to [these diseases]$_{a4}$. But [they]$_{p}$ cannot be crossed with the cultivated ones on account of the different number of chromosomes.

[1] This example is taken from the (unused) English version of the data for illustration purposes.

Table 1. Preliminary set of features extracted from the tagged text

Feature Name	Possible Values	Description
NUMBER_AGREEMENT	*True / False*	*True* if *p* and *a* agree in number; *False* otherwise
GENDER_AGREEMENT	*True / False*	*True* if *p* and *a* agree in gender; *False* otherwise
FUNCTION_AGREEMENT	*True / False*	*True* if *p* and *a* have the same function (subject or object) in the text; *False* otherwise
DISTANCE	*0...n*	The number of sentences between p and a (0=same sentence).
PREPOSITION_TYPE	*1..3*	1=no preposition; 2="of They" ("deles"); 3="in They" ("neles");

2595 instances of co-reference were produced in this way, being 483 positive and 2112 negative, with an average of 4.4 intermediate antecedents between each pronoun and the actual antecedent. About 10% of the positive instances were set aside with their negative counterparts for testing purposes. Thus, the test data comprised 234 instances and the reminder 2361 instances (being 435 positive or co-referential, and 1926 negative or non co-referential) became our training data.

Taking advantaged of the existing information in the tagged text, we extracted a preliminary set of features of each pair (*p*, *a*) which have generally been used in related work [e.g., 1,4,5,6,7].

The NUMBER_AGREEMENT and GENDER_AGREEMENT features are self-explanatory. FUNCTION_AGREEMENT is based on the intuition that having a pronoun in subject or object position might have an effect on the likelihood of its antecedent being in the same position in a previous sentence. The DISTANCE feature accounts for the idea that the likelihood of finding the antecedent may decrease as we move further away from the referring expression. Distance values in our corpus range from 0 to 15 (mean = 1.2 and std dev = 2.6). The PREPOSITION_TYPE feature is intended to verify whether the pronoun follows a preposition denoting possession ("deles", or "theirs") or place ("neles", or "in them") and how this might impact resolution.

According to [2], the tagger PALAVRAS tentatively generates a number of semantic features that we hoped to be able to explore in our work. In particular, we intended to define features based on the tags reportedly available for human and inanimate verbs, but these could not be produced from our data. Therefore, we are aware that the present set of features is most likely insufficient for our purposes, but given that the definition of additional (e.g., semantic) features is costly, we decided to first evaluate its contribution before expanding the training data. Our preliminary evaluation consists of two experiments (based on the EICAMM model and the induction of decision-trees) described in the next section.

4 Preliminary Results

Our first experiment is based on an unsupervised statistical approach, the EICAMM (Enhanced ICA Mixture Model) in [8], which is an extension of the ICA Mixture

Model (ICAMM) proposed in [9]. Using the entire set of features described in the previous section we obtained the following results, in which 1797 (76.11%) instances were correctly classified.

Table 2. EICAMM model results based on the entire set of features

Class	Precision	Recall	F-measure
Co-referential	0.431	0.931	0.590
Non Co-referential	0.979	0.722	0.831

We tried out different subsets of the original data, and found that by omitting `FUNCTION_AGREEMENT` the results were slightly improved - 1884 instances (79.80%) were correctly classified - but there was still no gain in F-measure for the co-referential cases.

Table 3. EICAMM model results without the `FUNCTION_AGREEMENT` feature

Class	Precision	Recall	F-measure
Co-referential	0.471	0.789	0.590
Non Co-referential	0.943	0.800	0.866

Our second experiment involved the induction of decision trees as in [5]. Using ten-fold cross-validation and all the features described in the previous section, the following results were obtained.

Table 4. Ten-fold cross-validation decision-tree induction based on the entire set of features

Class	Precision	Recall	F-measure
Co-referential	0.679	0.52	0.589
Non Co-referential	0.897	0.944	0.920

In this experiment, 2045 (86.62%) instances were correctly classified, but this was mainly due to the heavy imbalance between the number of co-referential and non co-referential cases in the training data. F-measure for the co-referential cases remained nevertheless close to the previous experiment. Moreover, closer inspection revealed that nearly half (48.05%) of the co-referential instances were actually misclassified by the decision tree.

Table 5. Ten-fold cross-validation decision-tree induction results without the `FUNCTION_AGREEMENT` feature

Class	Precision	Recall	F-measure
Co-referential	0.572	0.91	0.703
Non Co-referential	0.977	0.846	0.907

To find out the source of confusion, once again several feature combinations were examined and, not surprisingly, `FUNCTION_AGREEMENT` was found to be misleading. By leaving out this feature we produced the following results.

Now despite a slightly lower (2026, or 85.81%) number of instances correctly classified, the decision-tree induction showed an improvement in F-measure for the co-referential cases, and the corresponding confusion matrix (not shown) indicates that only 39 (9.85%) of the co-referential instances and 296 (18.16%) of the non co-referential instances were misclassified.

These results suggest that - at least for this data set - there was no useful relation between the syntactic position of the pronoun and its antecedent. However, the low precision levels for co-referential cases indicate that additional features (possibly making use of semantic knowledge) are indeed required. For that reason, we decided not to verify our preliminary results against the test data, which remain reserved for future use until a fuller set of features is defined.

Finally, we notice that even though the comparison with related work may be tempting (e.g., in [5], pruned trees for general co-reference resolution obtained F-measure of 0.86 using a much larger set of features) we should point out that besides addressing only a subset of the problem investigated in [5], our decision to disregard instances of reference to compound antecedents may have simplified the task considerably, as some of the most difficult cases to be resolved may have simply been left out.

5 Final Remarks

This paper described the task of pronominal co-reference annotation and resolution in Portuguese texts as a classification problem, in which we take advantage of information provided by a tagged corpus and a simple annotation tool that has been developed for this purpose. We described the annotation task and the annotation tool, the training data that we used and some preliminary results based on both unsupervised and supervised learning methods applied to pronoun resolution with an incomplete set of features derived from tagged text.

We are now in the process of revising our set of features. This will be followed by an investigation on a variety of learning methods to produce an algorithm for Portuguese personal pronouns resolution. Once this task is accomplished, we expect to use this algorithm to improve the translation performance in Portuguese, Spanish and English texts as part of an underlying MT project.

Acknowledgments

The authors acknowledge support from the University of São Paulo (USP / EACH), CNPq and FAPESP (grant number 2006/03941-7).

References

1. Mitkov, R.: Anaphora Resolution. Longman, New york (2002)
2. Bick, E.: The parsing system PALAVRAS: automatic grammatical analysis of Portuguese in a constraint grammar framework. PhD Thesis, Arhus University (2000)

3. Müller, C., Strube, M.: MMAX: A tool for the annotation of multi-modal corpora. In: IJCAI 2001, Seattle, pp. 45–50 (2001)
4. Soon, W.M., et al.: A Machine Learning Approach to Correference Resolution of Noun Phrases. Computational Linguistics 27(4) (2001)
5. McCarthy, J.F., Lehnert, W.G.: Using Decision Trees for Coreference Resolution. In: 14th International Conference on Artificial Intelligence IJCAI (1995)
6. Ng, V., Cardie, C.: Improving Machine Learning Approaches to Coreference Resolution. In: 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, pp. 104–111 (2002)
7. Mitkov, R.: Multilingual Anaphora Resolution. Machine Translation 14(3-4), 281–299 (1999)
8. Oliveira, P.R., Romero, R.A.F.: Enhanced ICA Mixture Model for Unsupervised Classification. In: Lemaître, C., Reyes, C.A., González, J.A. (eds.) IBERAMIA 2004. LNCS (LNAI), vol. 3315, pp. 205–214. Springer, Heidelberg (2004)
9. Lee, T., Lewicki, M.S., Sejnowski, T.J.: ICA mixture models for unsupervised classification of non-Gaussian classes and automatic context switching in blind signal separation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(10), 1078–1089 (2000)

Semantic and Syntactic Features for Dutch Coreference Resolution

Iris Hendrickx[1], Veronique Hoste[2], and Walter Daelemans[1]

[1] CNTS - Language Technology Group,
University of Antwerp, prinsstraat 13, Antwerp
Belgium
iris.hendrickx@ua.ac.be, walter.daelemans@ua.ac.be
[2] LT3 - Language and Translation Technology Team,
University College Ghent, Groot-Brittaniëlaan 45, Ghent,
Belgium
veronique.hoste@hogent.be

Abstract. We investigate the effect of encoding additional semantic and syntactic information sources in a classification-based machine learning approach to the task of coreference resolution for Dutch. We experiment both with a memory-based learning approach and a maximum entropy modeling method.

As an alternative to using external lexical resources, such as the low-coverage Dutch EuroWordNet, we evaluate the effect of automatically generated semantic clusters as information source. We compare these clusters, which group together semantically similar nouns, to two semantic features based on EuroWordNet encoding synonym and hypernym relations between nouns.

The syntactic function of the anaphor and antecedent in the sentence can be an important clue for resolving coreferential relations. As baseline approach, we encode syntactic information as predicted by a memory-based shallow parser in a set of features. We contrast these shallow parse based features with features encoding richer syntactic information from a dependency parser. We show that using both the additional semantic information and syntactic information lead to small but significant performance improvement of our coreference resolution approach.

1 Introduction

Coreference resolution is the task of resolving different descriptions of the same underlying entity in a given text. Written and spoken texts contain a large number of coreferential relations and a good text understanding largely depends on the correct resolution of these relations. Resolving ambiguous referents in a text can be a helpful preprocessing step for many NLP applications such as text summarization or question answering.

As an alternative to the knowledge-based approaches, in which there has been an evolution from the systems which require an extensive amount of linguistic and non-linguistic information (e.g. [1]) toward more knowledge-poor approaches (e.g. [2]), machine learning approaches have become increasingly popular for this problem. Most of the machine learning approaches (e.g. [3], [4], [5]) are classification-based approaches which use a two-step procedure. This approach requires a corpus annotated with coreferential links between NPs. Next, instances are created between every NP (candidate anaphor) and all of its preceding NPs (candidate antecedents). The first step involves the classification of each pair of NPs as coreferential or not. In a second step, coreferential chains are built on the basis of the positively classified instances. In order to overcome this two-step procedure problem, others such as [6] recently proposed to use features over sets of noun phrases instead of features of pairs of noun phrases.

Most of the current machine learning approaches to coreference resolution use a combination of lexical, positional, syntactic and semantic information sources. Current systems can resolve part of the coreference relations using shallow features, but some cases need deeper linguistic or world knowledge to be resolved, such as for example the referring expressing **House** in the example below.

> **The US House of Representatives** has passed a bill which would fund military operations in Iraq to the end of July. Further funding would be dependent on events in Iraq meeting certain, as yet undefined, benchmarks of progress. President Bush has already vetoed one Iraq funding bill and said he opposed the new proposal, but did say that the idea of benchmarks "made sense". The move came as **the White House** and Democrats struck an accord on standards for bilateral free trade deals. The deal was announced by **House** Speaker Nancy Pelosi, a Democrat, who hailed it as a result of the Democratic triumph in last year's congressional elections.

In this study, we investigate the integration of two semantic sources and a syntactic information source for Dutch coreference resolution. Given the lack of broad-coverage lexical resources for Dutch, we investigate automatically generated semantic clusters [7] to model the semantic classes of NPs. We study the effect of using this information and we compare its effect to the use of two other semantic features based on the Dutch EuroWordNet [8]. Secondly, we investigate the effect of adding features extracted from full parsing in our coreference application for Dutch and we contrast this full-parsing based approach with a shallow parse based approach.

The remainder of this paper is structured as follows. Section 2 gives an overview of the related literature on this topic. Section 3 gives a general overview of the system architecture and in Section 4 and 5 we discuss the construction of the semantic and syntactic features. Section 6 describes the experimental setup, whereas results and conclusions are presented in Sections 7 and 8.

2 Related Work

In the last years, we can observe an increased interest in the use of semantic resources for coreference resolution. Especially WordNet [9] has been and remains a very useful information source for coreference resolution [10,11,12,13,14]. In the last years we observe an increased interest in the integration of additional semantic sources. [15], for example, code semantic information as semantic relations based on the ACE relation ontology relations such as 'membership' and show the beneficial effect on coreference resolution. [13] study the effect of three semantic sources, viz. WordNet, taxonomies extracted from Wikipedia and semantic role labeling and show that these semantic features improve their system. [16] and [17] explore several semantic information sources such as ACE semantic classes and a thesaurus expressing semantic similarity created by [18]. [19] investigate the extraction of automatically discovered patterns which express semantic relatedness information for coreference resolution.

If we consider the use of syntactic features in the existing machine learning systems, we can observe that many systems use some form of shallow syntactic features such as [4,14]. Some systems also look at deeper syntactic information sources. We will briefly describe three of them. [20] explore syntactic features extracted from dependency parse trees for English, Arabic and Chinese. Part of these features are inspired by the binding theory. They find significant improvements for English and Arabic but not for Chinese. [21] look at predicate-argument structure statistics but found no improvement for the task of pronoun resolution for English. [22] successfully explore the use of parse trees as a structural feature in a kernel-based method for pronoun resolution.

3 Architecture

The first phase of our supervised machine learning approach to coreference resolution is training a classifier on the annotated documents. We start with transforming the annotated documents into training instances. First, the raw texts are preprocessed to determine the noun phrases in the text and to produce information about these nouns. The following preprocessing steps were taken. First, tokenisation was done to split punctuation from adjoining words. For the recognition of names in the text, a memory-based named entity recognition approach [23] was used, which distinguishes between persons, organizations and locations. Part-of-speech tagging and text chunking was performed by the memory-based tagger MBT [24] trained on the Spoken Dutch Corpus (http://lands.let.ru.nl/cgn). Finally, grammatical relation finding was performed to determine grammatical relations between chunks, e.g. subject, object, etc. [25].

On the basis of the preprocessed texts, training instances are created. After the detection of the NPs by the text chunker, every NP is linked to its preceding NPs, with a restriction of 20 sentences backwards. A pair of NPs that belongs to the same coreferential chain, gets a positive label; all other pairs get a negative label. To limit the instance set size we restrict the search scope to 3 sentences for pronominal anaphors and for noun pairs which do not share the same head. For each pair, a feature vector is created to describe the NPs and their relation. These instances are the training set for the classifier.

A combination of different information sources can be used to predict coreferential relations between noun phrases. For our coreference resolution system, we used a combination of positional features (features indicating the number of sentences/NPs between the anaphor and its possible antecedent), morphological and lexical features (such as features which indicate whether a given anaphor, its candidate antecedent or both are pronouns, proper nouns, demonstrative or definite NPs), syntactic features which inform on the syntactic function of the anaphor and its candidate antecedent and check for syntactic parallelism, string-matching features which look for complete and partial matches and finally several semantic features. For the construction of these semantic features, we took into account lists with location names, male and female person names. Furthermore, we looked for female/male pronouns and for gender indicators such as 'Mr.', 'Mrs.' and 'Ms.'. One feature also looked at the named entity type (organization, person, location) of both NPs. Further information was also extracted from the Dutch EuroWordNet synonym and hypernym relations, which we will describe in the following section.

4 Semantic Information Sources

Semantic information can be an important clue to determine whether two referents point to the same entity. For Dutch there are few sources available to obtain semantic knowledge about words. One well-known source is the Dutch part of EuroWordNet [8], a multilingual lexical database. EuroWordNet has approximately 46K entries for Dutch nouns.

We use EuroWordNet to construct two binary features **is_synonym** and **is_hypernym**. These features code for every pair of referents whether their descriptions can be found in EuroWordNet in some synonym or hypernym relation[1]. In case of ambiguous words, we check for all senses of the word.

As a second source we use semantic clusters [7]. These clusters were extracted with unsupervised k-means clustering on the Twente Nieuws Corpus[2], a corpus containing Dutch news paper text. The corpus was first preprocessed by the Alpino parser [26] to extract syntactic relations. The top-10,000 lemmatized nouns including names were clustered into a 1000 groups based on the similarity of their syntactic relations. Table 1 shows four clusters extracted from the Twente Nieuws corpus. These clusters contain both common nouns and names.

For each pair of referents we construct three features as follows. For each referent the lemma of the head word is looked up in the list of clusters. We construct a binary feature marking whether the head words of the referents occur in the same cluster (**same_cluster**) and two features (**cluster1, cluster2**) presenting the cluster number of each referent or zero otherwise. The observation that a potential anaphor is member of a particular cluster may not be informative. However combinations of certain cluster numbers can be informative. For

[1] Two referents with complete string match are also considered as synonyms and hypernyms.

[2] Available from: http://wwwhome.cs.utwente.nl/∼druid/TwNC/TwNC-main.html

Table 1. Four semantic clusters extracted with unsupervised k-means clustering. The first column of numbers presents the names of the clusters.

201	{barrière belemmering drempel hindernis hobbel horde knelpunt obstakel struikelblok} (English: barrier impediment threshold hindrance bump hurdle bottleneck obstacle block)
223	{biertje borrel cocktail cola drankje glaasje kopje pilsje} (English: beer booze cocktail cola drink glass cup brew)
320	{Andreotti Berlusconi Bildt Carl_Bildt Craxi Gajdar Jegor_Gajdar Lubbers Martens Margaret_Thatcher Ruud_Lubbers Silvio_Berlusconi Thatcher}
395	{ambtgenoot collega expremier leider minister minister-president opvolger oud-premier partijgenoot premier president vice-premier} (English: fellow colleague ex-premier leader minister Prime_Minister former_premier political_associate premier president vice-president)

example an anaphor "minister-president" is member of cluster 320 in Table 1. A potential antecedent "Margaret Thatcher" is a member of cluster 395. The combination of these two feature values can give a strong clue for a coreferential relation.

To get an insight in the impact of these semantic features, we calculated the percentages of instances in which a particular semantic feature has a non-zero value, shown in Table 2. Only 3.4% of the instances describes a coreferential relation. We computed the percentages on the full set of instances and on the small subset of positive instances[3]. Looking at the full instance set in the first column of the table, the WordNet features are only active in 2% of the instances. But looking at the subset of positive instances, the percentages increase to 36%. This increase implies a clear correlation between the positive class and the active WordNet features.

We also observe an increase for the same_cluster feature. The cluster1 or cluster2 feature are active in 60% of the instances of the full set. On the positive class subset, the percentages drop to 35-37%. This can be explained by the fact that the percentage of pronouns is relatively higher in the subset of positive instances, and pronouns get a zero as cluster value. We also measured to what extent the WordNet feature and the same_cluster feature overlap. In the full instance set 41% of the instances for which the same_cluster is active, has also a positive is_synonym feature. This low percentage of overlap confirm that the two semantic sources cover different parts of the instance space.

5 Syntactic Information

Another important clue for resolving coreferential relations is the syntactic function of the anaphor and antecedent in the sentence. We code syntactic information as predicted by the memory-based shallow parser in our feature set as

[3] Computed at 90% training part of our data set containing 327,728 instances, and 11,062 positive instances.

Table 2. Percentage of instances in which each semantic feature is active, computed at both the full set of instances and the small subset of the positive class instances

feature	% inst	% positive inst
is_synonym	2.2	36.4
is_hypernym	2.3	36.1
cluster1	60.1	35.0
cluster2	59.0	37.2
same_cluster	2.3	17.6

described in Section 3. We investigate whether the richer syntactic information of a full parser would be a helpful information source for our task. We use the Alpino parser [26], an automatic broad-coverage dependency parser for Dutch to generate the following 11 additional features:

Named Entity label as produced by the Alpino parser, one for the anaphor and one for the antecedent.

Number agreement between the anaphor and antecedent, presented as a four valued feature (values: *sg, pl, both, measurable_nouns*).

Dependency labels as predicted for (the head word of) the anaphor and for the antecedent.

Same dependency label the case that both anaphor and antecedent have the same dependency label is coded as a binary feature.

Dependency path between the governing verb and the anaphor, and between the verb and antecedent.

Clause information is coded as two binary features, is the anaphor / antecedent part of the main clause or not.

Root overlap binary feature that codes overlap between 'roots' or lemmas of the anaphor and antecedent. In the Alpino parser, the root of a noun phrase is the form without inflections. Special cases are compounds and names. Compounds are split and we use the last element in the comparison. For names we take the complete strings.

Next we give an example of these features. The sentence in Example 1 contains a coreferential link between the anaphor "het bedrijf" (the company) and the name "Ford Genk". We list the features as predicted by Alpino. An obvious error is the named entity label of the antecedent, which should have been labeled as 'organization'.

Example 1.
Algemeen directeur Jan Gijsen van Ford Genk maakt bekend dat het bedrijf de volgende twee jaar 1400 banen wil schrappen.
(*English: Head director Jan Gijsen of Ford Genk announces that the company will cut 1400 jobs in the next two years.*)

1. named entity label anaphor: noun
2. named entity label antecedent: person-male

3. number agreement: both (anaphor is singular, antecedent labeled as both)
4. dependency label anaphor: subject
5. dependency label antecedent: object1
6. label match: no
7. dependency path anaphor: [[schrap,hd/su],[wil,hd/su]]
8. dependency path antecedent: [[maak_bekend,hd/su,directeur,hd/mod,van, hd/obj1]]
9. clause anaphor: not in main clause
10. clause antecedent: is in main clause
11. root overlap: no

6 Experimental Setup

We use a Dutch corpus of Flemish news articles, KNACK-2002, annotated with coreference information for NPs [27]. In a first experiment we evaluate the effect of the two semantic sources described in Section 4. We run four experiments with the feature set combinations with and without the WordNet- or cluster-based features. The feature set size varies from 42 features (without WordNet- and cluster-based features) to 47 (with both types of features).

We compare two different machine learning algorithms; memory-based learning [28] and maximum entropy modeling [29]. We use the Timbl software package [30] as our implementation of memory-based learning. For maximum entropy modeling we use the implementation Maxent [31].

In a second experiment we add the features extracted from the Alpino parser output described in Section 5 to the full feature set of 47 features including both types of semantic sources. As the information in these features may largely overlap with the information already presented in the features produced by the memory-based shallow parser, we decided to use genetic algorithms to automatically select an optimal feature selection. Genetic algorithms (GA) have been proposed [32] as an useful method to find an optimal setting in the enormous search space of possible parameter and feature set combinations. We run experiments with a generational genetic algorithm for feature set and algorithm parameter selection of Timbl with 30 generations and a population size of 10. As a comparison we run the GA for both the instance set with vectors of 47 features and for the set with 59 features.

The standard approach to evaluate a coreference resolution system is to compare the predictions of the system to a hand-annotated gold standard test set in cross-validation experiments. The performance of the system can be measured at two levels. One can evaluate the performance of the classifier and determine how well it predicted the presence of a coreference relation for a pair of NPs. In this case, we measure the precision, recall and F-score of the labeled positive NP pairs. We will denote this as evaluation at the instance level. One can also evaluate the construction of the complete coreference chains which can be measured with the MUC scoring software from Vilain et al. [33].

In each experiment we use ten-fold cross validation on 242 documents of KNACK-2002 with both Timbl and Maxent. The GA optimization is done for Timbl and not for Maxent. Timbl is more sensitive to feature redundancy than Maxent as Maxent performs feature weighting internally. The GA is run on the first fold of the ten fold, as running the GA is rather time-consuming. The found optimal setting was also used for the other folds. We also compute a baseline score for the evaluation of the complete coreference chains. The baseline assigns each NP in the test set its most nearby NP as antecedent.

7 Results

The results of the evaluation of the effect of the semantic information sources are shown in Table 3 and 4. Each column presents the results of one of the feature set variations with and without the WordNet features or the cluster-based features. Table 3 presents the micro-averaged F-scores measured at the instance level for Timbl and Maxent. For Timbl adding the WordNet features does not really show any effect, while adding the cluster-based features does show a small improvement. For Maxent adding the WordNet features or the cluster-based features separately gives a small drop in performance. Combining both features has a stronger effect and improves the F-score of Maxent with 1%. The MUC scores presented in Table 4 show the same trends.

Table 3. Micro-averaged F-score computed in 10-fold cross validation experiments for Timbl and Maxent with various feature set variations measured at the instance level

	−WordNet	+WordNet	−WordNet	+WordNet
	−cluster	−cluster	+cluster	+cluster
Timbl	46.45	46.43	47.11	47.45
Maxent	49.20	48.71	48.77	49.94

Table 4. Average MUC F-scores computed in 10-fold cross validation experiments for Timbl and Maxent with various feature set variations

	−WordNet	+WordNet	−WordNet	+WordNet
	−cluster	−cluster	+cluster	+cluster
Timbl	44.6	44.6	45.6	45.6
Maxent	45.9	45.5	45.7	46.7

The results of our second experiment in which we evaluate the effect of adding features derived from the output of a dependency parser are shown in Table 5 (F-scores at the instance level) and 6 (MUC scores at the chain level).[4]

[4] Note that the F-score of Maxent with 47 features shown in the third row of Table 5 is a repetition of F-score the last cell of Table 3.

A first observation is the improvement given by the GA optimization for Timbl. Timbl with 47 features and default algorithmic parameters setting reaches a F-score of 47.45% (Table 3), with optimized settings the F-score of Timbl improves to 54.8% (Table 5).

The differences in F-score at the instance level are small as shown in Table 5. When we look at the score computed at the chain level, we see an improvement of 3% in F-score for Timbl and 1% for Maxent. For Timbl adding the additional features improves the recall at the cost of precision. For Maxent on the other hand both precision and recall are improved by adding the extra features.

Table 5. Micro-averaged F-score and accuracy computed in 10 fold cross validation experiments. Timbl is run with the settings as selected by the genetic algorithm, Maxent with all features.

	recall	precision	F-score	accuracy
TIMBL, GA, 47 features	44.8	70.5	54.8	97.6
TIMBL, GA, 59 features	48.4	64.1	55.1	97.4
MAXENT, 47 features	39.9	66.6	49.9	97.4
MAXENT, 59 features	40.0	68.6	50.5	97.4

Table 6. MUC-scores computed in 10 fold cross validation experiments. Timbl is run with the settings as selected by the genetic algorithm, Maxent with all features.

	recall	precision	F-score
baseline	81.1	24.0	37.0
TIMBL, GA, 47 features	36.8	70.2	48.2
TIMBL, GA, 59 features	44.0	61.4	51.3
MAXENT, 47 features	35.7	67.2	46.7
MAXENT, 59 features	36.8	68.0	47.6

8 Conclusions

We have shown that both the semantic sources and the syntactic information are useful features for our coreference resolution module. We tested these information sources with two different classifiers, memory-based learning and maximum entropy modeling. We evaluated the effect of two types of semantic information sources, namely information extracted from WordNet and information extracted from unsupervised learned semantic clusters. Our experiments showed that for Maxent, adding one semantic source can slightly decrease the performance. However, combining the WordNet- and cluster-based features gives a small positive effect for both classifiers. In a second experiment we added features derived from a dependency parser to the feature set. The effect of these additional features is marginal when measured at the instance level, but we do see a small improvement when we evaluate on complete coreference chains.

Acknowledgments

This research is funded by the Dutch-Flemish NTU/STEVIN programme (url: http://taalunieversum.org/stevin). We would like to thank Tim Van de Cruys for sharing his data sets of semantic clusters and Gosse Bouma for running the Alpino parser.

References

1. Rich, E., LuperFoy, S.: An architecture for anaphora resolution. In: Proceedings of the Second Conference on Applied Natural Language Processing, pp. 18–24 (1988)
2. Mitkov, R.: Robust pronoun resolution with limited knowledge. In: Proceedings of the 17th International Conference on Computational Linguistics (COLING-1998/ACL-1998), pp. 869–875 (1998)
3. McCarthy, J.: A Trainable Approach to Coreference Resolution for Information Extraction. PhD thesis, Department of Computer Science, University of Massachusetts, Amherst MA (1996)
4. Soon, W., Ng, H., Lim, D.: A machine learning approach to coreference resolution of noun phrases. Computational Linguistics 27(4), 521–544 (2001)
5. Ng, V., Cardie, C.: Combining sample selection and error-driven pruning for machine learning of coreference rules. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002), pp. 55–62 (2002)
6. Culotta, A., et al.: First-order probabilistic models for coreference resolution. In: Proceedings of HLT/NAACL, pp. 81–88 (2007)
7. Van de Cruys, T.: Semantic clustering in dutch. In: Proceedings of the Sixteenth Computational Linguistics in the Netherlands (CLIN), pp. 17–32 (2005)
8. Vossen, P. (ed.): EuroWordNet: a multilingual database with lexical semantic networks. Kluwer Academic Publishers, Norwell (1998)
9. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
10. Poesio, M., et al.: Learning to resolve bridging references. In: Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL 2004), pp. 143–150 (2004)
11. Harabagiu, S., Bunescu, R., Maiorano, S.: Text and knowledge mining for coreference resolution. In: Proceedings of the 2nd Meeting of the North American Chapter of the Association of Computational Linguistics (NAACL-2001), pp. 55–62 (2001)
12. Markert, K., Nissim, M.: Comparing knowledge sources for nominal anaphora resolution. Computational Linguistics 31(3), 367–401 (2005)
13. Ponzetto, S.P., Strube, M.: Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In: Proceedings of the Human Language Technology Conference of the NAACL, Main Conference, pp. 192–199 (2006)
14. Ng, V., Cardie, C.: Improving machine learning approaches to coreference resolution. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002), pp. 104–111 (2002)
15. Ji, H., Westbrook, D., Grishman, R.: Using semantic relations to refine coreference decisions. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp. 17–24 (2005)
16. Ng, V.: Semantic class induction and coreference resolution. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Association for Computational Linguistics, pp. 536–543 (2007)

17. Ng, V.: Shallow semantics for coreference resolution. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-2007), pp. 1689–1694 (2007)
18. Lin, D.: Automatic retrieval and clustering of similar words. In: COLING-ACL, pp. 768–774 (1998)
19. Yang, X., Su, J.: Coreference resolution using semantic relatedness information from automatically discovered patterns. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pp. 528–535 (2007)
20. Luo, X., Zitouni, I.: Multi-lingual coreference resolution with syntactic features. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp. 660–667 (2005)
21. Kehler, A., et al.: The (non)utility of predicate-argument frequencies for pronoun interpretation. In: Proceedings of HLT-NAACL, pp. 289–296 (2004)
22. Yang, X., Su, J., Tan, C.L.: Kernel-based pronoun resolution with structured syntactic knowledge. In: Proceedings of the 21st International Conference on Computational Linguistics, pp. 41–48 (2006)
23. Tjong Kim Sang, E.: Memory-based named entity recognition. In: Proceedings of CoNLL-2002, Taipei, Taiwan, pp. 203–206 (2002)
24. Daelemans, W., et al.: Memory based tagger, version 2.0, reference guide. Technical Report ILK Technical Report - ILK 03-13, Tilburg University (2003)
25. Tjong Kim Sang, E., Daelemans, W., Höthker, A.: Reduction of dutch sentences for automatic subtitling. In: Computational Linguistics in the Netherlands 2003, Selected Papers from the Fourteenth CLIN Meeting, pp. 109–123 (2004)
26. Bouma, G., van Noord, G., Malouf, R.: Alpino: Wide-coverage computational analysis of dutch. In: Computational Linguistics in The Netherlands 2000 (2001)
27. Hoste, V., de Pauw, G.: Knack-2002: a richly annotated corpus of dutch written text. In: The fifth international conference on Language Resources and Evaluation (LREC) (2006)
28. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. Institute of Electrical and Electronics Engineers Transactions on Information Theory 13, 21–27 (1967)
29. Berger, A., Della Pietra, S., Della Pietra, V.: Maximum Entropy Approach to Natural Language Processing. Computational linguistics 22(1) (1996)
30. Daelemans, W., et al.: TiMBL: Tilburg Memory Based Learner, version 5.1, reference manual. Technical Report ILK-0402, ILK, Tilburg University (2004)
31. Le, Z.: Maximum Entropy Modeling Toolkit for Python and C++. Natural Language Processing Lab, Northeastern University, China (2004)
32. Daelemans, W., Hoste, V., De Meulder, F., Naudts, B.: Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. In: Lavrač, N., et al. (eds.) ECML 2003. LNCS (LNAI), vol. 2837, pp. 84–95. Springer, Heidelberg (2003)
33. Vilain, M., et al.: A model-theoretic coreference scoring scheme. In: Proceedings of the Sixth Message Understanding Conference (MUC-6), pp. 45–52 (1995)

Stat-XFER: A General Search-Based Syntax-Driven Framework for Machine Translation

Alon Lavie

Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
alavie@cs.cmu.edu.edu

Abstract. The CMU Statistical Transfer Framework (Stat-XFER) is a general framework for developing search-based syntax-driven machine translation (MT) systems. The framework consists of an underlying syntax-based transfer formalism along with a collection of software components designed to facilitate the development of a broad range of MT research systems. The main components are a general language-independent runtime transfer engine and decoder, along with several different tools for creating the various underlying language-pair-specific resources that are required for building a specific MT system for any given language pair. We describe the general framework, its unique properties and features, and its application to the construction of MT research prototype systems for a diverse collection of language pairs.

1 Introduction

The field of Machine Translation (MT) has dramatically shifted in the course of the past decade. Modern state-of-the-art approaches to MT rely on machine learning methods of increasing complexity and sophistication in order to automatically acquire their underlying translation models from available data resources. Phrase-based Statistical MT (PB-SMT) [1,2,3] has become the predominant approach in recent years. In PB-SMT, simple statistical modeling methods are used to acquire likely phrase-to-phrase translation equivalents from large volumes of sentence-parallel text corpora. In the absence of large sentence-parallel data, the statistical estimation methods break down, and the approach becomes ineffective. Vast sentence-parallel corpora exist only for a limited number of language pairs (primarily pairs of European languages, Chinese, Japanese and Arabic), severely limiting the applicability of this approach. While the amount of online resources for many languages will undoubtedly grow over time, many of the languages spoken by smaller ethnic groups and populations in the world will not have such resources within the foreseeable future. Corpus-based MT approaches will therefore not be effective for such languages for some time to come.

Furthermore, even for language pairs with large amounts of sentence-parallel data such as Chinese-English, the phrase-based models are often too simple and naive for capturing many complex divergences between the languages. There has been increasing recognition in the MT research community in recent years that high-quality fully-automatic MT will require learning translation models that can capture advanced syntax and semantic representations and how they correspond across languages. Automatically acquired syntax-based models for MT have started to receive increasing attention in the last few years [4,3,5].

Over the past six years, the Avenue MT research group at Carnegie Mellon, under DARPA and NSF funding, has been developing a new MT framework that is designed to address many of the above challenges. The framework is inspired by many of the ideas of modern statistical MT. Most prominently, it is founded on the basic notion of search-based "decoding". The framework consists of an underlying syntax-based transfer formalism, a general, language-independent translation engine, and a collection of software components designed to facilitate the acquisition of the underlying language resources required for development of an MT system for any specific language pair. These resource acquisition tools target different scenarios, ranging from low-resource to high-resource availability, and support the development of a broad range of MT research systems. The framework has been designed to be able to handle large-scale broad-coverage lexical resources and transfer grammars. The acquisition of these resources can be done in diverse and creative ways, effectively combining automatic acquisition from data with human knowledge. We refer to this framework using the name "*statistical transfer*", or in short, Stat-XFER.

The Stat-XFER framework was originally developed to support rapid MT prototype development for translation between low-resource source languages (such as Hebrew) and high resource target languages (such as English). Over the past year, the Stat-XFER framework has been greatly extended to also support effective automatic acquisition of translation resources from vast parallel corpora. The focus of this paper, however, is mostly on scenarios involving low-resource source languages. We describe the general framework, its unique properties and features, and its application to the construction of MT research prototype systems for a diverse collection of language pairs. We use our Hebrew-to-English MT prototype system developed under the Stat-XFER framework to highlight many of the important aspects of the system.

2 The Stat-XFER Framework

The Stat-XFER framework uses a declarative formalism for symbolic transfer grammars. A grammar consists of a collection of *synchronous context-free* rules, which can be augmented by unification-style feature constraints. These transfer rules specify how phrase structures in a source-language correspond and transfer to phrase structures in a target language, and the constraints under which these rules should apply. The framework also includes a fully-implemented transfer engine that applies the transfer grammar to a source-language input sentence

```
{NP1,2}
;;SL: $MLH ADWMH
;;TL: A RED DRESS
;;Score:2
NP1::NP1 [NP1 ADJ] -> [ADJ NP1]
(
 (X2::Y1)
 (X1::Y2)
 ((X1 def) = -)
 ((X1 status) =c absolute)
 ((X1 num) = (X2 num))
 ((X1 gen) = (X2 gen))
 (X0 = X1)
)
```

```
{NP1,3}
;;SL: H $MLWT H ADWMWT
;;TL: THE RED DRESSES
;;Score:4
NP1::NP1 [NP1 "H" ADJ] -> [ADJ NP1]
(
 (X3::Y1)
 (X1::Y2)
 ((X1 def) = +)
 ((X1 status) =c absolute)
 ((X1 num) = (X3 num))
 ((X1 gen) = (X3 gen))
 (X0 = X1)
 )
```

Fig. 1. NP Transfer Rules for Nouns Modified by Adjectives from Hebrew to English

at runtime, and produces collections of scored word and phrase-level translations according to the grammar. Scores are based on a log-linear combination of several features, and a beam-search controls the underlying parsing and transfer process. The framework was designed to support research on a variety of methods for automatically acquiring transfer grammars from limited amounts of elicited word-aligned data. The framework also supports manual development of transfer grammars by experts familiar with the two languages.

The Stat-XFER framework has been applied to building research prototype MT systems for quite a number of language pairs over the past five years. The most developed prototype systems to date are our Hebrew-to-English and Chinese-to-English systems. The Hebrew system is described in detail in later sections of this paper. The Chinese system has been under development for the past year, and is being used as one of several engines for Chinese-to-English translation within the IBM-led Rosetta team as part of the DARPA/GALE program. Other integrated Stat-XFER prototypes include a Hindi-to-English system developed under the DARPA/TIDES "Surprise Language Exercise" in June-2003 [6] [7], and preliminary systems for German-to-English, Dutch-to-English and French-to-English. We have also been applying the approach to several native languages in North and South America, starting with a Mapudungun-to-Spanish system[1]. A prototype system for Inupiaq-to-English[2] is in initial stages of development. We are currently also collaborating with research groups in Brazil and in Turkey on developing MT prototypes for Portuguese-to-English and Turkish-to-English.

2.1 The Transfer Formalism

The design of the transfer rule formalism itself was guided by the consideration that the rules must be simple enough to be learned by an automatic process, but also powerful enough to allow manually-crafted rule additions and changes to improve the automatically learned rules. To illustrate the rule formalism, we

[1] Mapudungun is a native language of southern Chile.
[2] Inupiaq is a native language of northern Alaska.

show two transfer rules for structurally transferring nouns modified by adjectives from Hebrew to English, depicted in Figure 1.

The following list summarizes the components of a transfer rule. In general, the x-side of a transfer rules refers to the source language (SL), whereas the y-side refers to the target language (TL).

- **Type information:** This identifies the type of the transfer rule and in most cases corresponds to a syntactic constituent type. Sentence rules are of type `S`, noun phrase rules of type `NP`, etc. The formalism also allows for SL and TL type information to be different.
- **Part-of speech/constituent information:** For both SL and TL, we list a linear sequence of components that constitute an instance of the rule type. These can be viewed as the 'right-hand sides' of context-free grammar rules for both source and target language grammars. The elements of the list can be lexical categories, lexical items, and/or phrasal categories.
- **Alignments:** Explicit annotations in the rule describe how the set of source language components in the rule align and transfer to the set of target language components. Zero alignments and many-to-many alignments are allowed.
- **X-side constraints:** The x-side constraints provide information about features and their values in the source language sentence. These constraints are used at run-time to determine whether a transfer rule applies to a given input sentence.
- **Y-side constraints:** The y-side constraints are similar in concept to the x-side constraints, but they pertain to the target language. At run-time, y-side constraints serve to guide and constrain the generation of the target language sentence.
- **XY-constraints:** The xy-constraints provide information about which feature values transfer from the source into the target language. Specific TL words can obtain feature values from the source language sentence.

2.2 Runtime System Architecture

To describe the runtime archirecture of the Stat-XFER framework, we use our integrated Hebrew-to-English prototype for illustrative purposes. The core components, consisting of the *transfer engine* and the *decoder*, however, are language independent. The system consists of the following main components: a Hebrew input sentence is pre-processed, and then sent to a *morphological analyzer*, which produces all possible analyses for each input word, represented in the form of a lattice of possible input word lexemes and their morphological features. The input lattice is then passed on to the *transfer engine*, which applies a collection of lexical and structural *transfer rules* in order to parse, transfer and generate English translations for all possible word and phrase segments of the input. Each possible translation segment is scored by a combination of various features. The collection of translation segments is stored in an output lattice data-structure.

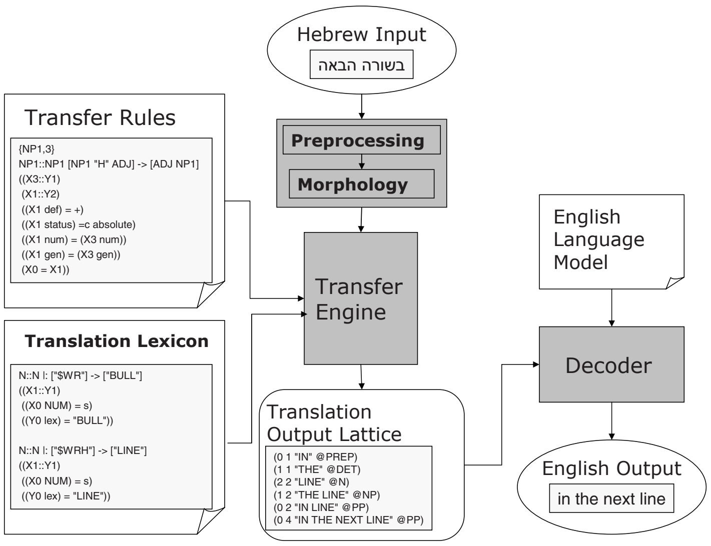


Fig. 2. Architecture of the Hebrew-to-English Transfer-based MT System

The transfer engine uses a beam-search to control the number of possible translation segments explored. The lexical transfer rules used by the transfer engine are derived from a *bilingual lexicon*, while the higher-level structural transfer rules come from either a manually-developed or automatically-acquired transfer grammar. In the final stage, the English lattice is fed into a *decoder* which uses a log-linear combination of several features to search and select a combination of sequential translation segments that together represent the best scoring translation of the entire input sentence. A schematic diagram of the system architecture can be seen in Figure 2.

2.3 The Transfer Engine

The transfer engine is the module responsible for applying the comprehensive set of lexical and structural transfer rules, specified by the translation lexicon and the transfer grammar (respectively), to the source-language (SL) input lattice, producing a comprehensive collection of target-language (TL) output segments. The output of the transfer engine is a lattice of alternative translation segments. The alternatives arise from syntactic ambiguity, lexical ambiguity, and multiple synonymous choices for lexical items in the translation lexicon.

The transfer engine incorporates the three main processes involved in transfer-based MT: parsing of the SL input, transfer of the parsed constituents of the SL to their corresponding structured constituents on the TL side, and generation of the TL output. All three of these processes are performed based on the transfer grammar – the comprehensive set of transfer rules that are loaded into the transfer engine at runtime. Parsing, transfer and generation are fully integrated into an interleaved bottom-up "parse-and-transfer" algorithm, which is essentially an extended Chart Parser. Parsing is performed based solely on the source-language side of the transfer rules. A chart is populated with all constituent structures that were created in the course of parsing the SL input with the source-side portion of the transfer grammar. A parallel TL chart is populated in lock-step, containing the translations created by transfering the source-side constituents as specified in the transfer rules. The bottom-up process is initialized by populating the TL chart with the lexical translations of all source words, based on all available lexical transfer rules. TL lexical generation, driven by a TL morphology engine, can also be applied at this initial stage. The TL chart maintains "stacks" of scored translation options for all substrings of the SL input. As parsing progresses, whenever a new source-side constituent is created, the transfer "instructions" of the completed rule are applied, thus creating the possible translations that correspond to the SL constituent. The set of translations is then added to the appropriate "stack" within the TL chart. Feature constraints contained within the rules are also applied in an integral interleaved fashion. "X-side" constraints are applied whenever a source-side constituent is completed. "X-Y" constraints and "Y-side" constraints are applied when performing transfer. Constraints do not generate additional translation alternatives. They can block rules from applying, or "weed out" possible translations created by any rule application. Finally, the set of generated TL output strings that corresponds to the collection of all TL chart entries is collected into a TL lattice, which is then passed on for decoding. The transfer engine was designed to support both manually-developed structural transfer grammars and grammars that can be automatically acquired from bilingual data. A more detailed description of the transfer engine can be found in [8].

2.4 Decoding

In the final stage, a monotonic decoder is used in order to create complete translation hypotheses from the lattice created during the transfer stage. The translation units in the lattice are organized according to the positional start and end indices of the input fragment to which they correspond. The lattice typically contains translation units of various sizes for different contiguous fragments of input. These translation units often overlap. The lattice also includes multiple word-to-word (or word-to-phrase) translations, reflecting the ambiguity in selection of individual word translations.

The task of the decoder is to select a linear sequence of adjoining but non-overlapping translation units that maximizes the overall score of the target language string given the source language string. The decoder uses a log-linear scoring model that combines scores from several different features. The current set of features include a *language model* of English, a score derived from the rule probabilities, two lexical probability scores (for "target given source" and "source given target"), a measure that reflects the number of translation fragments being combined and a feature that reflects the source-to-target relative sentence length. For language modeling, we use the Suffix Array Toolkit (SALM) developed at CMU [9]. The framework also supports using the SRI language modeling toolkit. The decoder is monotonic in the sense that it cannot reorder any translation units from the lattice.

3 The Hebrew-to-English Stat-XFER System

Machine translation of Hebrew is challenging due to two main reasons: the high lexical and morphological ambiguity of Hebrew and its orthography, and the paucity of available resources for the language. We developed a first, fully functional, version of the Hebrew-to-English Stat-XFER system [10] over the course of a two-month period with a total labor-effort equivalent to about four person-months of development. To the best of our knowledge, our system is the first broad-domain machine translation system for Hebrew. We used existing, publicly available resources which we adapted in novel ways for the MT task, and directly addressed the major issues of lexical, morphological and orthographical ambiguity.

3.1 The Hebrew Language

Modern Israeli Hebrew, henceforth *Hebrew*, exhibits clear Semitic behavior. In particular, its lexicon, word formation and inflectional morphology are typically Semitic. The major word formation machinery is root-and-pattern, where roots are sequences of three (typically) or more consonants and patterns are sequences of vowels and, sometimes, also consonants, with "slots" into which the root's consonants are inserted. Inflectional morphology is highly productive and consists mostly of suffixes, but also prefixes and circumfixes.

The Hebrew script,[3] not unlike the Arabic one, attaches several short particles to the word which immediately follows them. These include, *inter alia*, the definite article *H* ("the"), prepositions such as *B* ("in"), *K* ("as"), *L* ("to") and *M* ("from"), subordinating conjunctions such as *$* ("that") and *K$* ("when"), relativizers such as *$* ("that") and the coordinating conjunction *W* ("and"). The script is rather ambiguous as the prefix particles can often also be parts of the stem. Thus, a form such as *MHGR* can be read as a lexeme "immigrant", as *M-HGR* "from Hagar" or even as *M-H-GR* "from the foreigner". Note that there is no deterministic way to tell whether the first *m* of the form is part of the pattern, the root or a prefixing particle (the preposition *M* ("from")).

An added complexity arises from the fact that there exist two main standards for the Hebrew script: one in which vocalization diacritics, known as *niqqud* "dots", decorate the words, and another in which the dots are omitted, but where other characters represent some, but not all of the vowels. Most of the modern printed and electronic texts in Hebrew use the "undotted" script. While a standard convention for this script officially exists, it is not strictly adhered to, even by the major newspapers and in government publications. Thus, the same

[3] To facilitate readability we use a transliteration of Hebrew using ASCII characters in this paper.

word can be written in more than one way, sometimes even within the same document. This fact adds significantly to the degree of ambiguity, and requires creative solutions for practical Hebrew language processing applications.

The challenge involved in constructing an MT system for Hebrew is amplified by the poverty of existing resources [11]. The collection of corpora for Hebrew is still in early stages [12] and all existing significant corpora are monolingual. Hence the use of aligned bilingual corpora for MT purposes is currently not a viable option. There is no available large Hebrew language model which could help in disambiguation. No publicly available bilingual dictionaries currently exist, and no grammar is available from which transfer rules can be extracted. Still, we made full use of existing resources which we adapted and augmented to fit our needs.

3.2 Hebrew Input Pre-processing

Our system is currently designed to process Hebrew input represented in UTF-8, but can also handle Microsoft Windows encoding. The morphological analyzer we use (see next sub-section) was designed, however, to produce Hebrew in a romanized (ASCII) representation. We adopted this romanized form for all internal processing within our system, including the encoding of Hebrew in the lexicon and in the transfer rules. The same romanized transliteration is used for Hebrew throughout this paper. The main task of our pre-processing module is therefore to map the encoding of the Hebrew input to its romanized equivalent. This should allow us to easily support other encodings of Hebrew input in the future. The pre-processing also includes simple treatment of punctuation and special characters.

3.3 Morphological Analysis

We use a publicly available morphological analyzer which is distributed through the Knowledge Center for Processing Hebrew. It is based on the morphological grammar of [13], but is re-implemented in Java so that it is faster and more portable [14]. The analyzer produces all the possible analyses of each input word. Analyses include the lexeme and a list of morpho-syntactic features such as number, gender, person, tense, etc. The analyzer also identifies prefix particles which are attached to the word. Our experiments with development data indicate that, at least for newspaper texts, the overall coverage of the analyzer is in fact quite reasonable. The texts we have used so far do not exhibit large amounts of vowel spelling variation, but we have not quantified the magnitude of the problem very precisely.

While the set of possible analyses for each input word comes directly from the analyzer, we developed a novel representation for this set to support its efficient processing through our translation system. The main issue addressed is that the analyzer may split an input word into a sequence of several output lexemes, by separating prefix and suffix lexemes. Moreover, different analyses of the same input word may result in a different number of output lexemes. We deal with

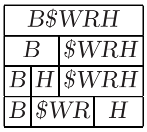


Fig. 3. Lattice Representation of a set of Analyses for the Hebrew Word *B$WRH*

```
Y0: ((SPANSTART 0)          Y1: ((SPANSTART 0)      Y2: ((SPANSTART 1)
     (SPANEND 4)                 (SPANEND 2)             (SPANEND 3)
     (LEX B$WRH)                 (LEX B)                 (LEX $WR)
     (POS N)                     (POS PREP))             (POS N)
     (GEN F)                                             (GEN M)
     (NUM S)                                             (NUM S)
     (STATUS ABSOLUTE))                                  (STATUS ABSOLUTE))

Y3: ((SPANSTART 3)          Y4: ((SPANSTART 0)      Y5: ((SPANSTART 1)
     (SPANEND 4)                 (SPANEND 1)             (SPANEND 2)
     (LEX $LH)                   (LEX B)                 (LEX H)
     (POS POSS))                 (POS PREP))             (POS DET))

Y6: ((SPANSTART 2)          Y7: ((SPANSTART 0)
     (SPANEND 4)                 (SPANEND 4)
     (LEX $WRH)                  (LEX B$WRH)
     (POS N)                     (POS LEX))
     (GEN F)
     (NUM S)
     (STATUS ABSOLUTE))
```

Fig. 4. Feature-Structure Representation of a set of Analyses for the Hebrew Word *B$WRH*

this issue by converting our set of word analyses into a lattice that represents the various sequences of possible lexemes for the word. Each of the lexemes is associated with a feature structure which encodes the relevant morpho-syntactic features that were returned by the analyzer.

As an example, consider the word form *B$WRH*, which can be analyzed in at least four ways: the noun *B$WRH* ("gospel"); the noun *$WRH* ("line"), prefixed by the preposition *B* ("in"); the same noun, prefixed by the same preposition and a hidden definite article (merged with the preposition); and the noun *$WR* ("bull"), with the preposition *B* as a prefix and an attached pronominal possessive clitic, *H* ("her"), as a suffix. Such a form would yield four different sequences of lexeme tokens which will all be stored in the lattice. To overcome the limited lexicon, and in particular the lack of proper nouns, we also consider each word form in the input as an unknown word and add it to the lattice with no features. This facilitates support of proper nouns through the translation dictionary. Figure 3 graphically depicts the lattice representation of the various analyses, and Figure 4 shows the feature-structure representation of the same analyses.

While two modules for morphological disambiguation of the output of the analyzer are currently being developed [15,16], their reliability is limited. We prefer to store all the possible analyses of the input in the lattice rather than disambiguate, since our transfer engine can cope with a high degree of ambiguity, and information accumulated in the translation process can assist in ambiguity resolution later on, during the decoding stage. A ranking of the different analyses of each word could, however, be very useful. For example, the Hebrew word form *AT* can be either the (highly frequent) definite accusative marker, the (less frequent) second person feminine personal pronoun or the (extremely rare) noun "spade". We currently give all these readings the same weight, although we intend to rank them in the future.

3.4 Word Translation Lexicon

The bilingual word translation lexicon was constructed based on the Dahan dictionary [17], whose main benefit is that we were able to obtain it in a machine readable form. This is a relatively low-quality, low-coverage dictionary. To extend its coverage, we use both the Hebrew-English section of the dictionary and the inverse of the English-Hebrew section. The combined lexicon was enhanced with a small manual lexicon of about 100 entries, containing some inflected forms not covered by the morphological analyzer and common multi-word phrases, whose translations are non-compositional.

Significant work was required to ensure spelling variant compatibility between the lexicon and the other resources in our system. The original Dahan dictionary uses the dotted Hebrew spelling representation. We developed scripts for automatically mapping the original forms in the dictionary into romanized forms consistent with the undotted spelling representation. These handle most, but not all of the mismatches. Due to the low quality of the dictionary, a fair number of entries require some manual editing. This primarily involves removing incorrect or awkward translations, and adding common missing translations. Due to the very rapid system development time, most of the editing done so far was based on a small set of development sentences. Undoubtedly, the dictionary is one of the main bottlenecks of our system and a better dictionary will improve the results significantly. The final resulting translation lexicon is automatically converted into the lexical transfer rule format expected by our transfer engine. A small number of lexical rules (currently 20), which require a richer set of unification feature constraints, are appended after this conversion. The translation lexicon contains only *lexeme base forms.* At runtime, morphological analysis (for Hebrew) produces the lexemes for each input word. Morphological generation (for English) is responsible for producing the various surface forms for each target-side lexeme, and transfer rule constraints create translation segments that are grammatically consistent from these surface forms.

3.5 The Hebrew-English Transfer Grammar

The Hebrew-to-English transfer grammar developed so far was initially developed manually in about two days by a bilingual speaker who is also a member of the system development team, and is thus well familiar with the underlying formalism and its capabilities. It was later revised and extended by a linguist working for about a month. The current grammar is very small and reflects the most common local syntactic differences between Hebrew and English. It contains a total of 36 rules, including 21 noun-phrase (NP) rules, one prepositional-phrase (PP) rule, 6 verb complexes and verb-phrase (VP) rules, and 8 higher-phrase and sentence-level rules for common Hebrew constructions. As we demonstrate in Section 4, this small set of transfer rules is already sufficient for producing reasonably legible translations in many cases. Figure 1 depicts an example of transfer rules for structurally transferring nouns modified by adjectives from Hebrew to English. The rules enforce number and gender agreement between the noun and the adjective. They also account for the different word order exhibited by the two languages, and the special location of the definite article in Hebrew noun phrases.

4 Results and Evaluation

The current system is targeted for translation of newspaper texts. it was developed with minimal amounts of manual labor (beyond the work that went into the existing resources used). In total, we estimate the amount if labor spent directly on the MT system to be about four to six months of human labor. Most of this time was devoted to the construction of the bilingual lexicon and stabilizing the front-end Hebrew processing in the system (Morphology and input representation issues). Once the system was reasonably stable, we devoted about two weeks of time to improving the system based on a small development set of data. For development we used a set of 113 sentences from the Hebrew daily *HaAretz*. Average sentence length was approximately 15 words. Development consisted primarily of fixing incorrect mappings before and after morphological processing and modifications to the bilingual lexicon. The small transfer grammar was also developed during this period. Given the limited resources and the limited development time, we find the results to be highly encouraging. For many of the development input sentences, translations are reasonably comprehensible. Figure 5 contains a few select translation examples from the development data.

To quantitatively evaluate the results achieved so far we tested the system on a set of 62 unseen sentences from *HaAretz*. Two versions of the system were tested on the same data set: a version using our manual transfer grammar and a version

maxwell anurpung comes from ghana for israel four years ago and since worked in cleaning in hotels in eilat

a few weeks ago announced if management club hotel that for him to leave israel according to the government instructions and immigration police

in a letter in broken english which spread among the foreign workers thanks to them hotel for their hard work and announced that will purchase for hm flight tickets for their countries from their money

Fig. 5. Select Translated Sentences from the Development Data

Table 1. System Performance Results with and without the Transfer Grammar

System	BLEU		NIST		Precision	Recall
No Grammar	0.0606	[0.0599,0.0612]	3.4176	[3.4080,3.4272]	0.3830	0.4153
Manual Grammar	0.1013	[0.1004,0.1021]	3.7850	[3.7733,3.7966]	0.4085	0.4241

with no transfer grammar at all, which amounts to a word-to-word translation version of the system. Results were evaluated using several automatic metrics for MT evaluation, which compare the translations with human-produced reference translations for the test sentences. For this test set, two reference translations were obtained. We use the BLEU [18] and NIST [19] automatic metrics for MT evaluation. We also include aggregate unigram-precision and unigram-recall as additional reported measures. The results can be seen in Table 1. To assess statistical significance of the differences in performance between the three versions of the system, we apply a commonly used bootstrapping technique [20] to estimate the variability over the test set and establish confidence intervals for each reported performance score. As expected, the manual grammar system outperforms the no-grammar system according to all the metrics.

5 Conclusions and Future Work

The focus of this article has been on the functional aspects of the Stat-XFER framework and on the implementational details of our Stat-XFER Hebrew-to-English prototype MT system. The critical issues of how to generally acquire both translation lexicons and transfer grammars were not addressed in this paper. Our group has been working extensively on developing acquistion methods under a variety of scenarios. The main approach we have been developing targets low-resource languages for which little or no sentence-parallel data is available. Our methodology under such scenarios is based on *elicitation*. We assume the availability of a small number of bi-lingual speakers of the two languages, but these need not be linguistic experts. The bi-lingual speakers create a comparatively *small* corpus of word aligned phrases and sentences (on the order of magnitude of a few thousand sentence pairs) using a specially designed elicitation tool. From this data, a transfer-rule learning module can automatically infer hierarchical syntactic transfer rules. The collection of transfer rules can then be used in our run-time system to translate previously unseen source language text into the target language. Details about this approach are described in [6] and [21].

Over the past year, we have been extensively developing an acquisition approach for language pairs for which large amounts of sentence-parallel data are available. We are currently applying these new methods for large-scale resource acquistion for our Chinese-to-English Stat-XFER system. This new approach is based on extracting translation resources from parallel sentences that are annotated with their parse structures. We use a relatively small manually word-aligned corpus for the purpose of extracting high-quality transfer rules. We use broad, automatically word-aligned, parallel corpora for extracting broad-coverage translation lexicons. The application of these methods to building a large-scale Chinese-to-English Stat-XFER system is still in progress. Preliminary results are encouraging and indicate that the system is capable of producing translations that are more grammatical and fluent than current phrase-based approaches.

Acknowledgments

This research was supported in part by NSF grants IIS-0121631 (AVENUE) and IIS-0534217 (LETRAS), and by the DARPA TIDES and GALE programs. The Hebrew Stat-XFER MT project is a joint collaboration between the CMU AVENUE group and Dr. Shuly Wintner's CL research group at the University of Haifa. It is supported by the Israel Science Foundation (grant No. 137/06); by the Caesarea Rothschild Institute for Interdisciplinary Application of Computer Science at the University of Haifa; and by NSF's Office of International Science and Education. The author thanks all members of the AVENUE MT research group at CMU for their dedicated research work described in this paper.

References

1. Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-based Translation. In: Proceedings of HLT-NAACL 2003, Association for Computational Linguistics, Edmonton, Alberta, Canada, pp. 127–133 (2003)
2. Venugopal, A., Vogel, S., Waibel, A.: Effective Phrase Translation Extraction from Alignment Models. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003), Sapporo, Japan, pp. 319–326 (2003)
3. Chiang, D.: A Hierarchical Phrase-based Model for Statistical Machine Translation. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005), Ann Arbor, Michigan, pp. 263–270 (2005)
4. Imamura, K., et al.: Example-based Machine Translation Based on Syntactic Transfer with Statistical Models. In: Proceedings of COLING-2004, Geneva, Switzerland, pp. 99–105 (2004)
5. Galley, M., et al.: Scalable Inference and Training of Context-Rich Syntactic Translation Models. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp. 961–968 (2006)
6. Lavie, A., et al.: Experiments with a Hindi-to-English Transfer-based MT System under a Miserly Data Scenario. Transactions on Asian Language Information Processing (TALIP) 2 (2003)
7. Lavie, A., et al.: A Trainable Transfer-based Machine Translation Approach for Languages with Limited Resources. In: Proceedings of Workshop of the European Association for Machine Translation (EAMT-2004), Valletta, Malta (2004)
8. Peterson, E.: Adapting a Transfer Engine for Rapid Machine Translation Development. Master's thesis, Georgetown University (2002)

9. Zhang, Y., Vogel, S.: Suffix Array and its Applications in Empirical Natural Language Processing. Technical Report CMU-LTI-06-010, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA (2006)
10. Lavie, A., et al.: Rapid Prototyping of a Transfer-based Hebrew-to-English Machine Translation System. In: Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-2004), Baltimore, MD, pp. 1–10 (2004)
11. Wintner, S.: Hebrew Computational Linguistics: Past and Future. Artificial Intelligence Review 21, 113–138 (2004)
12. Wintner, S., Yona, S.: Resources for Processing Hebrew. In: Proceedings of the MT-Summit IX workshop on Machine Translation for Semitic Languages, New Orleans (2003)
13. Yona, S., Wintner, S.: A Finite-State Morphological Grammar of Hebrew. Natural Language Engineering (to appear, 2007)
14. Wintner, S.: Finite-state Technology as a Programming Environment. In: Gelbukh, A. (ed.) CICLing 2007. LNCS, vol. 4394, pp. 97–106. Springer, Heidelberg (2007)
15. Adler, M., Elhadad, M.: An Unsupervised Morpheme-Based HMM for Hebrew Morphological Disambiguation. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp. 665–672. Association for Computational Linguistics (2006)
16. Shacham, D.: Morphological Disambiguation of Hebrew. Master's thesis, University of Haifa (2007)
17. Dahan, H.: Hebrew–English English–Hebrew Dictionary. Academon, Jerusalem (1997)
18. Papineni, K., et al.: BLEU: a Method for Automatic Evaluation of Machine Translation. In: Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, PA, pp. 311–318 (2002)
19. Doddington, G.: Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In: Proceedings of the Second Conference on Human Language Technology (HLT-2002) (2002)
20. Efron, B., Tibshirani, R.: Bootstrap Methods for Standard Errors, Confidence Intervals and Other Measures of Statistical Accuracy. Statistical Science 1, 54–77 (1986)
21. Probst, K., et al.: MT for Resource-Poor Languages Using Elicitation-Based Learning of Syntactic Transfer Rules. Machine Translation 17 (2002)

Statistical Machine Translation into a Morphologically Complex Language

Kemal Oflazer

Faculty of Engineering and Natural Sciences
Sabancı University
Istanbul, Tuzla, 34956, Turkey
oflazer@sabanciuniv.edu

Abstract. In this paper, we present the results of our investigation into phrase-based statistical machine translation from English into Turkish – an agglutinative language with very productive inflectional and derivational word-formation processes. We investigate different representational granularities for morphological structure and find that (i) representing both Turkish and English at the morpheme-level but with some selective morpheme-grouping on the Turkish side of the training data, (ii) augmenting the training data with "sentences" comprising only the content words of the original training data to bias root word alignment, and with highly-reliable phrase-pairs from an earlier corpus-alignment (iii) re-ranking the n-best morpheme-sequence outputs of the decoder with a word-based language model, and (iv) "repairing" translated words with incorrect morphological structure and words which are out-of-vocabulary relative to the training and the language model corpus, provide an non-trivial improvement over a word-based baseline despite our very limited training data. We improve from 19.77 BLEU points for our word-based baseline model to 26.87 BLEU points for an improvement of 7.10 points or about 36% relative. We briefly discuss the applicability of BLEU to morphologically complex languages like Turkish and present a simple extension to compare tokens not in a all-or-none fashion but taking lexical-semantic and morpho-semantic similarities into account, implemented in our BLEU+ tool.

1 Introduction

Statistical machine translation from English-to-Turkish poses a number of difficulties. Typologically English and Turkish are rather distant languages: while English has very limited morphology and rather fixed SVO constituent order, Turkish is an agglutinative language with a very rich and productive derivational and inflectional morphology, and a very flexible (but SOV dominant) constituent order. One implication of complex morphology is that, in parallel texts, Turkish words usually align to multiple words on the English side. When done at the word level, this is very noisy and masks the more (statistically) meaningful alignments at the sub-lexical level. Another issue of practical significance is the lack of large scale parallel text resources, with no substantial improvement expected in the near future. Thus we have to exploit our available resources maximally instead of relying on future availability of more data.

The paper is structured as follows: We first briefly discuss issues in statistical machine translation and Turkish, and review related work. We then outline how we exploit morphology, and present results from our baseline word-based and morphologically segmented models, and then present improvements provided by using a high-reliable segment of the phrase-table as additional training data, and repairing words. We then show some translation samples and discuss the applicability of BLEU to morphologically complex languages like Turkish, presenting a simple extension to compare tokens not in a all-or-none fashion, but taking lexical-semantic and morpho-semantic similarities into account.

2 Turkish and Statistical Machine Translation

Our initial experiments with statistical machine translation into Turkish [1] showed that when English – Turkish parallel data were aligned at the word level, a Turkish word would typically have to align with a complete phrase on the English side, and that sometimes these phrases on the English side could be discontinuous, and suggested that that exploiting sub-lexical structure would be a fruitful avenue to pursue. For instance, the Turkish word *tatlandırabileceksek* could be translated as (and hence would have to align to something equivalent to) 'if we were going to be able to make [something] acquire flavor'. This word could be aligned as follows (shown with co-indexation of Turkish surface morphemes and English words):[1]

$$\text{(tat)}_1\text{(lan)}_2\text{(dır)}_3\text{(abil)}_4\text{(ecek)}_5\text{(se)}_6\text{(k)}_7$$
$$\text{(if)}_6\text{(we are)}_7\text{(going to)}_5\text{(be able)}_4\text{(to make)}_3\text{[something](acquire)}_2\text{(flavor)}_1$$

The productive morphology of Turkish implies potentially a very large vocabulary size, as noun roots have about 100 inflected form and verbs have much more. These numbers are much higher when derivations are considered: one can generate thousands of words from a single root when, say, only at most two derivations are allowed. Thus, sparseness is an important issue given that we have very modest parallel resources available.. However, Turkish employs about 30,000 root words and about 150 distinct suffixes, so when morphemes are used as the units in the parallel texts, the sparseness problem can be alleviated to some extent.

Our approach in this paper is to represent Turkish words with their morphological segmentation. We use lexical morphemes instead of surface morphemes, as most surface distinctions are manifestations of word-internal phenomena such as vowel harmony, and morphotactics. With lexical morpheme representation, we can abstract away such word-internal details *and* conflate statistics for seemingly different suffixes, as at this level of representation words that look very

[1] Note that on the English side, the filler for [something] would come in the middle of this phrase.

different on the surface, look very similar.[2] For instance, although the words *evinde* 'in his house' and *masasında* 'on his table' look quite different, the lexical morphemes except for the root are the same: `ev+sh+nda` vs. `masa+sh+nda` (see Oflazer and Durgar-El Kahlout [3] for details.)

We should however note that although employing a morpheme based representations dramatically reduces the vocabulary size on the Turkish side, it also runs the risk of overloading the decoder mechanisms to account for *both* word-internal morpheme sequencing and sentence level word ordering.

Using morphology in statistical machine translation has been recently addressed by researchers translation from or into morphologically rich(er) languages. Niessen and Ney [4] have used morphological decomposition to improve alignment quality. Yang and Kirchhoff [5] use phrase-based backoff models to translate words that are unknown to the decoder, by morphologically decomposing the unknown source word. They particularly apply their method to translating *from* Finnish – another language with very similar structural characteristics to Turkish. Corston-Oliver and Gamon [6] normalize inflectional morphology by stemming the word for German-English word alignment. Lee [7] uses a morphologically analyzed and tagged parallel corpus for Arabic-English statistical machine translation. Zolmann et al. [8] also exploit morphology in Arabic-English statistical machine translation. Popovic and Ney [9] investigate improving translation quality from inflected languages by using stems, suffixes and part-of-speech tags. Goldwater and McClosky [10] use morphological analysis on Czech text to get improvements in Czech to English statistical machine translation. Recently, Minkov et al. [11] have used morphological postprocessing *on the output side* using structural information and information from the source side, to improve translation quality.

3 Exploiting Morphology

Our parallel data consists mainly of documents in international relations and legal documents from sources such as the Turkish Ministry of Foreign Affairs, EU, etc. We process these as follows:

i) We segment the words in our Turkish corpus into lexical morphemes whereby differences in the surface representations of morphemes due to word-internal phenomena are abstracted out to improve statistics during alignment. Note that as with many similar languages, the segmentation of a surface word is generally ambiguous, we first generate a representation using our morphological analyzer [12] that contains both the lexical segments and the morphological features encoded for all possible segmentations and interpretations of the word and perform morphological disambiguation using morphological features [13]. Once the contextually salient morphological interpretation

[2] This is in a sense very similar to the more general problem of lexical redundancy addressed by Talbot and Osborne [2] but our approach does not require the more sophisticated solution there.

is selected, we discard the features leaving behind the lexical morphemes making up a word.[3]

ii) We tag the English side using TreeTagger [14], which provides a *lemma* and a *part-of-speech* for each word. We then remove any tags which do not imply an explicit morpheme or an exceptional form. So for instance, if the word *book* gets tagged as *+NN*, we keep *book* in the text, but remove *+NN*. For *books* tagged as *+NNS* or *booking* tagged as *+VVG*, we keep *book* and *+NNS*, and *book* and *+VVG*. A word like *went* is replaced by *go +VVD*.[4]

iii) From these morphologically segmented corpora, we also extract for each sentence, the sequence of roots for open class content words (nouns, adjectives, adverbs, and verbs). For Turkish, this corresponds to removing *all* morphemes and any roots for closed classes. For English, this corresponds to removing all words tagged as closed class words along with the tags such as *+VVG* above that signal a morpheme on an open class content word. We use this to augment the training corpus and bias content word alignments, with the hope that such roots may get a better chance to align without any additional "noise" from morphemes and other function words.

Table 1 presents various statsitical information about this parallel corpus.One can note that Turkish has many more distinct word forms (about twice as many as English), but has much less number of distinct content words than English.[5] For language models in decoding and n-best list rescoring, we use, in addition to the training data, a monolingual Turkish text of about 100,000 sentences (in a segmented and disambiguated form).

A typical sentence pair in our (fully-segmented) data looks like the following, where we have highlighted the content root words with bold font, coindexed them to show their alignments and bracketed the "words" that BLEU evaluation on test would consider.

T: [**kat**$_1$ +hl +ma] [**ortaklık**$_2$ +sh +nhn] [**uygula**$_3$ +hn +ma +sh] [,] [**ortaklık**$_4$] [**anlaşma**$_5$ +sh] [**çerçeve**$_6$ +sh +nda] [**izle**$_7$ +hn +yacak +dhr] [.]

E: the **implementation**$_3$ of the **accession**$_1$ **partnership**$_2$ will be **monitor**$_7$ +vvn in the **framework**$_6$ of the **association**$_4$ **agreement**$_5$.

Note that when the morphemes/tags (tokens starting with a +) are concatenated, we get the "word-based" version of the corpus, since surface words are directly

[3] This disambiguator has about 94% accuracy.

[4] Ideally, it would have been very desirable to actually do derivational morphological analysis on the English side, so that one could for example analyze *accession* into *access* plus a marker indicating nominalization.

[5] The training set in the first row of 1 was limited to sentences on the Turkish side which had at most 90 tokens (roots and bound morphemes) in total in order to comply with requirements of the GIZA++ alignment tool. However when only the content words are included, we have more sentences to include since much less number of sentences violate the length restriction when morphemes/function word are removed.

Table 1. Statistics on Turkish and English training and test data, and Turkish morphological structure

TURKISH	Sent.	Words (UNK)	Unique Words	Morph.	Unique Morph.	Morph./ Word	Unique Roots	Unique Suffixes
Train	45,709	557,530	52,897	1,005,045	15,081	1,80	14,976	105
Content	56,609	436,762	13,767					
Tune	200	3,258	1,442	6,240	859	1.92	810	49
Test	649	10,334 (545)	4,355	18,713	2,297	1.81	2,220	77
ENGLISH								
Train	45,709	723,399	26,747					
Content	56,609	403,162	19,791					
Test	649	13,484 (231)	3,220					

recoverable from the concatenated representation. We use this word-based representation also for word-based language models used for rescoring.

4 Experiments and Results

We employ the phrase-based statistical machine translation framework [15], and use the Moses toolkit [16], and the SRILM language modelling toolkit [17], and evaluate our decoded translations using the BLEU measure [18], using a *single* reference translation.

We performed four sets of experiments employing different morphological representations on the Turkish side and adjusting the English representation accordingly wherever needed.

1. **Baseline:** English and Turkish sentences are represented with “full” words. For example `[kitap+sh+nhn]` (representing *kitabının* (of his book) would be used on the Turkish side and `[book+NNS]` (representing *books*) on the English side.
2. **Full Morphological Segmentation:** English and Turkish sentences are represented tokens representing root words, bound morphemes/tags. For example for the examples above, the three tokens `[kitap] [+sh] [+nhn]` would be used on the Turkish side and the two tokens `[book] [+NSS]` on the English side.
3. **Root+Morphemes Segmentation:** Turkish sentences are represented with roots and combined morphemes. For English sentences, we used the same representation in (2). For example for the Turkish word above, the two tokens `[kitap] [+sh+nhn]` would be used.
4. **Selective Morphological Segmentation:** A systematic analysis of the alignment files produced by GIZA++ for a small subset of the training sentences showed that certain morphemes on the Turkish side were almost consistently never aligned with anything on the English side: e.g., the compound

noun marker morpheme in Turkish (+sh) does not have a corresponding unit on the English side, as English noun-noun compounds do not carry any overt markers. Such markers were never aligned to anything or were aligned almost randomly to tokens on the English side. Further, since we perform derivational morphological analysis on the Turkish side but not on the English side, we also noted that most verbal nominalizations on the English side were just aligned to the verb roots on the Turkish side and the additional markers on the Turkish side indicating the nominalization, and various agreement markers etc., were mostly unaligned.
For just these cases, we selectively attached such morphemes (and in the case of verbs, the intervening morphemes) to the root, but otherwise kept other morphemes, especially any case morphemes, still by themselves, as they almost often align with prepositions on the English side quite accurately. [6]
In this case, the Turkish word above would be represented by the two tokens `[kitap+sh] [+nhn]`. English words are still represented as in case 2 above.

For each the four representational schemes we went through the following process:

1. The training corpus was augmented with the content word parallel data.[7]
2. A 5-gram morpheme-based language model was constructed for Turkish (to be used by the decoder) using the Turkish side of the training data along with an additional monolingual Turkish text of about 100K sentences *represented in the same scheme as the Turkish side of the training data.*
3. Training was performed and the phrase table was extracted using a maximum phrase size of 7. Minimum error rate training with the tune set did not provide any tangible improvements.[8]
4. The test corpus was decoded using the Moses decoder with modified parameters *-dl -1* to allow for long distance movement and *-weight-d 0.1* to avoid penalizing long distance movement.[9] The decoder also produced 1000-best candidate translations.

[6] It should be noted that what to selectively attach to the root should be considered on a per-language basis; if Turkish were to be aligned with a language with similar morphological markers, this perhaps would not have been needed. Again one perhaps can use methods similar to those suggested by Talbot and Osborne [2].

[7] Using the content word data improved performance for all representations *except* the baseline.

[8] We ran MERT on the baseline model and the morphologically segmented models forcing *-weight-d* to range a very small around 0.1, but letting the other parameters range in their suggested ranges. Even though the procedure came back claiming that it achieved a better BLEU score on the tune set, running the new model on the test set did not show any improvement at all. This may have been due to the fact that the initial choice of *-weight-d* along with *-dl* set to -1 provides such a drastic improvement that perturbations in the other parameters do not have much impact.

[9] We arrived at this combination by experimenting with the decoder to avoid the almost monotonic translation we were getting with the default parameters. These parameters boosted the BLEU scores substantially compared to default parameters used by the decoder.

5. For representation schemes 2 through 4, the 1000-best candidates were then converted into word-based representation (by just attaching any morpheme/ tag tokens to the stem to the left) and rescored using weighted combination of the 4-gram *word-based* language model score and the translation score produced by the decoder. The combination weights were optimized on the tune corpus.
6. The top rescored candidate translations were selected and compared with the (single) reference translation using the BLEU measure.

The results of these set of experiments are presented in Table 2.

Table 2. BLEU Results for the four representational schemes

Experiment/Decoder Parameters	**BLEU**
Word-based Baseline/Default Parms	16.13
Word-based Baseline/Modified Parms	19.77
Full morphological Segmentation/Default Parms	13.55
Full morphological Segmentation/Modified Parms	22.18
Root+Morphemes Segmentation/Modified Parms	20.12
Selective Morphological Segmentation/Modified Parms	**24.61**

The best BLEU results are obtained with selective morphological segmentation (24.61) and represents a relative improvement of 23%, compared to the respective baseline of 19.77. One should also note that the default decoding parameters used by the Moses decoder produces much worse results especially for the fully segmented model.

Our further experiments are only executed on top of the results of the best performing representation – selective morphological segmentation.

4.1 Augmenting the Training Data

In order to overcome the disadvantages of the small size of our parallel data, we experimented with ways of using portions of the phrase table that is generated by the training process, as additional training data.

The phrase extraction process performs English-Turkish and Turkish-English alignments using the GIZA++ tool and then combines these alignments with some additional post-processing and extracts "phrases", sequences of source and target tokens that align to tokens in the other sequence. Such phrases do not necessarily correspond to linguistic phrases.

The following is a very small portion of the phrase table generated by the Moses training process for the selective morphological segmentation representation above:

```
good word ||| müjde ||| 1 0.25 0.5 0.0037281 2.718
good ||| düzgün ||| 0.0714286 0.0322581 0.00487805 0.0018382 2.718
good ||| en iyi ||| 0.388889 0.25589 0.0341463 0.00605536 2.718
good ||| en ||| 0.00833333 0.0194715 0.00487805 0.0257353 2.718
good ||| eşya ||| 0.030303 0.32967 0.00487805 0.0551471 2.718
good ||| güzel ||| 0.2 0.0645161 0.0097561 0.0036765 2.718
good ||| iyi bir ||| 0.2 0.492308 0.0146341 0.0126794 2.718
good ||| iyi ||| 0.85 0.492308 0.497561 0.235294 2.718
good +NNS ||| mal +lar ||| 0.540741 0.605839 0.356098 0.152574 2.718
```

The first and second parts of any entry in the phrase table are the English (e) and Turkish (t) parts of a pair of aligned phrases. Among the sequences of the numbers that follow, the first is $p(e|t)$, the conditional probability that the English phrase is e given that the Turkish phrase is t; the third number is $p(t|e)$ and captures the probability of the symmetric situation.

Among these phrase table entries, those with $p(e|t) \approx p(t|e)$ and $p(t|e)+p(e|t)$ larger than some threshold can be considered as reliable mutual translations in that they mostly translate to each other and not much to others. So we extracted those phrases with $0.9 \leq p(e|t)/p(t|e) \leq 1.1$ and $p(t|e) + p(e|t) \geq 1.5$ and added them to further bias as the alignment process.

The six steps listed earlier were repeated for this augmented selectively morphologically segmented training corpus. The BLEU result that was obtained was 26.16, showing a 32.3% relative improvement over the 19.77 baseline, and 6.3% relative improvement over the previous result.

5 Word Repair

The detailed BLEU results of 26.16, [53.0/29.9/20.3/14.6] for our best performing model, indicates that only 53% of the words in the candidate translations are determined correctly. However, when *all words* in both the candidate and reference translations are reduced to roots and BLEU is computed again we get the *root* BLEU results of 30.62, [64.6/35.7/23.4/16.3]. This shows that we are getting 64.6% of the roots in the translations correct but only 53% of the words forms are correct indicating that for such cases, the roots are correct but the full word forms are either incorrect or do not match the correct word form in the reference translation. Such words can be classified into three groups:

1. Morphologically malformed words – words with the correct root word but with morphemes that are either categorically incorrect (e.g., case morpheme on a verb), or morphotactically incorrect (e.g., morphemes in the wrong order).
2. Morphologically well-formed words which are out-of-vocabulary (OOV) relative to the training corpus and the language model corpus.[10]
3. Morphologically well-formed words which are *not* out-of-vocabulary relative to the training corpus and the language model corpus, but do not match the reference.

[10] Note that since Turkish has a very large number possible word forms, there really are no well-formed words which are OOV, though there may be well-formed words which are extremely low frequency. It is such words that we aim to identify here.

Words in classes 1 and 2 can be identified easily: Words in class 1 would be rejected using our morphological analyzer, while words for case 2 would be accepted by the morphological analyzer but would not be in the vocabulary of the training and language model corpora. However, but we have no way knowing without looking at the reference if a word falls in class 3.

The approach we have taken to deal with the words for case 1 is as follows:

1. Using a finite state model of lexical morpheme structure of possible Turkish words, with morphemes being as the symbols (except for the letters in roots), we use error-tolerant finite state recognition [19] to generate morphologically correct word forms with the same root but with the morpheme structure up to 2 unit morpheme edit operations (add, delete, substitute, transpose morphemes) away. We do this for every morphologically malformed word in a candidate translation sentence. For instance, the word form (in lexical morpheme representation) *gel+da+ydh* is malformed and possible correction at distance 1 are {*gel+yacak+ydh*, *gel+mhs+ydh*, *gel+dh+ydh*, *gel+sa+ydh*, *gel+ya+ydh*}. We convert the sentence to a lattice representation replacing each malformed with the correct alternatives.
2. The resulting lattice is then rescored with the language model to pick the best alternative for each malformed word. In this step, the morpheme-based language model performed better than the word-based-language model.

When words that are one morpheme operation away were considered as possible alternatives, the BLEU score improved to 26.46; with words that are two morpheme operations away, the improvements was to 26.49.

We took a similar approach for handling words for case 2. We generated alternatives for these morphologically correct but OOV words that were 1 and 2 morpheme operation distance away, but this time we restricted the alternatives to the vocabulary of the training and language model corpora. With both distances 1 and 2 we got a further improvement to 26.87 BLEU points. All in all, word repair provides an additional improvement of 2.7% relative improvement (compared to 26.16) and the final BLEU score represents a relative improvement of 35.9% over the baseline.

6 Sample Translations

When we consider input English sentences that are between 5 to 15 words, the translation quality of our system is considerably better than the BLEU score for the complete test set. Below we present translations of three sentences from the test data along with the literal paraphrases of the translation and the reference versions. The first two are quite accurate and acceptable translations while the third clearly has missing, incorrect but also interesting parts: we see that the English *key* is translated to the Turkish *kilit* (lock) which is the correct collocational translation.

Input: 1 . everyone's right to life shall be protected by law .
Translation: 1 . herkesin yaşama hakkı kanunla korunur.
Literally: 1. everyone's living right is protected with law .
Reference: 1 . herkesin yaşam hakkı yasanın koruması altındadır .
Literally: 1. everyone's life right is under the protection of the law .

Input: promote protection of children's rights in line with eu and international standards .
Translation: çocuk haklarının korunmasının **ab ve uluslararası standartlar**a uygun şekilde geliştirilmesi .
Literally: develop protection of children's rights in accordance with eu and international standards .
Reference: ab ve uluslararası standartlar doğrultusunda çocuk haklarının korunmasının teşvik edilmesi .
Literally: in line with eu and international standards promote/motivate protection of children's rights .

Input: as a key feature of such a strategy, an accession partnership will be drawn up on the basis of previous european council conclusions.
Translation: bu stratejinin kilit unsuru bir katılım ortaklığı belgesi hazırlanacak kadarın temelinde , bir önceki avrupa konseyi sonuçlarıdır .
Literally: a lock feature of this strategy accession partnership document will be prepared ??? based are previous european council resolutions .
Reference: bu stratejinin kilit unsuru olarak, daha önceki ab zirve sonuçlarına dayanılarak bir katılım ortaklığı oluşturulacaktır.
Literally: as a lock feature of this strategy an accession partnership based on earlier eu summit resolutions will be formed .

7 Discussion and Conclusions

For English-to-Turkish statistical machine translation, employing a language-pair specific morphological representation somewhere in between using full word-forms and fully morphologically segmented representations along with augmenting the limited training data with content words and highly reliable phrases provides the most leverage. Repairing morphologically malformed words and OOV words provides some additional improvement – there may be some more improvements along these lines by applying repair to low confidence words that can be identified by a scheme suggested by Zens and Ney [20].

Translation into Turkish seems to involve processes that are somewhat more complex than standard statistical translation models: sometimes a single word on the Turkish is synthesized from the translations of two or more phrases, and errors in any translated morpheme or its morphotactic position render the synthesized word incorrect, even though the rest of the word can be quite fine. This though indirectly implies that BLEU is particularly harsh for Turkish and the morpheme based-approach, because of the all-or-none nature of token comparison when computing the BLEU score. Furthermore, there are also cases where words with different morphemes have very close morphosemantics, convey the relevant meaning and are almost interchangeable:

– *gel+hyor* (*geliyor* - he is coming) vs. *gel+makta* (*gelmekte* - he is (in a state of) coming) are essentially the same. On a scale of 0 to 1, one could rate these at about 0.95 in similarity.
– *gel+yacak* (*gelecek* - he will come) vs. *gel+yacak+dhr* (*gelecektir* - he *will* come) in a sentence final position. Such pairs could be rated perhaps at 0.90 in similarity.
– *gel+dh* (*geldi* - he came (evidential past tense)) vs. *gel+mhs* (*gelmiş* - he came (hearsay past tense)). These essentially mark past tense but differ in how the speaker relates to the event and could be rated at perhaps 0.70 similarity.

We have developed a tool, *BLEU+* [21] that implements *a slightly different formulation of token similarity* in BLEU computation considering both root word similarity, by considering synonyms (e.g. as in Meteor [22] and hypernyms, using a WordNet, and morphosemantic similarity considering (almost) synoymous morphemes. *BLUE+* can also compute METEOR scores, oracle BLEU scores assuming all morphologically malformed words are perfectly corrected, and also root BLEU scores providing for a better understanding of the quality and the limits of the output translation.

Acknowledgements

This work was supported by TÜBİTAK – The Turkish National Science and Technology Foundation under project grant 105E020. Some of the work reported here was performed while the author was on sabbatical at the Language Technologies Institute at Carnegie Mellon University. İlknur Durgar El-Kahlout helped with a lot of the details and experiments. Şeyma Mutlu implemented the word-repair code.

References

1. Durgar El-Kahlout, I., Oflazer, K.: Initial explorations in English to Turkish statistical machine translation. In: Proceedings on the Workshop on Statistical Machine Translation, pp. 7–14. Association for Computational Linguistics, New York City (2006)
2. Talbot, D., Osborne, M.: Modelling lexical redundancy for machine translation. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, July 2006, pp. 969–976 (2006)
3. Oflazer, K., Durgar El-Kahlout, I.: Exploring different representational units in English-to-Turkish statistical machine translation. In: Proceedings of the Second Workshop on Statistical Machine Translation, pp. 25–32. Association for Computational Linguistics, Prague, Czech Republic (2007)
4. Niessen, S., Ney, H.: Statistical machine translation with scarce resources using morpho-syntatic information. Computational Linguistics 30, 181–204 (2004)

5. Yang, M., Kirchhoff, K.: Phrase-based backoff models for machine translation of highly inflected languages. In: Proceedings of EACL, pp. 41–48 (2006)
6. Corston-Oliver, S., Gamon, M.: Normalizing German and English inflectional morphology to improve statistical word alignment. In: Proceedings of AMTA, pp. 48–57 (2004)
7. Lee, Y.-S.: Morphological analysis for statistical machine translation. In: Proceedings of HLT-NAACL 2004 - Companion Volume, pp. 57–60 (2004)
8. Zollmann, A., Venugopal, A., Vogel, S.: Bridging the inflection morphology gap for Arabic statistical machine translation. In: Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers, New York City, USA, pp. 201–204 (2006)
9. Popovic, M., Ney, H.: Towards the use of word stems and suffixes for statistical machine translation. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC), pp. 1585–1588 (2004)
10. Goldwater, S., McClosky, D.: Improving statistical MT through morphological analysis. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Vancouver, British Columbia, Canada, pp. 676–683 (2005)
11. Minkov, E., Toutanova, K., Suzuki, H.: Generating complex morphology for machine translation. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pp. 128–135. Association for Computational Linguistics, Prague, Czech Republic (2007)
12. Oflazer, K.: Two-level description of Turkish morphology. Literary and Linguistic Computing 9, 137–148 (1994)
13. Yüret, D., Türe, F.: Learning morphological disambiguation rules for Turkish. In: Proceedings of the Human Language Technology Conference of the NAACL, Main Conference, New York City, USA, pp. 328–334 (2006)
14. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: Proceedings of International Conference on New Methods in Language Processing (1994)
15. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: Proceedings of HLT/NAACL (2003)
16. Koehn, P., et al.: Moses: Open source toolkit for statistical machine translation. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007) – Companion Volume (2007)
17. Stolcke, A.: Srilm – an extensible language modeling toolkit. In: Proceedings of the Intl. Conf. on Spoken Language Processing (2002)
18. Papineni, K., et al.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, University of Pennsylvania, pp. 311–318 (2002)
19. Oflazer, K.: Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. Computational Linguistics 22, 73–90 (1996)
20. Zens, R., Ney, H.: N-gram posterior probabilities for statistical machine translation. In: Proceedings on the Workshop on Statistical Machine Translation, pp. 72–77. Association for Computational Linguistics, New York City (2006)
21. Tantuğ, C., Oflazer, K., Durgar El-Kahlout, I.: BLEU+: a tool for fine-grained BLEU computation (Submitted, 2007)
22. Banerjee, S., Lavie, A.: METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In: Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, Michigan, pp. 65–72 (2005)

Translation Paraphrases in Phrase-Based Machine Translation

Francisco Guzmán and Leonardo Garrido

Center for Intelligent Systems
ITESM Campus Monterrey, Mexico
guzmanhe@gmail.com, leonardo.garrido@itesm.mx

Abstract. In this paper we present an analysis of a phrase-based machine translation methodology that integrates paraphrases obtained from an intermediary language (French) for translations between Spanish and English. The purpose of the research presented in this document is to find out how much extra information (i.e. improvements in translation quality) can be found when using Translation Paraphrases (TPs). In this document we present an extensive statistical analysis to support conclusions.

1 Introduction

Statistical methods have proven to be very effective when addressing linguistic problems, specially when dealing with Machine Translation [1]. There have been several attempts to improve the performance of such systems. Non-syntactic phrase-based translation systems[2] certainly outperform word-based systems[3].

Nevertheless, Statistical Machine Translation (STMT) effectiveness is limited to situations where large amounts of data are available. Such a condition, limits the performance of SMT systems over"low density" language pairs [4]. Scarce training data, often leads to a low coverage problem, that is, a low amount of learned translations for a language pair.

There are several efforts trying to improve translation quality of STMT systems. Many state-of-the-art systems involve the introduction of syntactic information to phrase-based machine translations [5,6,7,8,9].

On the other hand, we find several efforts which do not use syntactic information. One main topic of discussion is the usage of paraphrases. For example Callison [4] improves translation quality by giving alternatives to broaden coverage of a phrase-based machine translation system through the use of paraphrases. They use paraphrases in cases when a phrase is not found in their phrase-tables. Other effort is conducted by Guzman and Garrido [10] who obtain what they call "translation paraphrases" from pivoting through an intermediary language.

In this paper we analyze their methodology to assess whether the inclusion of Translation Paraphrases (TP) in a STMT system are useful to improve translation quality, in comparison to systems that do not include such features.

This paper is organized as follows: In Sec.2 we explain the methodology followed throughout our experimentation. In Sec.3 explain thoroughly the results and their statistical analysis. In Sec.4, we discuss the implications of results and we propose further research directions regarding translation paraphrases.

1.1 Translation Paraphrases

The strategy proposed by [10] to tackle the coverage problem is to extend phrase-tables that are used for phrase-based STMT with translation paraphrases learned from a third language. Figure 1 exemplifies this point. In their scope, translation paraphrases are the mechanism of preserving meaning through translation. While bridging through a third language, translation paraphrases are to give more flexible interpretations of source texts, as well as to reinforce translations that are more likely to be good translations regardless of the translation process.

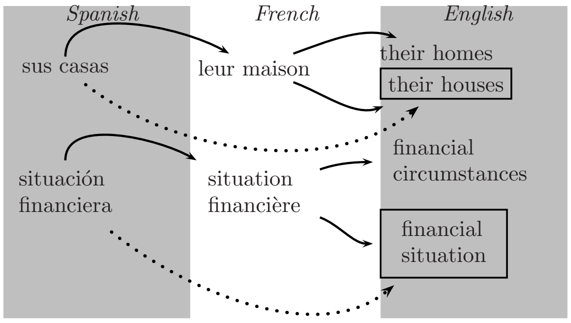


Fig. 1. Example of a translation paraphrase: When translating from Spanish to English with a Spanish-English trained phrase-table, we only get "their homes" English phrase as an alternative to "sus casas" Spanish phrase. However, if using translation paraphrases issued from French, we get "their homes" and "their houses" alternatives.

2 Experimental Setup

2.1 System Training

Every STMT system needs to be trained over a pair of aligned corpora. Aligned corpora are collections of documents, for which each line in one document has its counterpart in other language, which has been translated by a human (Fig. 2). Using the information in these documents, a STMT systems constructs a model that estimates the likelihood of a phrase in one language to be translated into another.

After the model training, we end up with a phrase-table, which is a collection of correspondences between phrases in both languages with their corresponding probability of being translated into each other.

In our experiments we trained the three combinations of language pairs in the set {English, French, Spanish }. Thus, we obtained three phrase tables: English-French, English-Spanish and French-Spanish. For the purposes of our experiments,

12 there needs firstly to be clarity between all of the groups of this house and then between this house and the commission .	**12** la clarté doit tout d'abord régner entre tous les groupes de cette assemblée et ensuite entre le parlement et la commission .
13 we should not find ourselves late in the day in the unfortunate position where the one or other institution creates an unnecessary fracture in institutional relationships .	**13** nous ne devons pas nous retrouver en fin de compte dans la position malheureuse où l'une ou l'autre institution crée une rupture inutile dans les relations institutionnelles .

Fig. 2. Example of an aligned corpora, extracted from Europarl corpus

we trained over aligned corpora containing 10k, 20k, 40k, 80k and 100k sentence-pairs (k-size) issued from the European Parliament Proceedings Corpus (Europarl) [11] from year 2001. For the model training, we used Giza++ [12].

2.2 Phrase Table Consolidation

In their paper Guzman and Garrido[10] describe a methodology for creating TPs from a trilingual aligned corpus. They combine the phrase-tables issued from training English-French and French-Spanish language pairs to obtain a English-Spanish Translation Paraphrases phrase-table using the following equation:

$$p_{\mathrm{otp}}(e|s) = \sum_{f} p(e|f)p(f|s) \quad . \tag{1}$$

That is, the marginalized probability of translating a Spanish phrase to a French phrase and then translating that phrase to an English phrase. For instance, let us call the phrase-table containing these new probabilities, the Only-TP phrase table.

Furthermore, Guzman and Garrido describe a method for combining the Only-TP phrase-table with a phrase-table trained directly from English-Spanish (a Non-TP phrase-table) using the following model:

$$p_{\mathrm{mix}}(e|s) = \alpha\, p_{\mathrm{otp}}(e|s) + (1-\alpha)\, p_{\mathrm{ntp}}(e|s) \quad . \tag{2}$$

In our experiments we trained Non-TP and Only-TP phrase-tables for each of the k-sizes and afterwards we combined them to produce mixed phrase-tables by using (2) while varying alpha from 0.1 to 0.9. After this stage, we ended up with 55 phrase tables (eleven for each k-size). For clarity, see Fig.3.

Having a phrase-table, half of the training to produce a STMT system is done. The second half is to fine-tune the decoder, which speaking generally is the piece of software that uses the information in the phrase-tables to produce a translation. With a phrase-table we can build a rough non-tuned STMT system that might be able of performing low quality translations. Therefore the second part of the training phase has to do with tuning the parameters of the decoder for an optimal output.

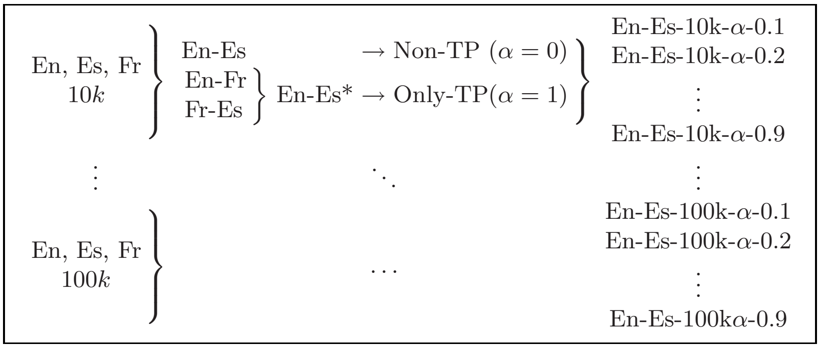


Fig. 3. Outline of the experimental procedure to merge phrase-tables. First we train the En-Es, Fr-Es and En-Fr models to obtain their corresponding phrase-tables. Then we obtain the translation paraphrase-table from merging En-Fr and Fr-Es phrase-tables. Finally, we merge the En-Es* translation paraphrase-table with the En-Es phrase-table at different levels to obtain the mixed phrase-tables.

2.3 MERT Training

The Minimum Error Rate Training (MERT) [13] is the process with which we tune the factors of the log-linear model described in Och and Ney in[14]. Roughly, the process consists in testing combinations of parameters and determine which combination give the best output. This is done by translating an specific document and then evaluating the translation quality. In other words, we train the decoder's parameters for it to be an "expert" in translating a given set of documents.

In this training phase we used a random subset of 100 lines randomly extracted from the documents of Europarl Corpus of 2002. We ran the MERT over each of the 55 phrase-tables, to ensure that each one was configured to perform at its best.

2.4 Translation

Upon the conclusion of our systems' training, we wanted to test the performance of each configuration against a controlled testing set of 30 samples. Each of those samples contained 50 lines of text, which were randomly extracted from the Europarl test set [11], containing documents from October 2000 to December 2000. This random sampling process was done in order to diminish the effects of the clustering of translation difficulty. An equivalent process was performed by [15] in their "broad sampling" where they followed a deterministic rule to form samples containing lines of text from different parts of the corpus.

For translating the samples, we used the moses decoder [16] with the parameters issued from the MERT training.

In order to evaluate the phrasal translation quality, we used the BLEU metric [17] (which is one of the standard measurements of quality of translation) with a single source of reference translation. Although recent studies suggest that BLEU's correlation with human judgments is not as strong as previously thought

[18], doing manual evaluation implies having infrastructure and resources (human judges, evaluation framework, etc.) which we do not currently possess.

3 Results and Discussion

The results of our experiments are summarized in Tab.1: At a first glance it is difficult to perform an analysis by just looking at these results. The first piece of information that out stands is the maxima for each group of training sentence pairs (k-group). From the table we can see that as the number of pairs increases, the maxima moves from an α of 0.6 at 10k to 0.5 at 20k and 40k; and to 0.4 at 80k and 100k. This suggests that as we increase the training data size, translation paraphrases become less handy.

To better analyze the information gathered throughout our experiments we performed an statistical analysis for each group.

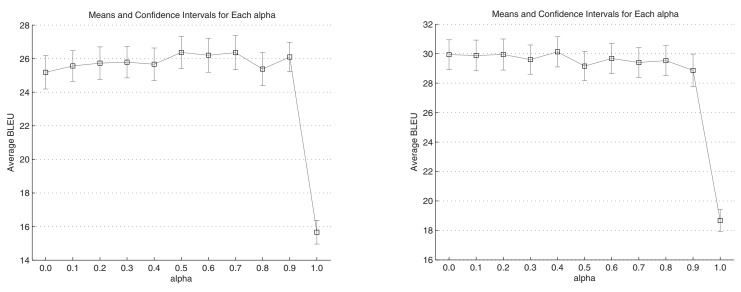
Table 1. Experimental results presented by alpha and number of training sentence pairs. For each registry we have the average BLEU of the 30 translation problems ($\bar{x}$) and their standard error($\sigma_{\bar{x}}$).

	10k		20k		40k		80k		100k	
α	$\bar{x}$	$\sigma_{\bar{x}}$	$\bar{x}$	$\sigma_{\bar{x}}$	$\bar{x}$	$\sigma_{\bar{x}}$	$\bar{x}$	$\sigma_{\bar{x}}$	$\bar{x}$	$\sigma_{\bar{x}}$
0.0	24.01	0.43	25.19	0.49	27.58	0.44	29.93	0.49	30.60	0.48
0.1	24.22	0.46	25.56	0.45	28.08	0.48	29.88	0.51	30.51	0.51
0.2	24.39	0.46	25.73	0.47	28.17	0.47	29.94	0.52	30.40	0.48
0.3	24.36	0.46	25.79	0.46	28.04	0.46	29.60	0.49	30.69	0.49
0.4	24.24	0.46	25.66	0.47	28.26	0.46	**30.12**	0.50	**30.72**	0.48
0.5	24.23	0.44	**26.37**	0.47	**28.69**	0.45	29.15	0.49	30.43	0.50
0.6	**24.61**	0.46	26.20	0.49	28.23	0.45	29.67	0.50	30.31	0.48
0.7	24.10	0.47	26.36	0.50	28.34	0.46	29.40	0.50	30.63	0.47
0.8	24.18	0.46	25.38	0.48	28.23	0.44	29.52	0.50	30.29	0.47
0.9	23.94	0.43	26.10	0.43	27.44	0.45	28.86	0.54	29.51	0.44
1.0	13.74	0.35	15.66	0.34	17.31	0.37	18.68	0.37	19.39	0.37

3.1 Confidence Intervals

Since our research question was to find if we obtained any improvements in translation quality by the usage of TPs, a good starting point was to look at the confidence intervals for each mean, to determine if the systems belong to the same population (that is, they share a common population mean and thus, no improvement has been made).

In the Fig. 4 we display the confidence intervals for 20k and 80k to a level of 95%. As one can see, for both graphs it might seem clear that, excluding the rightmost system (α=1.0), the others belong to the same group. Nevertheless, this conclusion wouldn't be as valid if we had different groups. For instance, if for 20k we only analyze the confidence intervals for α={0.0,0.5,1.0} we wouldn't



(a) **(b)**

Fig. 4. Confidence Intervals for (a)20k and (b) 80k k-sizes

draw the same conclusions since their means do not fall into each other's confidence intervals, suggesting that their means are different. Therefore we needed to perform other analysis to obtain sound conclusions. Withal, what we do can conclude is that a system trained with only translation paraphrases (α=1.0) perform worse than any other system.

3.2 One Way Analysis of Variance (ANOVA)

The ANOVA test is useful when dealing with several groups for which we want test if they belong to a single population (meaning that they share the same mean). Although, it serves only to test the null hypothesis of all means being equal, and does not tells us anything about differences between individual groups, it is relevant that we reject the experiment's null hypothesis (all means are equal) so we can decouple individual groups to do pairwise comparison under a least significant differences scheme (LSD).

In Tab. 2 we show the results of the ANOVA tests for every k-group. As we can see the p-values are very low (basically zero), allowing us to reject the experiment's null hypothesis for each of the k-groups. This is not news, because from plotting confidence intervals we could see that the mean performance of the systems with α=1.0 was very different from the others. But as we said before, rejecting the experiment's null hypothesis allows us to perform pairwise comparisons.

3.3 Unplanned Pairwise Comparisons

Since we had many groups of α (α-groups) to compare, we decided to analyze only three α-groups: Non-TP (α=0.0), Best-TP (the α-group with best performance within a k-group) and Only-TP (α=1.0). The method for comparisons that we used was an approximate randomization test for the paired sample comparison of means. We used this test because for every system, we ran them over the same set of problems, and therefore the different samples were not independent. Besides, being a computer intensive method, we needed not to care

Table 2. ANOVA results for each k-group

k-size	Source	SS	df	MS	F	pvalue
10k	Groups	3011	10	301.1	51	0.00
	Error	1883.3	319	5.9		
	Total	4894.3	329			
20k	Groups	2866	10	286.6	44.87	0.00
	Error	2037.7	319	6.39		
	Total	4903.7	329			
40k	Groups	3213.4	10	321.34	53.13	0.00
	Error	1929.3	319	6.05		
	Total	5142.7	329			
80k	Groups	3297.9	10	329.79	45.32	0.00
	Error	2321.1	319	7.28		
	Total	5619	329			
100k	Groups	3343.5	10	334.35	50.05	0.00
	Error	2131	319	6.68		
	Total	5474.5	329			

about parametric distributions' requirements for validity. The only assumptions we made is that our samples are random and representative.

3.4 The Approximate Randomization Test

This test allow us to test the null hypothesis that the means of α-groups are equal. Having two samples of individuals S_A and S_B formed by the BLEU metrics for the translation of each problem in the test set, the test statistic for a pairwise comparison of two means is given by the expression:

$$\theta = \sum_{i=1}^{N} (a_i - b_i)/N \text{ for } a_i \in S_A \text{ and } b_i \in S_B \tag{3}$$

where N is the number of translation problems given to each system (30).

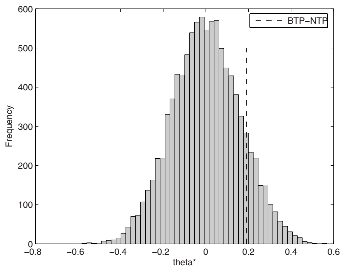
In a randomization test, we randomly shuffle the sign of individual differences to get a sampling distribution of θ^* which is a pseudo-statistic.

Running 9999 iterations of this randomization test for the pairs (Non-TP,Best-TP), (Non-TP,Only-TP) and (Best-TP,Only-TP) for k=80k, we get the $\theta*$ distributions displayed in Fig.5.

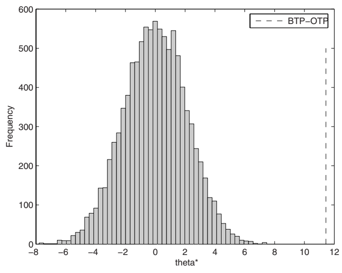
From these distributions we can take the probability of $P(\theta^* \geq \theta)$ which is displayed in the Tab. 3.

Using these results, we can run the following hypothesis: $H_0 : \theta = 0$ using the alternative hypothesis of $H_1 : \theta > 0$ for each one, using the probability just obtained as the p-value.

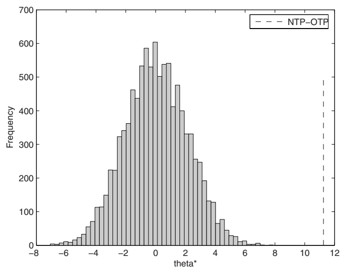
As we can see, there is strong evidence that suggests that the means of Best-TP and Non-TP groups are greater than the mean of the Only-TP group. But



(a)



(b)



(c)

Fig. 5. Sampling distributions for the paired-sample comparisons of: (a) Best-TP vs Non-TP, (b) Best-TP vs Only-TP and (c) Non-TP vs Only-TP

Table 3. Probabilities of $p(\theta^* > \theta)$ for 80k

comparison	$p(\theta^* > \theta)$
Best-TP vs Non-TP	0.1112
Best-TP vs Only-TP	0.0001
Non-TP vs Only-TP	0.0001

there is not enough evidence that can ensure that the means of Non-TP and Best-TP groups are not equal at a significance level of 5%. For this last comparison, we should then keep the null hypothesis.

3.5 Summarized Comparisons

In the Tab.4 we show the p values for every pair and every k-size.

From this table, we observe that for k-groups 10k, 20k and 40k the Best-TP systems perform better than the Non-TP systems. Nevertheless as the size of training data increases (80k ,100k) such improvements are not significant. Therefore we can say that mixed TP systems lead to significant improvements in translation quality when training resources are scarce.

Table 4. Summarized results for pairwise comparisons presented by comparison and k-group

	Non-TP vs Best-TP		Best-TP vs Only-TP		Non-TP vs Only-TP	
size-k	θ	pvalue	θ	pvalue	θ	pvalue
10k	0.6003	0.0001	10.87	0.0001	10.268	0.0001
20k	1.1817	0.0000	10.71	0.0001	9.52	0.0001
40k	1.1	0.0000	11.38	0.0001	10.27	0.0001
80k	0.1910	0.1112	11.443	0.0001	11.252	0.0001
100k	0.121	0.1116	11.325	0.0001	11.204	0.0001

3.6 Best Practical Bound

In figure 6, we observe the average BLEU for groups Non-TP and Best-TP as well as the best practical bound (BPB). The BPB is the average of the best scores for each of the translation problems on the experiment at every k-group. The BPB allows us to detect ceiling effects (very hard problems that might obscure results).

This graph helps us to notice that at low k-values the Best-TP is close to the BPB but the differences with the Non-TP are significant so we can conclude that the Best-TP is the system that performs the best under almost every problem. Nevertheless, as k-size increases Non-TP and Best-TP become closer to each other, but distant from the BPB. This suggests that both could be performing better. Therefore we can conclude that no ceiling effect was observed and thus our results hold valid.

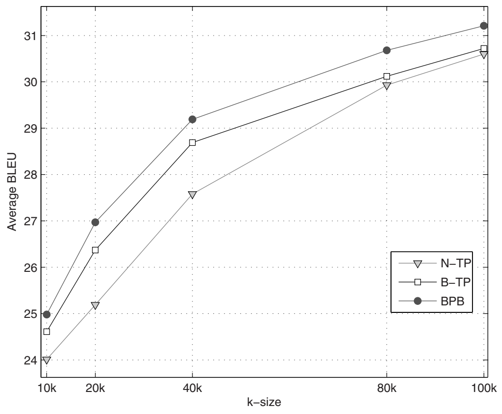


Fig. 6. Average BLEU vs. k-size for Best-TP, Non-TP and the Best Practical Bound (BPB)

4 Conclusions and Future Work

In this paper we analyzed the results obtained from using the translation paraphrases proposed by [10]. From our experiments we can draw the following conclusions:

1. As we increase the size of training corpora, we observe that the best translations are found at lower alphas, suggesting that for large training corpora, TPs have a lower impact.
2. For small training sizes, there is evidence that suggests that there is a significant improvement in translation quality by the utilization of TPs but at larger levels, there is no statistical evidence that suggest that a system's performance is affected by TPs. Therefore we can conclude that TPs bring significant improvements when dealing with scarce data.
3. TPs by themselves produce poor translations. Therefore they should not be used alone, but merged into phrase-tables of Non-TP systems.
4. There was no evidence that showed that we ran into a ceiling effect.

To assess the feasibility of using TPs as a translation aid, we need to test their translation quality improvements when dealing with scarce data. That is to test the improvements from merging small-corpus-trained Non-TP phrase-tables with large-corpus-trained Only-TP phrase-tables to verify whether the translation quality is bound (or not) to the size of the Non-TP phrase-tables.

Other experiments that are to be done is to assess the benefits of using TPs when addressing out-of-domain translation problems. So far we have been working under the same context: Europarl. It would be interesting to test the performance of TPs when dealing with translation problems from other sources. This could shed some light over the possibility of using TPs as a resource for out-of-domain translations.

Finally, we suspect that TPs' effectiveness is bound to the intermediate language used. In this study we used French as an intermediate between English and Spanish, because it seemed somewhat intuitive (French is related to Spanish and English). Nevertheless this assumption might not be optimal. Therefore an exploratory study to assess which intermediate language performs better for a given pair of languages, using information from contrastive linguistics, will be of great interest.

References

1. Brown, P.F., et al.: The mathematics of statistical machine translation: parameter estimation. Computational Linguistics 19, 263–311 (1993)
2. Koehn, P., Och, F., Marcu, D.: Statistical phrase-based translation. In: Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL), Edmonton, Canada (2003)
3. Zens, R., Ney, H.: Improvements in phrase-based statistical machine translation. In: Proceedings of the Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL), Boston, MA, pp. 257–264 (2004)

4. Callison-Burch, C., Koehn, P., Osborne, M.: Improved statistical machine translation using paraphrases. In: Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Association for Computational Linguistics, Morristown, NJ, USA, pp. 17–24 (2006)
5. Langlais, P., Gotti, F.: Phrase-based smt with shallow tree-phrases. In: Proceedings of the Workshop on Statistical Machine Translation, Association for Computational Linguistics, New York City, pp. 39–46 (2006)
6. Giménez, J., Màrquez, L.: Combining linguistic data views for phrase-based SMT. In: Proceedings of the ACL Workshop on Building and Using Parallel Texts, Ann Arbor, Michigan, Association for Computational Linguistics, pp. 145–148 (2005)
7. Alexandra Birch, M.O., Koehn, P.: Ccg supertags in factored statistical machine translation. In: ACL Workshop on Statistical Machine Translation (2007)
8. Hassan, K.S.H., Way, A.: Supertagged phrase-based statistical machine translation. In: 45th Annual Meeting of the Association for Comp. Linguistics (2007)
9. Vilar, J.M., Vidal, E.: A recursive statistical translation model. In: Proceedings of the ACL Workshop on Building and Using Parallel Texts, Ann Arbor, Michigan, Association for Computational Linguistics, pp. 199–207 (2005)
10. Guzman, F., Garrido, L.: Using translation paraphrases from trilingual corpora to improve phrase-based statistical machine translation: A preliminary report. In: MICAI (2007)
11. Koehn, P.: Europarl: A parallel corpus for statistical machine translation. MT Summit 2005 (2005)
12. Och, F., Ney, H.: Statistical machine translation. In: EAMT Workshop, Ljubljana, Slovenia, pp. 39–46 (2000)
13. Och, F.J.: Minimum error rate training in statistical machine translation. In: Proc. of the Association for Computational Linguistics, Sapporo, Japan (2003)
14. Och, F.J., Ney, H.: Discriminative training and maximum entropy models for statistical machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 295–302 (2002)
15. Koehn, P.: Statistical significance tests for machine translation evaluation. In: EMNLP (2004)
16. Koehn, P., et al.: Moses: Open source toolkit for statistical machine translation. In: Annual Meeting of the Association for Computational Linguistics (ACL), Prague, Czech Republic (2007)
17. Papineni, K., et al.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the Association of Computational Linguistics, pp. 311–318 (2002)
18. Chris Callison-Burch, M.O., Koehn, P.: Re-evaluating the role of bleu in machine translation research. In: EACL (2006)

n-Best Reranking for the Efficient Integration of Word Sense Disambiguation and Statistical Machine Translation

Lucia Specia[1,2], Baskaran Sankaran[2], and Maria das Graças Volpe Nunes[1]

[1] NILC/ICMC - Universidade de São Paulo
Trabalhador São-Carlense, 400, São Carlos, 13560-970, Brazil
lspecia@icmc.usp.br, gracan@icmc.usp.br
[2] Microsoft Research India
"Scientia", 196/36, 2nd Main, Sadashivanagar, Bangalore-560080, India
baskaran@microsoft.com

Abstract. Although it has been always thought that Word Sense Disambiguation (WSD) can be useful for Machine Translation, only recently efforts have been made towards integrating both tasks to prove that this assumption is valid, particularly for Statistical Machine Translation (SMT). While different approaches have been proposed and results started to converge in a positive way, it is not clear yet how these applications should be integrated to allow the strengths of both to be exploited. This paper aims to contribute to the recent investigation on the usefulness of WSD for SMT by using n-best reranking to efficiently integrate WSD with SMT. This allows using rich contextual WSD features, which is otherwise not done in current SMT systems. Experiments with English-Portuguese translation in a syntactically motivated phrase-based SMT system and both symbolic and probabilistic WSD models showed significant improvements in BLEU scores.

1 Introduction

The need for Word Sense Disambiguation (WSD) in Machine Translation (MT) systems has been discussed since the early research on MT back in the 1960's. While MT was primarily addressed by rule-based approaches, the consequences of the lack of semantic disambiguation was already emphasized in DARPA's report by Bar-Hillel [2], which resulted in a considerable reduction in the funding for research on MT that time. Meanwhile, WSD grew as an independent research area, without focusing on any particular application.

With the introduction in the 1990's of the Statistical Machine Translation (SMT) approach [3], it has been taken for granted that SMT systems can implicitly address the sense disambiguation problem, especially by their word alignment and target language models. However, the SMT systems normally consider a very short window as context and therefore lack richer information coming from larger contexts or other knowledge sources.

Parallelly, the WSD scenario has evolved considerably and it is now feasible to extract large amounts of shallow information from corpus and knowledge from linguistic resources, and therefore efficient WSD can be performed on a large scale [17]. As a consequence, the question of whether WSD models with richer contextual features can actually improve SMT performance started to be investigated, which is also the aim of this work.

In what follows, we start by giving a brief overview about previous work on the integration of WSD with SMT (Section 2). We describe the WSD and SMT systems used in our experiments in Sections 3 and 4. We then present the n-best reranking approach proposed for the integration of these systems (Section 5) and finally describe the experiments performed along with the results (Section 6).

2 Related Work

Systematic experiments on the usefulness of WSD for SMT were first brought to the research community by Marine Carpuat and Dekai Wu [5]. The predictions provided by a Chinese WSD system for 20 words are mapped into English translations to be used in a word-based Chinese-English SMT system. The WSD system consists of an ensemble of four classifiers with position sensitive, syntactic and local collocational features. WSD predictions are used as hard constraints to limit the options available to the SMT decoder or to replace the translations found by the SMT system. This method has the disadvantage that WSD system arbitrarily overrides the decoder in selecting appropriate words without any principled mechanism. A negative impact in BLEU score [13] was reported with this approach.

Vickrey et al. [19] experimented with a disambiguation task for French-English translation, with the choices defined as the set of words aligned to the ambiguous word, as given by GIZA++ [11]. A logistic regression classifier with collocational and bag-of-words features is used to predict the translations. Positive results were reported for the translation of 1,859 words in two simulated translation tasks: word translation and blank-filling. However, no experiments were performed with a real SMT system.

Cabezas and Resnik [4] experimented with the integration of a Spanish-English WSD system with a phrase-based SMT system. A Support Vector Machine (SVM) classifier was used with low level contextual features and was trained on senses acquired from a word-aligned (by GIZA++) corpus. The predictions of the classifier were provided as additional alternative translations to the SMT decoder. Experiments with a corpus of 2,000 words resulted in a marginal improvement in BLEU, which did not seem to be statistically significant.

Recently, ([7] and [6]) describe two similar approaches aiming at Chinese-English translation. Chan et al. [7] use a hierarchical SMT system which learns context free grammar rules. WSD examples containing at most two words are extracted from the rules produced by the SMT system. The senses are the possible translations for these sub-phrases as given by the alignment information. An SVM classifier produces a disambiguation model for each sub-phrase using collocations, part-of-speech and bag-of-words as features. At decoding time, translation rules for sub-phrases are matched against the WSD predictions and two WSD features are computed, accounting for the probability of a successful match and the length of the translation. These are added to the SMT model, which is retrained to estimate the feature weights by means of the *Minimum Error Rate Training* (MERT) technique [10]. Experiments showed a significant improvement in BLEU.

Carpuat and Wu [6] use a phrase-based SMT system to extract sub-phrases of any size from the parallel corpus, while training the SMT system. These sub-phrases are then taken as disambiguation examples. During decoding, a feature referring to the probability of a translation matching the WSD prediction is added to the model. MERT is used to estimate the weight of all features. Experiments with several corpora and different evaluation metrics showed consistent improvements in the SMT performance with the integration of the WSD module.

The results of these approaches vary widely: from negative or neutral to significantly positive. Despite some of the positive results we believe that existent approaches either develop limited solutions or redefine the WSD task to an extent that it cannot be regarded as WSD anymore. Some of the key limitations include i) the shallow set of features used for WSD, ii) the way the sense repository is acquired (e.g., monolingual senses mapped into translations) and iii) the way the integration itself is accomplished (hard constraint overriding the SMT system to replace lexical choices, etc.). The shift from the traditional *word* sense disambiguation framework to a *phrase* sense disambiguation can be observed in the most recent approaches: there is not a pre-defined set of senses (or translations) and the basic disambiguation unit is a sub-phrase, instead of a word. This approach seems to be particularly appropriate for translating from languages like Chinese, in which the definition of "word" is not the same as in majority of the other languages. However, it requires a simplified view of the WSD task: since the disambiguation units are sub-phrases (not necessarily syntactically motivated), many knowledge sources cannot be exploited, e.g., syntactic relations or selectional preferences between the ambiguous sub-phrase and the other elements in the context. Additionally, it increases considerably the effort invested in WSD, as all sub-phrases are considered to be semantically ambiguous, which is not always the case. Data sparsity can also be a problem, as a large amount of training data is necessary to produce useful disambiguation models for longer sub-phrases. Finally, using only the translations provided by the word alignment system may be a limitation. For example as shown in [15], particularly for light content verbs, off-the-shelf tools like GIZA++ do not perform well and extra translations given by lexical resources can improve the coverage of senses that are rare in the training data.

Given all these issues, we believe the question on how to effectively integrate these two applications in a way that SMT can benefit from the richer information accessible to the WSD without biasing or overloading the SMT, is yet to be investigated. In this paper we report experiments with a simple and efficient way of integrating WSD predictions in a phrase-based SMT system, by using the n-best reranking approach. Unlike some of the earlier approaches, we define the

WSD approach in a multilingual context, exploiting a bilingual sense repository and knowledge sources in both source and target languages. Also importantly, we retain the traditional definition of WSD, i.e., single word disambiguation and this allows exploiting a rich set of knowledge sources. Moreover, we address only highly ambiguous words, which are considered to be very complex cases for MT.

3 WSD Systems

We use a supervised WSD approach trained on corpora tagged with the English-Portuguese translation of a set of ambiguous words. The focus is on the disambiguation of verbs, as they are usually difficult cases and have significant effect on the disambiguation of other elements in the sentence. More specifically, we concentrate on 10 highly frequent and ambiguous verbs, which were selected in a previous case study: *ask*, *come*, *get*, *give*, *go*, *live*, *look*, *make*, *take* and *tell*.

The sense repository for each verb is defined as the set of all possible translations for it in the corpus. Examples are described through the following set of features extracted from corpus and given by language processing tools or resources: (a) unigrams; (b) content words; (c) part-of-speech tags; (d) syntactic relations with respect to the verb; (e) collocations with respect to the verb; (f) a relative count of the overlapping words in the sense definitions of each sense of the verb and its contextual words; (g) selectional restrictions of the verbs ; (h) phrasal verbs possibly occurring in a sentence; (i) pairs of syntactically related words in the sentence occurring frequently in the corpus; (j) bigrams in the sentence occurring frequently in the corpus; (k) unigrams in the context of the target verb in the Portuguese translation, given by the parallel corpus and (l) collocations of the verb in the Portuguese translation.

We evaluated two types of WSD models: (a) symbolic (rule-based) models produced by an Inductive Logic Programming (ILP) algorithm, as proposed by [16], and (b) probabilistic models produced by the SVM implementation proposed by [8]. Both SVM and ILP have been showing good results in several disambiguation tasks [1]. The output of the symbolic models for a new sentence to be classified is simply the prediction (translation) of the verb in that sentence. The probabilistic models, on the other hand, provide as output a ranking of all the possible predictions, scored according to their probabilities. We believe this kind of output is more appropriate for SMT, since it allows for some uncertainty in the disambiguation process. More specifically, even if the actual translation is not the most probable as given by the WSD system, it will be taken into account (with a lower probability). This is particularly useful given that the WSD system is not 100% accurate, as we describe in Section 5. Moreover, sometimes two or more WSD predictions are "acceptable", but with the symbolic models only one can be considered.

4 Treelet SMT System

We use the Dependency Treelet [14] system for the experiments in this work. Treelet is a statistical, syntactically-motivated phrase-based system in which the translation is guided by treelet translation pairs, where a treelet is a connected subgraph of a dependency tree. In the training phase, the parallel corpus is word-aligned using GIZA++. The source sentences are also parsed using a rule-based dependency parser generating the dependency trees. The dependency tree is superimposed on the word aligned parallel data thereby resulting in aligned parallel dependency tree pairs. These are then split into treelets having up to four words on either sides. The system learns different models such as, word count, phrase count, direct and inverted channel models, order model, language model, etc. During decoding, the source sentence is parsed using the dependency parser and the generated dependency tree is partitioned into treelets with uniform probability distribution. The resultant treelets are then matched against the translation pairs learnt earlier. The target language treelets are finally joined to form a single tree constrained by the order model, language model, etc. Candidate translations t are scored according to a linear combination of feature functions f_j learnt in the training phase, as shown in Eq. 1,

$$score(t) = \sum_j \lambda_j f_j(t) \tag{1}$$

where:

- λ_j: model parameters estimated via MERT
- $f_j(t)$: value of the feature function j on the candidate t

5 Integrating WSD Predictions in the Treelet System

Our approach follows the n-best reranking technique proposed by Och et al. [12], in which new features can be added to a baseline SMT system and combined to the existing ones in a linear framework to select the best scoring candidate translation from an n-best list. The weights of all the feature functions including that of the WSD feature are optimized for BLEU using the MERT. As pointed out by Och et al., the n-best reranking technique allows for rapid experimentation in SMT with any kind of feature, such as long distance contextual features, which would be difficult and costly to be included in the SMT system otherwise.

A theoretical limitation of the n-best list reranking approach is that the possible improvements need to be available in the n-best list. A new feature function will not have the desired impact if the variation favored by it is not present in the n-best list. To overcome this limitation, we also investigate the extended n-best reranking proposed by Toutanova and Suzuki[18], where the n-best list is expanded by other candidates, according to the variations proposed by the new feature. We therefore experiment with both standard and expanded n-best reranking approaches for integration of SMT and WSD.

5.1 Standard n-Best List Reranking

In this method we add to the existing features of the SMT linear model an extra feature corresponding to the WSD prediction. Its value is the log of the probability of the WSD prediction that is found in the n-best candidate sentence for the ambiguous source word, when probabilistic WSD models are used. For the symbolic models, this value will be “1”, if the prediction found in the candidate matches the one proposed by the WSD model, and “0” otherwise. In order to compute this feature, we look for any of the possible translations given by the WSD system in that candidate sentence, in positions that are (at least partially) aligned to the ambiguous source word. The morphological variations of the possible translations (person, tense, number, etc.) are taken into account through the expansion of the list of possible WSD predictions using a Portuguese lexicon [9]. If none of the WSD predictions is found in the candidate sentence, a probability “0” is assigned to such candidate.

MERT is performed on a development set n-best list to re-estimate the weights of the feature functions including that of the WSD feature. These new feature weights are then used to rerank the n-best list of the test data.

For example, consider in Fig. 1 the top-3 candidate translations produced by the baseline SMT system for the sentence s_1 (its reference translation being r_1) in the experiments with the English-Portuguese translation of the verb *ask* (see Section 6).

s_1: He returned and *asked* me if I wanted anything else and whether I had enjoyed my meal.
r_1: Ele voltou, e **perguntou** se eu queria mais alguma coisa, se eu tinha gostado.

Ele voltou, e **pediu-me** se eu queria mais alguma coisa e se eu tinha gostado.
Ele voltou, e **perguntou** se eu queria mais alguma coisa, se eu tinha gostado ontem à noite e.
Ele voltou, e **perguntou** se eu queria mais alguma coisa, se tinha gostado.

Fig. 1. Top-3 n-best candidate translations for s_1 according to the baseline SMT system

The prediction given for this sentence by the symbolic WSD models is “perguntar” (inquire, enquire). It should be noticed that the top-scored sentence in 1 uses a different translation “pedir” (inflected as “pediu-me”) (make a request). The other two candidates, on the other hand, contain the correct prediction according to the WSD system, inflected as “perguntou”. The initial and new weights of the feature functions for the development data are shown in Table 1. The values of the SMT features and that of the WSD feature (using symbolic models) for the top-3 candidate sentences are shown in Table 2. Notice that the WSD feature just takes binary value since this is based on the symbolic model. The table also shows the final score for each candidate sentence in the last column. It can be observed that the third sentence will now get reranked as the best sentence, which is indeed a better translation than the first sentence.

In some cases, the prediction pointed by the symbolic WSD models is not the correct one, and this might mislead the SMT system. There may be also

Table 1. Feature weights before and after MERT

Feature	**Initial weight**	**New weight**
wsd	NA	0.379
word count	2.649	1.463
phrase count	-1.563	-2.585
target model	1	1
channel model mle	1.623	1.776
channel model direct	-0.230	-0.217
channel model inverted	0.197	0.216
template mle	0.756	0.756
template source order count	-0.467	1.791
order model	-0.062	-0.061

Table 2. Feature values and overall scores for the candidate translations in Fig. 1

wsd	word count	phrase count	target model	channel mle	channel direct	channel inverted	template mle	source order count	order model	final score
0	19	7	-79.54	-7.04	-40.11	-52.8	-0.59	1	-12.19	-86.54
1	20	6	-87.98	-8.61	-73.38	-60.12	-0.59	1	-15.52	-87.48
1	16	7	-66.25	-8.16	-33.58	-62.82	-0.95	1	-10.21	-83.22

alternative good options to the top prediction, which are not taken into account by these models, since they only provide the top prediction. To illustrate that the probabilistic classifiers can overcome these disadvantages, consider the sentence s_2, with the ambiguous verb *come*. The correct translation for this word in this sentence should be "chegar" (arrive). However, both ILP and SVM predict "vir" (come up, move forward) as the top translation. The SVM classifier predicted "chegar" and "vir" with probabilities 0.66 and 0.0202, respectively. The top-candidate in the n-best list originally contained the wrong translation "vir" and the first correct candidate occurred only after 10*th* place in the n-best list. Nevertheless, this candidate was reranked as the top candidate after the integration with the probabilistic WSD models. The feature values and overall scores of the top-candidates before (row-1) and after (row-2) reranking are shown in Table 3.

s_2 Simão Botelho had seen his judgment day *come* with unshakeable spirit.
r_2 Simão Botelho vira imperturbável **chegar** o dia do julgamento.

Table 3. Feature values and overall scores for the original and reranked top candidates

wsd	word count	phrase count	target model	channel mle	channel direct	channel inverted	template mle	source order count	order model	final score
-0.41	11	7	-78.05	-7.01	-21.42	-17.87	-1.89	0	-6.57	-92.89
-3.903	14	7	-75.12	-7.73	-66.26	-53.19	0	3	-20.16	-89.19

5.2 Expanded n-Best List Reranking

As explained earlier, the standard n-best reranking method is not effective if a better translation is not found in the n-best list. We believe that expanding the n-best list by generating new candidate sentences having the word predicted the WSD model can overcome this limitation, potentially yielding better results. The assumption, here, is that the WSD model can predict the correct translation, which is not found by the SMT system, since the former has access to much richer resources.

In order to include the most probable translation(s) suggested by the WSD module in the n-best list, for each distinct candidate translation , we generate up to m other candidates, where m corresponds to the number of WSD predictions for that test sentence with a probability above a certain threshold. Notice that in the case of the symbolic models, m can be at most $= 1$, that is, if the prediction found in the candidate sentence is not "the" one pointed by the symbolic WSD model, a single additional candidate will be generated containing that prediction.

The same procedure as in the first integration method is used to verify whether the candidate translation contains the top WSD predictions. If the original candidate already contains one of the m predictions, *m-1* copies of that candidate are generated by replacing the translation found by each of the other *m-1* possible predictions (unless these are already in the n-best list or in the expanded n-best list). The morphological features of the translation provided by the SMT system in the original sentence are used to inflect the remaining WSD predictions. This information is extracted from the Portuguese lexicon.

If the candidate sentence does not contain any of the possible WSD predictions for the ambiguous source verb in a position aligned to it, but contains a different verb, m candidates are generated for the m WSD predictions, keeping the morphological features of the verb found in the original candidate. Here the aligned position in the candidate sentence must contain one and only one verb, otherwise it is not possible to identify, without any additional information, which verb should be replaced by the WSD prediction in the new candidates. This is a reasonable assumption for English-Portuguese translation, as most of the verbs, even those occurring in phrasal expressions, are translated as a single word. Finally, if no verb is found in a candidate sentence in the position aligned to the ambiguous source verb, only the original sentence if kept.

The new n-best list for each test sentence will have up to *n+(n*m)* candidate translations. This number is usually not very high because of the threshold used for eliminating weak WSD predictions in m.

The value for the WSD feature is computed as in the standard method, for both symbolic and probabilistic WSD models for each candidate translations in the expanded n-best list. It is important to notice that while some SMT feature values in the newly generated candidate translations remain the same, such as word and phrase counts, others may need to be recomputed. As is it too costly to recompute all of them, we only update the **language model** and **direct** and **reverse word-alignment models**.

Again, a development set n-best list is used to estimate the weights of the feature functions including that of WSD via MERT and these weights are used to rerank the expanded n-best list for each test sentence.

It is important to notice that we are not simply replacing the existing translations for the ambiguous verbs in the n-best list by the top WSD predictions. Instead, all the candidates provided by the SMT system are kept in the n-best list, and new candidates are added to this list.

6 Experiments and Results

We experimented with the 10 highly frequent and ambiguous verbs mentioned in Section 3. To build the WSD system, a corpus containing 5000 sentences for each verb was constructed from randomly selected data of different domains and genres, including the Bible, literary fiction, European Parliament proceedings and a mixture of smaller sources. This corpus was automatically annotated with the translation of the verb using a tagging system based on parallel corpus, statistical information and translation dictionaries, as proposed by [15].

When evaluated independently with 80% of the corpus for training (4K) and the remainder for test (1K), the symbolic WSD models achieved an average accuracy of 0.74, while the top prediction of the probabilistic models achieved 0.69. Both accuracies are considerably higher than a baseline system which always vote for the majority translation (0.5).

The data used is SMT experiments is presented in Table 4. The training data consists of a superset of the data used to train the WSD system, extracted from the same sources, mostly from the European Parliament proceedings. Devset-1 corresponds to the sentences drawn randomly from the same sources and is used to tune the feature weights of the SMT system. The sentences in Devset-2 are the same used to train the WSD system and hence contain at least one ambiguous verb in every sentence. This set is used to estimate the WSD feature weight and the new weights of the SMT features. The sentences in Test-set are those used to test the WSD system and therefore also contain one of the ambiguous verbs. This was chosen as test set so that we could use the already existing predictions provided by the WSD systems.

We experimented with both standard and expanded n-best reranking approaches and the results of the experiments are summarized in Table 6. In the case of the probabilistic WSD models, a threshold of 0.1 was used to filter weak

Table 4. Data for SMT

	Sentence pairs
Training	700 K
Dev-set 1	4 K
Dev-set 2	4 K
Test-set	1 K

Table 5. BLEU scores for the experiments with 1,000 sentences containing the 10 verbs. The **Baseline SMT** results refer to the SMT system without the WSD feature, while **Method 1** refers to the standard n-best reranking and **SMT + WSD_2** to the expanded n-best reranking, both with the addition of the WSD feature provided by **ILP** or **SVM** disambiguation models.

WSD model	Baseline SMT	Method 1	Method 2
ILP	0.3248	0.34	0.3404
SVM	0.3248	0.35	0.3501

WSD predictions for the expanded n-best reranking method. The number of candidate translations in the expanded n-best list increased from 97, 140 to 125, 572, for symbolic WSD models) or 214, 311 (probabilistic models).

As shown in Table 6, both WSD models result in significant increase in BLEU (paired t-test with $p < 0.05$), compared to the baseline system: 1.5 and 2.5 BLEU points in absolute terms. As we expected, probabilistic WSD models yielded better results than the symbolic models, as they allow for some uncertainty in the disambiguation process. Surprisingly, the expanded n-best reranking method did not yield better than the standard reranking method in terms of BLEU, although we could notice that many of the top translations were not available in the original n-best list. Comparing the reranked n-best list for this method to that of the standard method, we found that the former does contain different sentences in many cases. However, the variations occur mostly in a single word, i.e., the translation of the ambiguous verb, and this does not have a significant impact in BLEU scores. It is important to remember that these sentences were artificially generated by replacing the translation in the original candidate by the one suggested by the WSD models, the rest of the sentence remaining the same. In the standard n-best reranking, on the other hand, when changes occur with resect to the baseline candidate sentence, they usually implicate in a considerably different candidate sentence, therefore noticeable to BLEU.

7 Conclusions

So far, little work has been done on integrating Word Sense Disambiguation and Machine Translation systems. The current approaches suffer from several shortcomings. Basically, they either develop limited solutions or redefine the WSD task in a way that it can no longer be regarded as WSD. In contrast, we use a method for integration which exploits the strengths of rich knowledge sources within the framework of traditional WSD and at the same time functions in synchrony with the SMT without either overriding or biasing it. We presented experiments with symbolic and probabilistic WSD systems and experimented with 10 verbs that are highly ambiguous: on average, they contain 21 possible translations in a corpus of 500 sentences per verb. We also considered two methods of integration: standard n-best and expanded n-best reranking. We demonstrated improvements in BLEU scores for both methods and WSD classifiers.

The proposed approach is a generic one and can be used with any SMT system that allows combining feature functions in a log-linear framework.

As future work we plan to show that the WSD model can also improve SMT scores for real-time texts, which may not have the ambiguous verbs in it. It is important to emphasize, however, that these verbs are highly frequent in English texts across different domains and genres (see http://www.kilgarriff.co.uk/bnc-readme.html). In different random selections of $1,000$ sentences from our corpus, we found that, on average, approximately 300 sentences contain one of the 10 verbs. Experiments with these random test sets will be performed.

Additionally, a systematic (manual) analysis of the reordered list of translations needs to be done, as BLEU is not very sensitive to such lexical variations, which reflet more in the adequacy of the sentences, instead of its fluency.

References

1. Agirre, E., Màrquez, L., Wicentowski, R.: Proceedings of SemEval-2007 - the Fourth International Workshop on Semantic Evaluations, Prague (2007)
2. Bar-Hillel, Y.: The Present Status of Automatic Translations of Languages, 91–163 (1960)
3. Brown, P.F., et al.: The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics 19(2) (1993)
4. Cabezas, C., Resnik, P.: Using WSD Techniques for Lexical Selection in Statistical Machine Translation. UMIACS Technical Report UMIACS-TR-2005-42 (2005)
5. Carpuat, M., Wu, D.: Word Sense Disambiguation vs. Statistical Machine Translation. In: 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005), Ann Arbor, pp. 387–394 (2005)
6. Carpuat, M., Wu, D.: Improving Statistical Machine Translation Using Word Sense Disambiguation. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-2007), Prague, pp. 61–72 (2007)
7. Chan, Y.S., Ng, H.T., Chiang, D.: Word Sense Disambiguation Improves Statistical Machine Translation. In: 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007), Prague, pp. 33–40 (2007)
8. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines (2001), Software available at `http://www.csie.ntu.edu.tw/cjlin/libsvm`
9. Nunes, M.G.V., et al.: The design of a Lexicon for Brazilian Portuguese: Lessons learned and Perspectives. In: II Workshop on Computational Processing of Written and Speak Portuguese (Propor), Curitiba, pp. 61–70 (1996)
10. Och, F.J.: Minimum error rate training in statistical machine translation. In: 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003), Sapporo, pp. 160–167 (2003)
11. Och, F.J., Ney, H.: Improved statistical alignment models. In: 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000), Hong Kong, pp. 440–447 (2000)
12. Och, F.J., et al.: A Smorgasbord of Features for Statistical Machine Translation. Human Language Technology / North American Chapter of the Association for Computational Linguistics (HLT/NAACL-04), Boston, pp. 161–168 (2004)

13. Papineni, K., et al.: BLEU: a method for automatic evaluation of machine translation. In: 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002), Philadelphia, pp. 311–318 (2002)
14. Quirk, C., Menezes, A., Cherry, C.: Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In: 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005), Ann Arbor, pp. 271–279 (2005)
15. Specia, L., Nunes, M.G.V., Stevenson, M.: Exploiting Parallel Texts to Produce a Multilingual Sense-tagged Corpus for Word Sense Disambiguation. Recent Advances in Natural Language Processing (RANLP-2005), Borovets, pp. 525–531 (2005)
16. Specia, L., Stevenson, M., Nunes, M.G.V.: Learning Expressive Models for Word Sense Disambiguation. In: 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007), Prague, pp. 41–48 (2007)
17. Stevenson, M., Wilks, Y.: The Interaction of Knowledge Sources for Word Sense Disambiguation. Computational Linguistics 27(3), 321–349 (2001)
18. Toutanova, K., Suzuki, H.: Generating Case Markers in Machine Translation. Human Language Technology / North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2007), Rochester, pp. 49–56 (2007)
19. Vickrey, D., et al.: Word-Sense Disambiguation for Machine Translation. Joint Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP-2005), Vancouver, pp. 771–778 (2005)

Learning Finite State Transducers Using Bilingual Phrases[⋆]

Jorge González, Germán Sanchis, and Francisco Casacuberta

Instituto Tecnológico de Informática
Universidad Politécnica de Valencia
{jgonzalez,gsanchis,fcn}@dsic.upv.es

Abstract. Statistical Machine Translation is receiving more and more attention every day due to the success that the phrase-based alignment models are obtaining. However, despite their power, state-of-the-art systems using these models present a series of disadvantages that lessen their effectiveness in working environments where temporal or spacial computational resources are limited. A finite-state framework represents an interesting alternative because it constitutes an efficient paradigm where quality and realtime factors are properly integrated in order to build translation devices that may be of help for their potential users. Here, we describe a way to use the bilingual information in a phrase-based model in order to implement a phrase-based ngram model using finite state transducers. It will be worth the trouble due to the notable decrease in computational requirements that finite state transducers present in practice with respect to the use of some well-known stack-decoding algorithms. Results for the French-English EuroParl benchmark corpus from the 2006 Workshop on Machine Translation of the ACL are reported.

1 Introduction

Machine translation (MT) is an important area of Information Society Technologies in different research frameworks of the European Union. While the development of a classical MT system requires a great human effort, *Statistical machine translation* (SMT) has proved to be an interesting framework due to being able to automatically build MT systems if adequate parallel corpora are provided [1].

Given a sentence $\mathbf{s}$ from a source language, it is commonly accepted that a convenient way to express the SMT problem is through the Bayes' rule [1]:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t}|\mathbf{s}) = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t}) \cdot \Pr(\mathbf{s}|\mathbf{t}) \tag{1}$$

where $\hat{\mathbf{t}}$ stands for the most likely hypothesis, according to the model, from all the possible output sentences $\mathbf{t}$. $\Pr(\mathbf{t})$ is frequently approached by a *language*

[⋆] This work has been partially supported by the EC (FEDER) and the Spanish projects TIN2006-15694-C02-01 and the Consolider Ingenio 2010 CSD2007-00018.

model, which assigns high probability to well formed target sentences, and $\Pr(\mathbf{s}|\mathbf{t})$ is modelled by a *translation model* that is based on stochastic dictionaries and alignment models [2,3].

Under this framework, the so called *phrase-based* models [5,6] have proved to provide a very natural framework for SMT. Computing the translation probability of a given *phrase*, i.e. a sequence of words, and hence introducing information about context, these SMT systems seem to have mostly outperformed single-word models, quickly evolving into the predominant technology in the state-of-the-art [15]. However, despite their power, current phrase-based decoders present some disadvantages that hinder their professional use as translation applications.

Nowadays, SMT results are still far from what could be considered as acceptable, when real translation tasks of certain complexity are tackled. Therefore, human supervision is always needed in order to correct the output of the systems [24]. In this context, a realtime working environment is crucial for the success of a SMT system as a useful tool for their users.

On the other hand, phrase-based models are generally extremely large to be loaded into memory at decoding time, then they have to be filtered using the whole test set, which must be previously known before starting to translate. In other words, a spontaneous translation application, where the user is free to propose any arbitrary task-dependent source sentence, is not allowed in the current phrase-based SMT technologies.

Moreover, to our knowledge, the acoustic model integration into phrase-based models is not possible. Therefore, they can never be employed in an integrated architecture of speech translation, together with a speech recognition system.

An alternative to Equation 1 is to perform a different transformation:

$$\hat{\mathbf{t}} = \underset{\mathrm{t}}{\operatorname{argmax}} \Pr(\mathbf{t}|\mathbf{s}) = \underset{\mathrm{t}}{\operatorname{argmax}} \Pr(\mathbf{s},\mathbf{t}) \tag{2}$$

In this case, the joint probability distribution can be adequately modelled by means of *stochastic finite-state transducers* (SFST) [7,8]. These models can deal with some word order issues together with the relation between input and output vocabularies [9,14]. Finite state transducers represent an efficient framework for translation purposes, where realtime applications can be successfully implemented. Moreover, they are equally suitable either for text-input or speech-input purposes, therefore allowing for an integrated architecture of speech translation.

In this work, we describe how the segments in a phrase-based model can be properly incorporated into a finite-state transducer that also takes into account some language model abilities. This can be successfully accomplished by means of the concept of *monotonous bilingual segmentation*. Phrase-based transducers have already shown a significative improvement with respect to their primitive word-based topologies [20], which are exclusively based on statistical alignments.

This framework adaptation causes the model to be simpler, then it can be fully allocated into memory. That allows an autonomous performance where the system does not have to know a-priori which are the sentences in the test set.

Results for the French-English EuroParl benchmark corpus from the 2006 Workshop on Machine Translation of the ACL [17] are reported. After the transfer from phrase-based models to finite states, a minor underperformance must be acceptable (due to the parameter reduction of the model), but that becomes negligible because of all the benefits that a finite state framework presents in practice, especially its successful application to realtime working environments.

The organization of this document is as follows: next section presents a review of phrase-based models; section 3 introduces the finite state framework, including the purpose of this article, that is, the use of the bilingual tuples in a phrase-based model to implement a phrase-based ngram model using finite state transducers; the experimental setup and results are described in section 4; and, finally, conclusions and further work are briefly summed up at the last section.

2 Phrase-Based Models

The derivation of phrase-based translation models stems from the concept of bilingual segmentation, i.e. sequences of source words and sequences of target words. It is assumed that only segments of contiguous words are considered, the number of source segments being the number of target segments (say K) and each source segment being aligned with only one target segment and vice versa.

A monotonicity constraint would allow for an exportation of translation models into a finite state paradigm since monotone translation models and finite state transducers are closely related models. Let I and J be the length of $\mathbf{t}$ and $\mathbf{s}$ respectively[2]. Then, the bilingual segmentation is formalised through two segmentation functions: μ for the target segmentation ($\mu_1^K : \mu_k \in \{1, \ldots, I\}$ & $\mu_k \geq \mu_{k-1}$ for $1 \leq k \leq K$ & $\mu_K = I$ $(\mu_0 = 0)$) and γ for the source segmentation ($\gamma_1^K : \gamma_k \in \{1, \ldots, J\}$ & $\gamma_k \geq \gamma_{k-1}$ for $1 \leq k \leq K$ & $\gamma_K = J$ $(\gamma_0 = 0)$). Since we will only be considering monotonous segmentations, we can establish that, for all k, the segments $s_{\gamma_{k-1}+1}^{\gamma_k}$ and $t_{\mu_{k-1}+1}^{\mu_k}$ must be uniquely aligned to each other.

By assuming that all possible segmentations of $\mathbf{s}$ in K phrases and all possible segmentations of $\mathbf{t}$ in K phrases have the same probability independent of K, then $\Pr(\mathbf{s}|\mathbf{t})$ can be written as:

$$\Pr(\mathbf{s}|\mathbf{t}) \propto \sum_{K} \sum_{\mu_1^K} \sum_{\gamma_1^K} \prod_{k=1}^{K} \Pr(s_{\gamma_{k-1}+1}^{\gamma_k} | t_{\mu_{k-1}+1}^{\mu_k}) \tag{3}$$

2.1 Learning Phrase-Based Models

Ultimately, when learning a phrase-based translation model, the purpose is to compute a *phrase translation table*, in the form of

$$s_j \ldots s_{j'} \,|||\, t_i \ldots t_{i'} \,|||\, \Pr(s_j \ldots s_{j'} | t_i \ldots t_{i'})$$

[2] Following a notation used in [2], a sequence of the form $z_i, \ldots, z_j$ is denoted as z_i^j. For some positive integers N and M, the image of a function $f : \{1, ..., N\} \rightarrow \{1, ..., M\}$ for n is denoted as f_n, and all the possible values of the function as f_1^N.

where the first column represents the input (source) phrase, the second column represents the output (target) phrase and the last column is the probability assigned by the model to the given phrase pair.

In the last years, a wide variety of techniques to produce phrase-based dictionaries have been researched and implemented [16]. Firstly, a direct learning of the parameters of the distribution $\Pr(s_j^{j'}|t_i^{i'})$ under the maximum likelihood framework by using a sentence-aligned corpus was proposed [5]. At the same time, heuristics for extracting all the possible segmentations being coherent with a word-aligned corpus [6,23] (where the alignments were learnt by means of the GIZA++ toolkit [4]) were also proposed. In parallel, an important part of the research community has suggested to focus the phrase extraction procedure on linguistically motivated word sequences, which are sometimes called *chunks*. Under these approaches, only word sequences that fulfil certain linguistically motivated rules, such as e.g. being a subtree of a syntax tree, are considered.

In this work, the training of phrase-based translation models was performed through some heuristic, (word) alignment-based, phrase extraction algorithms.

2.2 Decoding with Phrase-Based Models

Once a SMT system has been trained, a decoding algorithm is needed. Different search strategies have been suggested to define the way in which the search space is organised. Some authors [11,12] have proposed the use of an $A^\star$ algorithm, which adopts a *best-first* strategy that uses a stack (priority-queue) in order to organise the search space. On the other hand, a *depth-first* strategy was also suggested in [10], using a set of stacks to perform the search.

3 Finite State Framework

In general, despite being considered *state-of-the-art*, the practical use of most phrase-based SMT systems as translation devices is very constrained. Although they are getting the best translation rates, they do so by using a huge quantity of computational resources. Phrase-based translation models estimate so many parameters that their handling during decoding time makes them perform at a very small speed rate. These conditions are not allowed in realtime environments.

On the other hand, a finite state framework represents an interesting alternative because it constitutes an efficient paradigm where quality and realtime factors are properly integrated in order to build translation devices that may be really helpful for their potential users.

Due to the discrete success that MT systems are obtaining when real translation tasks of certain complexity are tackled, interactive working environments have become very popular. In the last years, Computer Assisted Translation (CAT) has been revealed as a powerful interface between human beings and MT systems [24]. Needless to say that a realtime performance is required for a productive utilization of the CAT framework.

3.1 Finite State Models

A stochastic finite-state automaton is a tuple $\mathcal{A} = (\Gamma, Q, i, f, P)$, where Γ is an alphabet of symbols, Q is a finite set of states, functions $i : Q \rightarrow [0, 1]$ and $f : Q \rightarrow [0, 1]$ refer to the probability of each state to be, respectively, initial and final, and parcial function $P : Q \times \{\Gamma \cup \lambda\} \times Q \rightarrow [0, 1]$ defines a set of transitions between pairs of states in such a way that each transition is labelled with a symbol from Γ (or the empty string λ), and is assigned a probability. An example of a stochastic finite-state automaton can be observed in figure 1. Moreover, consistency properties have to be respected for functions i, f and P in order to be able to define a distribution of probabilities on the free monoid.

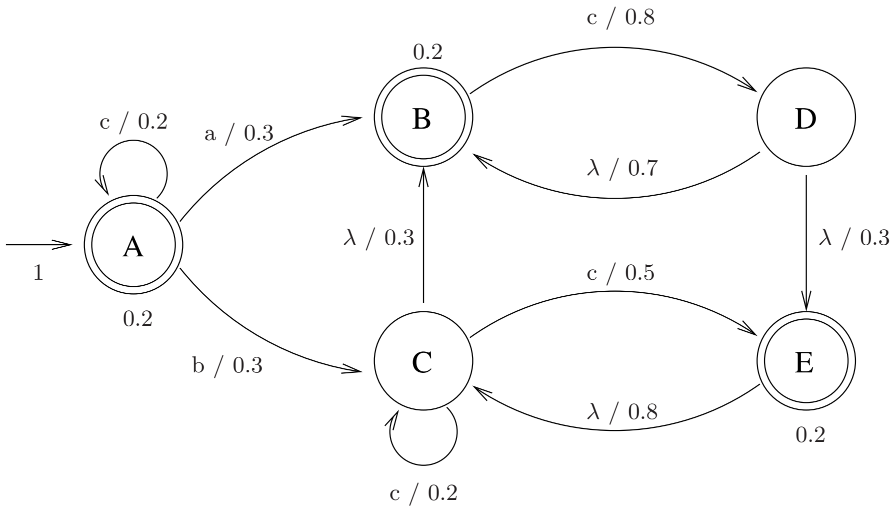


Fig. 1. A stochastic finite-state automaton

A stochastic finite-state transducer is defined similarly to a stochastic finite-state automaton, with the difference that transitions between states are labelled with pairs of symbols that belong to the cartesian product of two different (input and output) alphabets, Σ and Δ. Then, given some input/output strings $\mathbf{s}_1^J$ and $\mathbf{t}_1^I$, a stochastic transducer is able to associate them a joint probability $\Pr(\mathbf{s}_1^J, \mathbf{t}_1^I)$.

3.2 Inference of Stochastic Transducers

The GIATI paradigm [9,21] has been revealed as an interesting approach to infer stochastic finite-state transducers through the modelling of languages. Rather than learning translations, GIATI first converts every pair of parallel sentences in the training corpus into a corresponding extended-symbol string in order to, straight afterwards, infer a language model from. The ideas of GIATI, which were introduced in [9,21], were later developped in the framework of log-linear models [22].

More concretely, given a parallel corpus consisting of a finite sample C of string pairs: first, each training pair $(\bar{x}, \bar{y}) \in \Sigma^\star \times \Delta^\star$ is transformed into a string $\bar{z} \in \Gamma^\star$ from an extended alphabet, yielding a string corpus S; then, a stochastic finite-state automaton $\mathcal{A}$ is inferred from S; finally, transition labels in $\mathcal{A}$ are turned back into pairs of strings of source/target symbols in $\Sigma^\star \times \Delta^\star$, thus converting the automaton $\mathcal{A}$ into a transducer $\mathcal{T}$.

The first transformation is modelled by some labelling function $\mathcal{L} : \Sigma^\star \times \Delta^\star \rightarrow \Gamma^\star$, whereas the last transformation is defined by an inverse labelling function $\Lambda(\cdot)$, such that $\Lambda(\mathcal{L}(C)) = C$. Building a corpus of extended symbols from the original bilingual corpus allows for the use of many useful algorithms for learning stochastic finite-state automata (or equivalent models) that have been proposed in the literature about grammatical inference.

3.3 Training Phrase-Based Transducers

Some related work exists, where a phrase-based methodology is also employed in a finite state framework [14]. There, a generative translation process, which is composed of several transduction models, is applied. Each constituent distribution of the model, including some well-known aspects in SMT such as phrase reordering or spurious word insertion, is implemented as a finite state transducer. The GIATI paradigm, however, tries to merge all these operations into only one transduction model.

GIATI, as defined in subsection 3.2, is a general framework for designing transducer inference algorithms. Let us describe a phrase-based algorithm following this paradigm.

As seen in section 2.1, a phrase-based translation model constitutes a probabilistic dictionary of bilingual segments. Intuitively, all these phrases have been obtained from the computation of all the possible bilingual segmentations of the training corpus. Nevertheless, for a strict application of the GIATI paradigm, only one segmentation per sentence should be taken into account. This requirement can be efficiently approximated by translating[3] the source-training sentences by means of a phrase-based SMT system, since phrase-based decoding implies looking for the best segmentation.

Then, the string corpus to be modelled by a finite state automaton is composed of the sequences of phrase pairs that best fit every source-training sentence, according to the SMT system. This can be effectively seen as a reduction of the phrase translation table by selecting only those phrase pairs that correspond to the most likely segmentation (and translation) of each source sentence in the training corpus. Our phrase-based exportation scheme is depicted in figure 2.

To explain it more clearly, we will use a tiny example of a source sentence:

On demande une activité pour la mis en pratique

[3] Search must be constrained to find a monotonous solution.

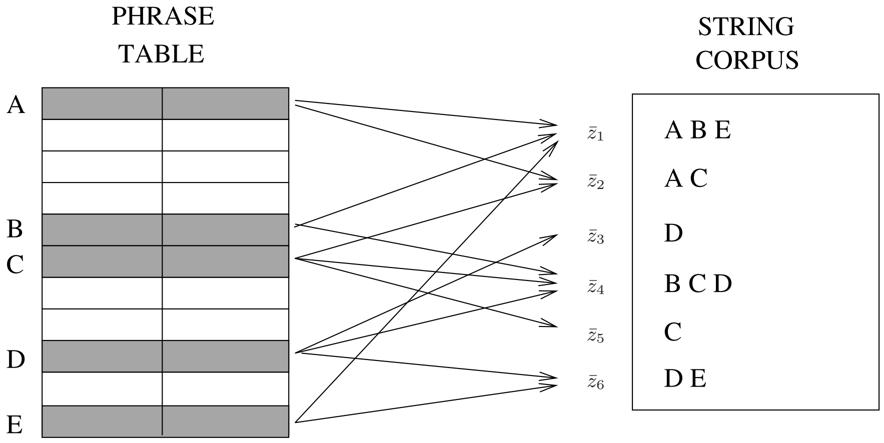


Fig. 2. Using phrase-based translation models in the GIATI paradigm

Let us assume that the phrase pairs that best fit, according to a monotonous MT system that employs a phrase-based dictionary $\mathcal{M}$ as translation model, are:

On demande une activité	*Action is required*
pour	*in order to*
la mis en pratique	*implement fully*

Now, the GIATI steps can be concreted as follows:

1. Obtaining a string corpus: for each source sentence $\bar{x}$ in the sample, the composed string is a sequence of $K_{\bar{x}}$ pairs, $(\bar{u}_i, \bar{v}_i) \in \mathcal{M}$, where $\bar{u}_1\bar{u}_2 \ldots \bar{u}_{K_{\bar{x}}} = \bar{x}$. Each of these pairs is considered to be *a single symbol.* Applying this algorithm to the segmentation of our example would produce the following string of three compound symbols:
   $S = \{$(On demande une activité, Action is required)
   (pour, in order to) (la mis en pratique, implement fully)$\}$
2. Inferring a finite-state automaton: a smoothed n-gram model can be inferred from the corpus of strings that was obtained in step 1. Such a model can be expressed in terms of a stochastic finite-state automaton [13].
3. Undoing transformations: a transducer can be obtained by considering every compound symbol not as a single token, but as the bilingual pair of input/output phrases that constitute the label of a transition in a transducer. A detailed description about the expansion of phrase-based transitions into their constituent words are given in [20]. Here, the application of the inverse labelling function to a transition from our example can be seen in figure 3.

Basically, the algorithm produces a transducer which includes a smaller amount of translation information (i.e., not all the phrases in $\mathcal{M}$) but keeping information about the order in which they might appear.

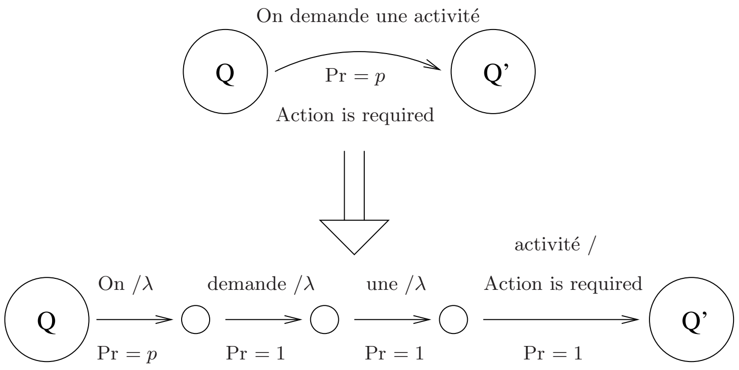


Fig. 3. A phrase-based inverse labelling function

3.4 Phrase-Based Finite State Decoding

Given an input sentence, the best output hypothesis is the one which corresponds to a path through the transduction model that, with the highest probability, accepts the input sequence as part of the input language of the transducer. Although the navigation through the model is constrained by the input sentence, the search space can be extremely large. As a result, only those hypotheses with the highest scores are considered as possible candidates to become the solution.

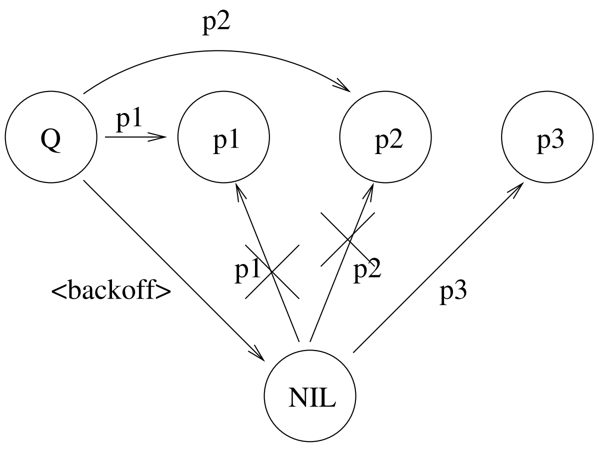


Fig. 4. Compatible transitions for a phrase-based bigram model. Given a reaching state Q, let us assume that the phrases *p1*, *p2* and *p3* are all compatible with the portion of the input sentence that has not been parsed yet. However, the bigram (Q, *p3*) did not occur throughout the training corpus, therefore there is no a direct transition from Q to *p3*. A backoff transition enables the access to *p3* because the bigram (Q, *p3*) turns into a unigram event that is actually inside the model. Unigram transitions to *p1* and *p2* must be ignored because their corresponding bigram events were successfully found one level above.

Table 1. Characteristics of the Fr-En EuroParl

		French	**English**
Training	No. of sentences	688,031	
	Running words	15.6 M	13.8 M
	Vocabulary	80,348	61,626
Dev-Test	No. of sentences	2,000	
	Running words	66,200	57,951

This search process is very efficiently carried out by a slightly adapted version of the well known beam-search Viterbi algorithm.

The system tries to translate several consecutive input words as a whole phrase, always allowing a backoff transition in order to cover all the compatible phrases in the model, not only the ones which have been seen after a given history, but after all its suffixes as well. One more constraint has to be included into the parsing algorithm: any directly reaching state Q' is unable to be reached through a path between Q and Q' that contains a backoff transition. Figure 4 shows a parsing example over a finite-state representation of a bigram model. More details about phrase-based finite state decoding can be found in [20].

4 Experiments

The experimentation was applied to the French-English EuroParl corpus, the benchmark corpus of the NAACL-2006 shared task of the Workshop on Machine Translation of the ACL.

The EuroParl corpus is built on the proceedings of the European Parliament, which are published on its web and are freely available. Because of its nature, this corpus has a large variability and complexity, since the translations into the different official languages are performed by groups of human translators. The fact that not all translators agree in their translating criteria implies that a given source sentence can be translated in various different ways throughout the corpus.

Since the proceedings are not available in every language as a whole, a different subset of the corpus is extracted for every different language pair, thus evolving into somewhat different corpora for each pair.

In the NAACL-2006 Workshop on Machine Translation, a shared task involving, among others, an EuroParl subcorpus for French-English, was proposed. This is the corpus that we have chosen for our experiments. The characteristics of this corpus can be seen in Table 1.

4.1 System Evaluation

We evaluated the performance of a SMT system by using the following evaluation measures:

WER *(Word Error Rate)*: The WER criterion calculates the minimum number of editions (substitutions, insertions or deletions) that are needed to convert the system hypothesis into the sentence considered ground truth. Because of its nature, this measure is very pessimistic.

BLEU *(Bilingual Evaluation Understudy) score*: This indicator computes the precision of unigrams, bigrams, trigrams, and tetragrams with respect to a set of reference translations, with a penalty for too short sentences [18]. BLEU measures accuracy, not error rate.

Speed : This factor is computed as the ratio between the number of running words in the test set and the required time (in seconds) for translation, without considering loading times. So, it is expressed in words per second.

Size : Space requirements for every modelling framework can be estimated by means of the number of phrase pairs in their respective translation models.

4.2 Results

This approach has been tested on a particular phrase-based translation tool: Moses [19]. Moses is a statistical machine translation system that allows you to automatically train phrase-based translation models for any language pair, when provided with a collection of translated texts (a parallel corpus).

Two different SMT systems were compared: one was built using exclusively the Moses toolkit, which we took as reference for comparison purposes; and the other one, which was constructed through the transfer of the corresponding Moses translation model into a finite state framework, under the implementation of the GIATI paradigm that has been described in Section 3.3. The translation results together with their computational capacities can be analysed in Table 2.

Table 2. Translation results and computational requirements. Speed is measured in words per second; Size refers to the vocabulary of phrase pairs.

	Moses	Moses => GIATI
WER	58.0	59.0
BLEU	30.1	29.3
Speed	2.4	198.2
Size	1,372,464	93,158

As it can be seen, the finite-state transfer of phrase-based translation models is quite efficient. Even more, taking into account that it is actually a rather simple model if compared to Moses or any other phrase-based decoding methodology. The performance loss of one point in the quality measures can be perfectly explained since the phrase translation table was reduced during the framework adaptation. That has supposed a decrease in the number of model parameters. Anyway, these phrase-based transducers clearly outperform our previous word-alignment-based GIATI implementations, which scored a BLEU of only 20.0.

However, this minor underperformance becomes negligible thanks to all the benefits that a finite state framework presents in practice with respect to the use of stack decoders, especially its application to realtime working environments.

5 Conclusions

Statistical phrase-based translation models are placed at the top of the state-of-the-art in SMT. However, despite their power, current phrase-based systems consume many computational resources, hence reducing their effective use as realtime applications.

A finite state framework represents an interesting alternative because it constitutes an efficient paradigm where quality and realtime factors are properly integrated in order to build translation devices that may be useful for their potential users.

In this work, we have shown how to make use of the bilingual pairs inside a phrase-based model in order to implement a phrase-based ngram model using stochastic finite state transducers with a negligible loss of translation quality. Furthermore, this minor underperformance is widely overcome by all the benefits that a finite state framework presents in practice under realtime requirements.

References

1. Brown, P.F., et al.: A statistical approach to machine translation. Computational Linguistics 16(2), 79–85 (1990)
2. Brown, P.F., et al.: The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics 19(2), 263–311 (1993)
3. Ney, H., et al.: Algorithms for statistical translation of spoken language. IEEE Transactions on Speech and Audio Processing 8(1), 24–36 (2000)
4. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics 29(1), 19–51 (2003)
5. Tomás, J., Casacuberta, F.: Monotone statistical translation using word groups. In: Proceedings of the Machine Translation Summit VIII, Santiago de Compostela, Spain, pp. 357–361 (2001)
6. Zens, R., Och, F.J., Ney, H.: Phrase-based statistical machine translation. In: Jarke, M., Koehler, J., Lakemeyer, G. (eds.) KI 2002. LNCS (LNAI), vol. 2479, pp. 18–32. Springer, Heidelberg (2002)
7. Casacuberta, F., et al.: Some approaches to statistical and finite-state speech-to-speech translation. Computer Speech and Language 18, 25–47 (1994)
8. Casacuberta, F., Vidal, E.: Machine translation with inferred stochastic finite-state transducers. Computational Linguistics 30(2), 205–225 (2004)
9. Casacuberta, F., Vidal, E., Picó, D.: Inference of finite-state transducers from regular languages. Pattern Recognition 38(9), 1431–1443 (2005)
10. Berger, A.L., et al.: Language Translation apparatus and method of using context-based translation models. United States Patent, No. 5510981 (1996)
11. Ortiz, D., García-Varea, I., Casacuberta, F.: An empirical comparison of stack-based decoding algorithms for statistical machine translation. In: Perales, F.J., et al. (eds.) IbPRIA 2003. LNCS, vol. 2652, Springer, Heidelberg (2003)

12. Germann, U., et al.: Fast Decoding and Optimal Decoding for Machine Translation. In: ACL 2001, Toulouse, France, pp. 228–235 (2001)
13. Llorens, D.: Suavizado de autómatas y traductores finitos estocásticos. Phd Thesis, Universidad Politécnica de Valencia (2000)
14. Kumar, S., Deng, Y., Byrne, W.: A weighted finite state transducer translation template model for statistical machine translation. Natural Language Engineering 12(1), 35–75 (2006)
15. Koehn, P., Monz, C.: Manual and Automatic Evaluation of Machine Translation between European Languages. In: NAACL 2006 Workshop on Statistical Machine Translation, pp. 102–121 (2006)
16. Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-Based Translation'. In: NAACL/HLT 2003. In: Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference, May 27–June 1 2003, Edmonton, Canada,(2003)
17. Koehn, P.: Europarl: A Parallel Corpus for Statistical Machine Translation. In: Proceedings of the 10th Machine Translation Summit, pp. 79–86 (2005)
18. Papineni, A.K., et al.: Bleu: A method for automatic evaluation of machine translation. In Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY (2001)
19. Koehn, P., et al.: Moses: Open Source Toolkit for Statistical Machine Translation. Annual Meeting of the Association for Computational Linguistics (demonstration session) (2007)
20. González, J., Casacuberta, F.: Phrase-based finite state models. In: Proceedings of the 6th International Workshop on Finite-State Methods and Natural Language Processing (2007)
21. Casacuberta, F.: Inference of finite-state transducers by using regular grammars and morphisms. In: Oliveira, A.L. (ed.) ICGI 2000. LNCS (LNAI), vol. 1891, pp. 1–14. Springer, Heidelberg (2000)
22. Mariño, J.B., et al.: N-gram-based Machine Translation. Computational Linguistics 32(4), 527–549 (2006)
23. Ortiz, D., García-Varea, I., Casacuberta, F.: Thot: a Toolkit To Train Phrase-based Statistical Translation Models. In: Proceedings of the 10th Machine Translation Summit, pp. 141–148 (2005)
24. Casacuberta, F., et al.: Human Interaction for high quality machine translation. In: Communications of the ACM (in press, 2007)

Learning Spanish-Galician Translation Equivalents Using a Comparable Corpus and a Bilingual Dictionary

Pablo Gamallo Otero[1] and José Ramom Pichel Campos[2]

[1] Departamento de Língua Espanhola, Faculdade de Filologia
Universidade de Santiago de Compostela, Galiza, Spain
[2] Departamento de Tecnologia Linguística da Imaxin|Software
Santiago de Compostela, Galiza

Abstract. So far, research on extraction of translation equivalents from comparable, non-parallel corpora has not been very popular. The main reason was the poor results when compared to those obtained from aligned parallel corpora. The method proposed in this paper, relying on *seed patterns* generated from external bilingual dictionaries, allows us to achieve similar results to those from parallel corpus. In this way, the huge amount of comparable corpora available via Web can be viewed as a never-ending source of lexicographic information. In this paper, we describe the experiments performed on a comparable, Spanish-Galician corpus.

1 Introduction

There exist many approaches to extract bilingual lexicons from parallel corpora [8,16,1,22,14]. These approaches share the same basic strategy: first, bitexts are aligned in pairs of segments and, second, word co-ocurrences are computed on the basis of that alignment. They usually reach high score values, namely about 90% precision with 90% recall. Unfortunately, parallel texts are not easily available, in particular for minority languages. To overcome this drawback, different methods to extract bilingual lexicons have been implemented lately using non-parallel, comparable corpora. These methods take up with the idea of using the Web as a huge resource of multilingual texts which can be easily organized as a collection of non-parallel, comparable corpora. A non-parallel and comparable corpus (hereafter "comparable corpus") consists of documents in two or more languages which are not translation of each other and deal with similar topics. However, the accuracy scores of such methods are not as good as those reached by the strategies based on aligned parallel corpora. So far, the highest values have not improved an 72% accuracy [18], and that's without taking into consideration the coverage of the extracted lexicon over the corpus.

This paper proposes a new method to extract bilingual lexicons from a POS tagged comparable corpus. Our method relies on the use of a bilingual dictionary to identify bilingual correlations between pairs of lexico-syntactic patterns. Such patterns will be used as "seed expressions" as follows: a lemma of the target language will be taken as a possible translation of a lemma in the source language if both lemmas co-occur with a great number of seed patterns. Beside the external dictionary, we also identify seed patterns with cognates previously selected from the comparable corpus. We will work not only on monoword lemmas but also on multiwords. Our results improve the accuracy reached by Rapp (i.e. 72%), for a coverage of more than 80%. These encouraging results show that the huge amount of comparable corpora via Web can be seen as an endless resource of lexicographic knowledge.

The article is organized as follows. In Section 2, we will situate our approach with regard to the state of art in comparable corpora extraction. Section 3 will be focused on defining the different steps of our approach. Then, in 4, we will describe the experiments performed on a Spanish-Galician corpus as well as an evaluation protocol. Finally, we will enumerate some conclusions and discuss future work.

2 Related Work

There are not many approaches to extract bilingual lexicons from non-parallel corpora in comparison to those using a strategy based on aligned, parallel texts. The most popular method to extract word translations from non-parallel, comparable corpora is described and used in [6,7,18,4,19]. The starting point of this strategy is as follows: word w_1 is a candidate translation of w_2 if the words with which w_1 co-occurs within a particular window are translations of the words with which w_2 co-occurs within the same window. This strategy relies on a list of bilingual word pairs (called *seed words*) provided by an external bilingual dictionary. So, w_1 is a candidate translation of w_2 if they tend to co-occur with the same seed words. The main drawback of this method is the use of word windows to define coarse-grained contexts. According to the Harris's hypothesis [13], counting co-occurrences within a window of size N is less precise than counting co-occurrences within local syntactic contexts. In the most efficient approaches to thesaurus generation [12,15], word similarity is computed using co-occurrences between words and specific syntactic contexts. Syntactic contexts are considered to be less ambiguous and more sense-sensitive than contexts defined as windows of size N. In order to define contexts with more fine-grained information, we build a list of bilingual lexico-syntactic templates. In [9], these templates were previously extracted from small samples of parallel corpus. In this paper, however, they are extracted directly from an external bilingual dictionary. As such templates represent unambiguous local contexts of words, they are discriminative and confident seed expressions to extract word translations from comparable texts. In [21], syntactic templates are also used for extraction of translations, but they were specified with semantic attributes introduced by hand. In [5], it is described a particular strategy based on a multilingual thesaurus instead of an external bilingual dictionary. Finally, some researchers have focused on a different issue: disambiguation of candidate translations. According to [17], the process of building bilingual lexicons from non-parallel corpora is a too difficult and ambitious objective. He preferred to work on a less ambitious task: to choose between several translation alternatives previously selected from a bilingual dictionary.

3 The Approach

Our approach consists of three steps: (1) text processing, (2) building a list of seed bilingual patterns by using a bilingual dictionary and a set of cognates previously selected from the corpus, and (3), translation equivalents extraction from a comparable corpus making use of the list of seed patterns.

3.1 Text Processing

POS Tagging and Multiword Extraction. First, the texts of both languages are lemmatized and POS tagged. Lemmatization also involves name entity recognition (i.e., identification of proper nouns). Proper nouns can be either mono or multiword units. Besides monowords lemmas and proper nouns, we also extract multiwords, that is, lemmas consisting of several lexical units with some degree of internal cohesion: e.g., "traffic jam", "tv channel", "take into account", etc. This type of expressions are extracted using basic patterns of POS tags such as N-PRP, N-A, V-N, etc. This task is performed on the comparable corpus, so we extract multiword candidates in both languages. Then, the list of multiword candidates is reduced with a basic statistical filter, which only selects those multiwords with a *SCP* coefficient higher than a empirically set threshold. Here, we follows the strategy described in [20].

Dependency Triplets and Lexico-Syntactic Patterns. Once the corpus has been POS tagged and the multiwords have been extracted, we build a collocation database where each entry consists of a lemma (either monoword unit or multiword) and the lexico-syntactic patterns with which it co-occurs in the corpus. The database is built in two steps. First, we make use of regular expressions to identify binary dependencies. Regular expressions represent basic patterns of POS tags which are supposed to stand for syntactic dependencies between two lemmas. In our approach, we work with dependencies between verbs, nouns, and adjectives. Second, we extract lexico-syntactic patterns from the dependencies and count the co-occurrences of lemmas with those lexico-syntactic patterns. Let's take an example. Suppose our corpus contains the following tagged sentence:

> a_D man_N see_V yesterday_R a_D very_R big_A dog_N with_PRP a_D broken_A leg_N

The first step consists in identifying dependencies between lemmas using basic patterns of POS tags. Dependencies are noted as triplets: $(head, rel, dependent)$. Table 1 shows the 5 triplets extracted from the sentence above using different

Table 1. Dependency triplets and patterns of POS tags

Dependencies	Patterns of POS tags
$(see, subj, man)$	$(\mathbf{N})(?: A\|R) * (\mathbf{V})$
(see, obj, dog)	$(\mathbf{V})(?: R\|D\|R\|A\|N) * (\mathbf{N})$
$(dog, with, leg)$	$(\mathbf{N})(?: R\|A) * (\mathbf{PRP})(?: D\|R\|A\|N) * (\mathbf{N})$
(dog, mod, big) $(leg, mod, broken)$	$(\mathbf{A})(?: N) * (\mathbf{N})$
$()$	$(\mathbf{N})(?: N) * (\mathbf{N})$
$()$	$(\mathbf{V})(?: R) * (\mathbf{PRP})(?: D\|R\|A\|N) * (\mathbf{N})$

Table 2. Collocation database of lemmas and lexico-syntactic patterns

Lemmas	Lexico-Syntactic Patterns and freqs.
man	$< (see, subj, N), 1 >$
see	$< (V, subj, man), 1 >$, $< (V, obj, dog), 1 >$
big	$< (dog, mod, A), 1 >$
dog	$< (N, mod, big), 1 >$, $< (N, with, leg), 1 >$
broken	$< (leg, mod, A), 1 >$
leg	$< (N, mod, broken), 1 >$, $< (dog, with, N), 1 >$

patterns of POS tags. The 5 extracted triplets instantiate 4 schemes of dependencies: adjective-noun, noun-verb, verb-noun, and noun-prep-noun. The sentence does not contain triplets instantiating the noun-noun and verb-prep-noun dependencies. Wildcards $(?: D|R|A|N)*$ stand for optional modifiers, that is, they represent sequences of determiners, adverbs, adjectives, or nouns that are not considered for triplets.

In the second step, the extracted triplets allow us to easily build the collocation database depicted in Table 2. The first line of the table describes the entry *man*. This noun co-occurs once with one lexico-syntactic pattern, which represents the subject position of the verb *see*. The second line describes the entry *see*, which co-occurs once with two lexico-syntactic patterns: a verb co-occurring with *man* in the subject position and a verb co-occurring with *dog* in the object position. The remaining lines describe the collocation information of the other nouns and adjectives appearing in the sentence above.

Notice we always extract 2 complementary lexico-syntactic patterns from a triplet. For instance, from $(dog, with, leg)$, we extract:

– $(N, with, leg)$
– $(dog, with, N)$

This is in accordance with the notion of co-requirement defined in [10]. In this work, two syntactically dependent words are no longer interpreted as a standard "predicate-argument" structure, where the predicate is the active function imposing syntactic and semantic conditions on a passive argument, which matches such conditions. On the contrary, each word of a binary dependency is perceived simultaneously as a predicate and an argument. In the example above, $(dog, with, N)$ is seen as a unary predicate that requires nouns denoting parts of dogs (e.g. legs), and simultaneously, $(N, with, leg)$ is another unary predicate requiring as argument entities having legs (e.g. dogs).

To simplify the process of extracting binary relations, long-distance dependencies are not taken into account. So, we do not propose the attachment between the verb "see" and the prepositional phrase "with a broken leg". In fact, our use of regular expressions over POS-tags emulates a parsing strategy based on the Right-Association heuristic. It is a robust analysis, and about 75% of the triplets are correctly extracted. Note that the patterns of tags in Table 1 work well for English texts. To extract triplets from texts in Romance languages such as Spanish, French, Portuguese, or Galician, we need to do, at least, 3 tiny changes: nouns as optional modifiers are not taken into account; a new pattern with dependent adjectives at the right position of nouns is required; the noun in the left position of a noun-noun dependency must be considered the head of the triplet. The experiments that will be described later were performed over Spanish and Galician corpora.

3.2 Generating Seed Lexico-Syntactic Patterns

To extract translation equivalents from a comparable corpus, a list of "seed" expressions is required. In our approach, the seed expressions used as cross-language pivot contexts are not bilingual pairs of words as in related work, but bilingual pairs of lexico-syntactic patterns (or "seed patterns"). The process of building a list of seed patterns consists of two steps: first, we generate a large list from an external bilingual dictionary (and from a set of cognates). Second, this list is reduced by filtering out those pairs of patterns that do not occur in the comparable corpus. We also remove those that are sparse or unbalanced in the corpus.

Patterns from Bilingual Dictionaries. In order to generate bilingual correlations between lexico-syntactic patterns, we make use of bilingual dictionaries. Let's suppose that an English-Spanish dictionary translates the noun *import* into the Spanish counterpart *importación*. To generate bilingual pairs of lexico-syntactic patterns from these two nouns, we follow basic rules such as: (1) if *import* is the subject of a verb, then its Spanish equivalent, *importación*, is also the subject; (2) if *import* is modified by an adjective at the left position, then its Spanish equivalent is modified by an adjective at the right position; (3) if *import* is restricted by a prepositional complement headed by the preposition *in*, then its Spanish counterpart is restricted by a prepositional complement headed by the preposition *en*. The third rule needs a closed list of English prepositions and their more usual Spanish translations. For each entry (noun, verb, or adjective), we only generated a subset of all possible patterns. Table 3 depicts the patterns generated from the bilingual pair *import-importación* and a restricted set of rules.

Table 3. Bilingual correlations between patterns generated from the translation pair: import-importación

English	**Spanish**
$(import, of\|to\|in\|for\|by\|with, N)$	$(importación, de\|a\|en\|para\|por\|con, N)$
$(N, of\|to\|in\|for\|by\|with, import)$	$(N, de\|a\|en\|para\|por\|con, importación)$
$(V, obj, import)$	$(V, obj, importación)$
$(V, subj, import)$	$(V, subj, importación)$
$(V, of\|to\|in\|for\|by\|with, import)$	$(V, de\|a\|en\|para\|por\|con, importación)$
$(import, mod, A)$	$(importación, mod, A)$

In order to have a larger list of bilingual patterns, we also use a complementary strategy based on the identification of cognates from the comparable texts. We call "cognates" two lemmas written in the same way. We select those cognates appearing in the texts that are not in the bilingual dictionary. Most of them are proper names and dates. As they can be treated as entries of a bilingual dictionary, we are able to generate more bilingual lexico-syntactic patterns using the same basic rules described above.

Filtering. The list generated in the previous process may contain lexico-syntactic patterns that do not occur in the comparable corpus, i.e., in the collocation database created in the first step of the approach (Subsection 3.1). Such patterns are removed. In addition, we also filter out those bilingual pairs that have one of these two properties: being sparse or being unbalanced in the comparable corpus. A bilingual pair of patterns is sparse if it has high dispersion. Dispersion is defined as the number of different lemmas occurring with a bilingual pair divided by the total number of lemmas in the comparable corpus. A bilingual pair is unbalanced when one of the patterns is very frequent while the other one is very rare. We use empirically set thresholds to separate sparse and unbalanced bilingual patterns from the rest. The final list of selected patterns will be used as seed expressions in the following step.

3.3 Identifying Translation Equivalents

The final step consists in extracting translation equivalents of lemmas with the help of the list of seed patterns. To compute the similarity between a lemma in the source language and a lemma in the target language, we conceive lemmas as vectors whose dimensions are the seed patterns. The value for each dimension is selected from the co-occurrence information stocked in the *collocation database* (see above Subsection 3.1). For instance, let's suppose that the collocation database contains the English lemma *uranium* co-occurring 14 times with the lexico-syntactic pattern $(import, of, N)$. As this English pattern was associated to the Spanish pattern $(importación, of, N)$ in the list of seed patterns, then, we have to build a vector for *uranium* whose value is 14 in the dimension defined by this pair of patterns. Note that all Spanish lemmas co-occurring 14 times with (*importación*, of, N) require vectors with the same value in the same dimension.

Similarity between lemmas l_1 and l_2 is computed using the following version of the *Dice* coefficient:

$$Dice(l_1, l_2) = \frac{2 * \sum_i min(f(l_1, p_i), f(l_2, p_i))}{f(l_1) + f(l_2)} \qquad (1)$$

where $f(l_1, p_i)$ represents the number of times the lemma l_1 co-occurs with a seed pattern p_i. In some experiments, instead of co-occurrences we used log-likelihood as association value between lemmas and patterns. This weighted version of the measure did not improve the results in a significant way, since unbalanced and sparse patterns were filtered out before computing similarity. So, all the experiments described and evaluated in the next section were performed using only co-occurrences as association value.

As a result, each lemma of the source language is associated a list of candidate translations. This list is ranked by degree of similarity.

4 Experiments and Evaluation

4.1 The Comparable Corpus

The experiments were performed on a Spanish and Galician comparable corpus, which is constituted by news from on-line journals published between 2005 and 2006. As the Spanish corpus, we used 13 million words of two newspapers: *La Voz de Galicia* and *El Correo Gallego*, and as Galician corpus 10 million words of *Galicia-Hoxe*, *Vieiros* and *A Nosa Terra*. the Spanish and Galician texts were lemmatized and POS tagged using a multilingual free software: Freeling [3]. Since the orientation of the newspaper is quite similar the two corpora can be considered as more or less comparable.

4.2 The Bilingual Dictionary

The bilingual dictionary used to generate part of the seed patterns is the lexical resource integrated in an open source machine translation system, OpenTrad, for Spanish-Galician [2]. The final objective of our experiments is to update that resource in order to improve the results of the machine translation system, which is used by *La Voz de Galicia*, the sixth newspaper in Spain concerning the number of readers. The dictionary contains about $25,000$ Spanish-Galician entries.

The amount of bilingual patterns generated from the entries of the dictionary is $539,561$. In addition, we generated $754,469$ further patterns from bilingual cognates. In sum, we got $1,294,030$. However, after the filtering process, the list of seed patterns is reduced to only $127,604$. This is the number of dimensions of lemma vectors.

4.3 The Evaluation Protocol

To evaluate the efficiency of our method in the process of extracting translation equivalents, we elaborate an evaluation protocol with the following characteristics. Accuracy is computed taking into account coverage at tree levels: 90%, 80%, and 50%. In our work, to choose a level of coverage, we need to rank lemmas of the source language by frequency and select those whose frequency covers a specific percentage of the total frequency in the corpus. More precisely, given a ranked list of all lemmas found in the corpus, a level of coverage is the frequency in the corpus of an ordered set of lemmas in the list divided by the frequency of all lemmas. This ratio was computed separately for three different POS categories: nouns, adjectives, and verbs. This way, a 90% of coverage for nouns means that the frequency of the nouns considered for evaluation is 90% with regard to the total frequency of all nouns in the corpus.

To compute accuracy, we need first to choose a specific POS category and a particular level of coverage. Then, we randomly extract 150 test lemmas of this category from the list of lemmas whose occurrences in the corpus achieve the level of coverage at stake. We compute two types of accuracy: *accuracy-1* is defined as the number of times a correct translation candidate of the test lemma is ranked first, divided by the number of test lemmas. *Accuracy-10* is the number of correct candidates appearing in the top 10, divided by the number of test lemmas. Indirect associations are judged to be incorrect.

As far as we know, no definition of coverage (nor recall) has ever been proposed in related work. In most evaluation protocols of previous work, authors only give information on the number of occurrences of the test words in the corpus. In some work, test words are the N most frequent expressions in the training corpus [7], while in other experiments, they are word types or lemmas with a frequency higher than N (where N is often >= 100) [11,4]. In fact, as absolute frequencies are dependent on the corpus size, they are not very useful if we try to compare the precision or accuracy among different approaches. By considering levels of coverage, which are independent of the corpus size, we try to overcome such a limitation.

4.4 Results

Table 4 shows the evaluation of our approach. For each POS category (including multiword nouns), and for each level of coverage (90%, 80%, and 50%), we compute *accuracy-1* and *accuracy-10.*

As far as nouns are concerned, the three levels of coverage (90%, 80%, and 50%) correspond to three lists of lemmas containing $9,798$, $3,534$, and 597 nouns, respectively. As nouns, we include all sort of proper names. Figure 1 depicts the progression of the two accuracies (1 and 10) at the three levels of coverage. With a coverage of 80%, accuracy is quite acceptable: between .80 and .90. At this level of coverage, the frequency of the test lemmas is ≥ 129. In fact, such a minimum frequency is not far from the thresholds proposed by related works, where the smallest frequency of test words was, in most cases, 100. However, in those works the best accuracy merely achieves 72% [18].

Table 4. Evaluation of our approach

Category	Cov(%)	Acc-1	Acc-10	lemmas
Noun	90%	.55	.60	9,798
Noun	80%	.81	.90	3,534
Noun	50%	.95	.99	597
Adj	90%	.61	.70	1,468
Adj	80%	.81	.87	639
Adj	50%	.94	.98	124
Verb	90%	.92	.99	745
Verb	80%	.97	.100	401
Verb	50%	.100	.100	86
multi-lex	50%	.59	.62	2,013

Table 5. Evaluation of the baseline method

Category	Cov(%)	Acc-1	Acc-10	lemmas
Noun	80%	.26	.54	3,534
Adj	80%	.43	.70	639

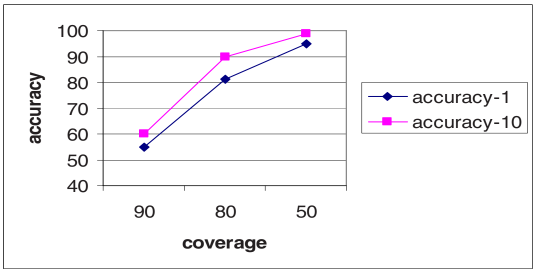


Fig. 1. Accuracy of nouns at 3 levels of coverage

Regarding accuracy of adjectives and verbs, there is a significant difference in their results. Whereas the accuracy of verbs is close to .100 at the coverage of 80%, adjectives only reach about .80 of accuracy with the same coverage. The main drawback with adjectives comes from the difficulties of the POS tagger to correctly disambiguate between adjectives and past participles.

As far as multiword nouns are concerned, accuracy is about .60 at the coverage of 50%. The main drawback regarding multiwords is their low frequency in the corpus. The minimum frequency of the 2,013 lemmas evaluated at this level is very low, 40, which prevents us from getting acceptable results. However, our results are better than those obtained by similar approaches using multiword terms, with .52 accuracy in the best case [6][1].

[1] The merit of this work is to extract translation equivalents from two very different languages: English and Japanese.

Finally, Table 5 depicts the results of a baseline method. The baseline strategy relies on seed words and windows of size 2 (i.e., 4 context positions) instead of on lexico-syntactic patterns. In fact, with this baseline, we tried to simulate some aspects of the approach described by [18]. To permit comparing this approach to ours, we used as similarity coefficient the dice measure defined above. As in [18], our baseline method only search for translation equivalents of nouns and adjectives. In table 5, we can observe the accuracy obtained using the baseline method when the coverage is situated at 80%. This accuracy is significatively lower that the scores reached by our approach. So, lexico-syntactic patterns seem to be more precise than contexts based on windows of size N. Notice that the accuracy in our simulation is lower than that obtained by Rapp (about 72%). Such a difference can be explained by the size of our training corpus, 10 times smaller than the corpus used by Rapp.

5 Conclusions and Future Work

Few approaches to extract word translations from comparable, non-parallel texts have been proposed so far. The main reason is that results are not yet very encouraging. Whereas for parallel texts, most work on word translation extraction reaches more than 90%, the accuracy for non-parallel texts has been around 72% up to now. The main contribution of the approach proposed in this paper is to use bilingual pairs of lexico-syntactic patterns as seed expressions. This makes a significant improvement to about 80/90% of word translations identified correctly if only the best candidate is considered, and about 90/95% if we consider the top 10. These results are not very far from those obtained by approaches based on parallel texts. Such results show that non-parallel, comparable corpora can be considered as an interesting source of lexicographic knowledge. Moreover, there is still a good margin to improve results. Given that comparable corpora are growing daily as the web is getting larger, it could be easy to update and enrich bilingual lexicons and translation memories in an incremental way. Our current work is precisely to retrieve monthly further documents from the web in order to make the training corpus larger and update our bilingual lexicon. This way, we aim at improving the specific bilingual resource used by OpenTrad, a Spanish-Galician machine translation system.

Acknowledgments

This work has been supported by the Galician Government, within the project ExtraLex, ref: PGIDIT07PXIB204015PR.

References

1. Ahrenberg, L., Andersson, M., Merkel, M.: A simple hybrid aligner for generating lexical correspondences in parallel texts. In: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL 1998), Montreal, pp. 29–35 (1998)

2. Armentano-Oller, C., et al.: Open-source portuguese-spanish machine translation. In: Vieira, R., et al. (eds.) PROPOR 2006. LNCS (LNAI), vol. 3960, pp. 50–59. Springer, Heidelberg (2006)
3. Carreras, X., Chao, I., Padró, L., Padró, M.: An open-source suite of language analyzers. In: 4th International Conference on Language Resources and Evaluation (LREC 2004), Lisbon, Portugal (2004)
4. Chiao, Y.-C., Zweigenbaum, P.: Looking for candidate translational equivalents in specialized, comparable corpora. In: 19th COLING 2002 (2002)
5. Dejean, H., Gaussier, E., Sadat, F.: Bilingual terminology extraction: an approach based on a multilingual thesaurus applicable to comparable corpora. In: COLING 2002, Tapei, Taiwan (2002)
6. Fung, P., McKeown, K.: Finding terminology translation from non-parallel corpora. In: 5th Annual Workshop on Very Large Corpora, pp. 192–202 (1997)
7. Fung, P., Yee, L.Y.: An ir approach for translating new words from nonparallel, comparable texts. In: Coling 1998, Montreal, Canada, pp. 414–420 (1998)
8. Gale, W., Church, K.: Identifying word correspondences in parallel texts. In: Workshop DARPA SNL (1991)
9. Gamallo, P.: Learning bilingual lexicons from comparable english and spanish corpora. In: Machine Translation SUMMIT XI, Copenhagen, Denmark (2007)
10. Gamallo, P., Agustini, A., Lopes, G.: Clustering syntactic positions with similar semantic requirements. Computational Linguistics 31(1), 107–146 (2005)
11. Gamallo, P., Pichel, J.R.: An approach to acquire word translations from non-parallel corpora. In: Bento, C., Cardoso, A., Dias, G. (eds.) EPIA 2005. LNCS (LNAI), vol. 3808, Springer, Heidelberg (2005)
12. Grefenstette, G.: Explorations in Automatic Thesaurus Discovery. Kluwer Academic Publishers, USA (1994)
13. Harris, Z.: Distributional structure. In: Katz, J.J. (ed.) The Philosophy of Linguistics, pp. 26–47. Oxford University Press, New York (1985)
14. Kwong, O.Y., Tsou, B.K., Lai, T.B.: Alignment and extraction of bilingual legal terminology from context profiles. Terminology 10(1), 81–99 (2004)
15. Lin, D.: Automatic retrieval and clustering of similar words. In: COLING-ACL 1998, Montreal (1998)
16. Melamed, D.: A portable algorithm for mapping bitext correspondences. In: 35th Conference of the Association of Computational Linguistics, Madrid, Spain, pp. 305–312 (1997)
17. Nakagawa, H.: Disambiguation of single noun translations extracted from bilingual comparable corpora. Terminology 7(1), 63–83 (2001)
18. Rapp, R.: Automatic identification of word translations from unrelated english and german corpora. In: ACL 1999, pp. 519–526 (1999)
19. Shao, L., Ng, H.T.: Mining new word translations from comparable corpora. In: 20th International Conference on Computational Linguistics (COLING 2004), Geneva, Switzerland, pp. 618–624 (2004)
20. Silva, J.F., Dias, G., Guilloré, S., Lopes, G.P.: Using localmaxs algorithm for the extraction of contiguous and non-contiguous multiword lexical units. In: Progress in Artificial Intelligence. LNCS (LNAI), pp. 113–132. Springer, Heidelberg (1999)
21. Tanala, T.: Measuring the similarity between compound nouns in different languages using non-parallel corpora. In: 19th COLING 2002, pp. 981–987 (2002)
22. Tiedemann, J.: Extraction of translation equivalents from parallel corpora. In: 11th Nordic Conference of Computational Linguistics, Copenhagen, Denmark (1998)

Context-Based Sentence Alignment in Parallel Corpora

Ergun Biçici

Koç University
Rumelifeneri Yolu 34450
Sariyer, Istanbul, Turkey
ebicici@ku.edu.tr

Abstract. This paper presents a language-independent context-based sentence alignment technique given parallel corpora. We can view the problem of aligning sentences as finding translations of sentences chosen from different sources. Unlike current approaches which rely on pre-defined features and models, our algorithm employs features derived from the distributional properties of words and does not use any language dependent knowledge. We make use of the context of sentences and the notion of Zipfian word vectors which effectively models the distributional properties of words in a given sentence. We accept the context to be the frame in which the reasoning about sentence alignment is done. We evaluate the performance of our system based on two different measures: sentence alignment accuracy and sentence alignment coverage. We compare the performance of our system with commonly used sentence alignment systems and show that our system performs 1.2149 to 1.6022 times better in reducing the error rate in alignment accuracy and coverage for moderately sized corpora.

Keywords: sentence alignment, context, Zipfian word vectors, multilingual.

1 Introduction

Sentence alignment is the task of mapping the sentences of two given parallel corpora which are known to be translations of each other to find the translations of corresponding sentences. Sentence alignment has two main burdens: solving the problems incurred by a previous erroneous sentence splitting step and aligning parallel sentences which can later be used for machine translation tasks. The mappings need not necessarily be 1-to-1, monotonic, or continuous. Sentence alignment is an important preprocessing step that affects the quality of parallel text.

A simple approach to the problem of sentence alignment would look at the lengths of each sentence taken from parallel corpora and see if they are likely to be translations of each other. In fact, it was shown that paragraph lengths for the English-German parallel corpus from the economic reports of Union Bank of Switzerland (UBS) are highly correlated with a correlation value of 0.991 [1]. A more complex approach would look at the neighboring sentence lengths as well. Our approach is based on this knowledge of context for given sentences from each corpus and the knowledge of distributional features of words, which we name Zipfian word vectors, for alignment purposes. A Zipfian word vector is an order-free representation of a given sentence in a corpus, in which the length and the number of words in each entry of the vector are determined based on the quantization of the frequencies of all words in the corpus.

In this paper, we consider sentence alignment based on the local context information. The resulting methodology is language-independent; it can handle non-monotonic alignments; it does not require any stemming, dictionaries, or anchors; it handles deletions; and it extends the type of alignments available up to 6-way. Sentence alignments of given parallel corpora are determined by looking at the local context of a given sentence which consists of the surrounding sentences. The problem of sentence alignment is a central problem in machine translation and similar in essence to many other problems that involve the identification of mappings. It is a subset of the problem of *sequence comparison*, which deals with difficult comparisons that arise when the correspondence of items in the sequences are not known in advance [2]. We used a publicly available and easily accessible dataset [3] for our experiments, so that our results can be easily replicated by others.

We observe that valuable information can be inferred from the context of given sentences and their distributional properties for alignment purposes. The following sections are organized as follows. In the next section, we review related work and present its limitations. In Sect. 3, we provide a formalization of the sentence alignment problem, define Zipfian word vectors, present our feature representation, and discuss context in sentence alignment and our alignment algorithm. In Sect. 4, we present the results of our experiments and the last section concludes.

2 Related Work

Brown *et. al.* [4] provide a statistical technique for sentence alignment using the number of word tokens in each sentence as well as the comments in their dataset (Canadian Hansards corpora[1]), which serve as anchor points. They define an alignment as a sequence of beads, which are considered as groupings of English and French sentences that lengths that are close. Gale and Church [1] observe that sentence lengths of source and target sentences are correlated. They limit their alignments to 1-1, 1-0, 0-1, 2-1, 1-2, and 2-2 types of mappings, where the numbers represent the number of sentences that map to each other. The reason for their choice in using sentence lengths in terms of characters rather than in terms of word tokens as was chosen by Brown *et. al.* [4] is that since there are more characters there is less uncertainty.

Both Brown *et. al.* and Gale and Church [1] assume that the corpus is divided into chunks and they ignore word identities. Chen [5] describes an algorithm that constructs a simple statistical word-to-word translation model on the fly during sentence alignment. The alignment of a corpus ($\mathcal{S}$, $\mathcal{T}$) is the alignment $\mathbf{m}$ that maximizes $P(\mathcal{T}, \mathbf{m} \mid \mathcal{S})$, where P denotes the probability. Chen found that 100 sentence pairs are sufficient to train the model to a state where it can align correctly. Moore's [6] sentence alignment model combines sentence-length-based and word-correspondence-based approaches, achieving high accuracy at a modest computational cost. Moore uses a modified version of the IBM Translation Model 1 [7]:

$$P(T \mid S) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^{m} \sum_{i=0}^{l} \mathrm{tr}(t_j | s_i),$$

[1] Available from Linguistic Data Consortium at `http://www.ldc.upenn.edu/`

where $\text{tr}(t_j|s_i)$ corresponds to the translation probability of the word $t_j \in T = \{t_1, \ldots, t_m\}$ given $s_i \in S = \{s_1, \ldots, s_l\}$ and ϵ is some small fixed number. Instead of $P(T|S)$, Moore makes use of $P(S,T)$ due to the noisy channel model [8], since S is hypothetically corrupted by some "noise" and turned into t.

Context and its selection is very important in many areas of natural language processing. Most of the work on context focuses on finding an optimal context size which gives good performance globally on the test cases. Yet this optimal value is sensitive to the type of ambiguity [9]. The dynamic nature of the context is noticed for the word sense disambiguation task by Yarowsky and Florian [10] and they further claimed that the context sizes for nouns, verbs, and adjectives should be in the 150, 60-80, and 5 word vicinity of a given word respectively. Wang [11] gives a nice example of word senses' context dependence in Fig. 1. As we increase the size of the context, the sense of the Chineese word varies between think and read. Ristad [12] makes use of a greedy heuristic to extend a given context for the purpose of finding models of language with fewer parameters and lower entropy. In our previous work [13], we examined alternatives for local context models for sentence alignment. As we did previously, in this work, we accept the context to be the frame in which the reasoning about sentence alignment is done.

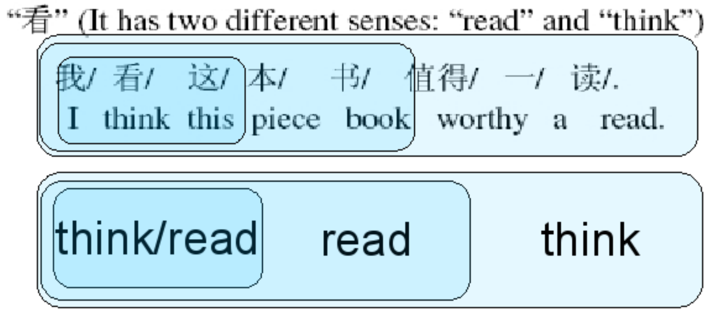

"看" (It has two different senses: "read" and "think")
我/ 看/ 这/ 本/ 书/ 值得/ 一/ 读/.
I think this piece book worthy a read.
think/read
read
think

Fig. 1. Word sense dependence on context

Earlier work on sentence alignment assume that the order of sentences in each corpus is preserved; as the beads on a string preserve the order, their models assume that the mapping function $\mathbf{m}$ is monotonic. Sentence alignment literature makes extensive use of simplifying assumptions (e.g. the existence of anchors, dictionaries, or stemming), biased success criterion (e.g. selecting only 1-1 type alignments or removing badly aligned sentences from consideration), and the use of datasets that cannot be qualitatively judged and compared to other results. In this paper, we overcome these limitations by removing simplifying assumptions about the dataset and generalizing the problem space by generalizing our representation of the data. Our goal is not to seek the best performance in only 1-1 type alignments since machine translation tasks cannot be reduced to 1-1 type alignments. Although 1-1 type alignments constitute 97.21% of the 52594 alignments overall, they cover only 96.01% of the 106504 sentences involved in the Multext-East [3] dataset. We also use a new measure of success, sentence alignment coverage, which also considers the number of sentences involved in the alignment. We use the Multext-East[2] corpus, which provides us access to large amounts of manually sentence-split and sentence-aligned parallel corpora and a good dataset for the evaluation of performance. As this dataset contains alignments for 8 different language pairs, it suits well for demonstrating our system's language independence.

3 Sentence Alignment

3.1 Problem Formulation

A *parallel corpus* is a tuple $(\mathcal{S}, \mathcal{T})$, where $\mathcal{S}$ denotes the source language corpus and $\mathcal{T}$ denotes the target language corpus such that $\mathcal{T}$ is the translation of $\mathcal{S}$. Since the translation could have been done out of order or lossy, the task of *sentence alignment* is to find a mapping function, $\mathbf{m} : \mathcal{S} \rightarrow \mathcal{T}$, such that a set of sentences $T \subseteq \mathcal{T}$ where $T = \mathbf{m}(S)$ is the translation of a set of sentences $S \subseteq \mathcal{S}$. Then, under the mapping $\mathbf{m}$, we can use T whenever we use S.

We assume that $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$ and $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}\}$, where $|\mathrm{corpus}|$ refers to the number of sentences in corpus and s_i and t_i correspond to the ith sentences in $\mathcal{S}$ and in $\mathcal{T}$ respectively. The sentences in $\mathcal{S}$ and $\mathcal{T}$ form an ordered set where an *ordered set* is an n-tuple, denoted by $\{a_1, a_2, \ldots, a_n\}_{\leqslant}$, such that there exists a total order, $\leqslant$, defined on the elements of the set. We also assume that a set of sentences $S \subseteq \mathcal{S}$ where $S = \{s_i, s_{i+1}, \ldots, s_j\}$ is chosen such that $\forall k, i \leq k < j,\ s_k \leqslant_{\mathcal{S}} s_{k+1}$. The same argument applies for a set of sentences selected from $\mathcal{T}$. Therefore, it is also meaningful to order two sets of sentences S_1 and S_2 selected from a given corpus $\mathcal{S}$ with the following semantics: Let start_{S_1} and start_{S_2} be the starting sentences of S_1 and S_2 correspondingly, then, $S_1 \leqslant_{\mathcal{S}} S_2 \Leftrightarrow \mathrm{start}_{S_1} \leqslant_{\mathcal{S}} \mathrm{start}_{S_2}$. A mapping $\mathbf{m} : \mathcal{S}_{\leqslant_{\mathcal{S}}} \rightarrow \mathcal{T}_{\leqslant_{\mathcal{T}}}$, is *monotone* or *order-preserving*, if for $S_1, S_2 \subseteq \mathcal{S}$, $S_1 \leqslant_{\mathcal{S}} S_2$ implies $\mathbf{m}(S_1) \leqslant_{\mathcal{T}} \mathbf{m}(S_2)$, where $\mathbf{m}(S_1), \mathbf{m}(S_2) \subseteq \mathcal{T}$.

The usual evaluation metric used is the percentage of correct alignments found in a given set of alignments, which we name sentence alignment accuracy. This measure does not differentiate between an alignment that involves only one sentence as in 1-0 or 0-1 type alignments and an alignment that involves multiple sentences as in 1-5. Therefore, we define sentence alignment coverage as follows:

Definition 1 (Sentence Alignment Coverage). *Sentence alignment coverage is the percentage of sentences that are correctly aligned in a given parallel corpus.*

Thus, for sentence alignment coverage, an alignment of type 1-5 is three times more valuable than an alignment of type 1-1.

3.2 Zipfian Word Vectors

It is believed that distribution of words in large corpora follow what is called Zipf's Law, where "a few words occur frequently while many occur rarely" [14]. We assume that distributions similar to Zipfian are ubiquitous in all parallel corpora. Based on this assumption, we create Zipfian word vectors by making use of the distributions of words in a given corpus.

[2] Also available at `http://nl.ijs.si/ME/V3/`

Definition 2 (Zipfian Word Vector). *Given a set of sentences, S, chosen from a given corpus, $\mathcal{S}$, where* $\mathrm{maxFreq}$ *represents the frequency of the word with the maximum frequency in $\mathcal{S}$, and a binning threshold, b, the Zipfian word vector representation of S is defined as a vector V of size $\frac{\log(\mathrm{maxFreq})}{\log(b)}$, where $V[i]$ holds the number of words in S that have a frequency of $\lfloor \frac{\log(\mathrm{freq}(w))}{\log(b)} \rfloor = i$ for word $w \in S$.*

Thus, each bin contains the number of words with similar frequencies in the given corpus. We assume that $\mathrm{ZWV(S)}$ is a function that returns the Zipfian word vector of a given set of sentences S. Thus, for a single sentence as in:

$S =$" big brother is watching you " , the caption beneath it ran .,

the Zipfian word vector becomes:

$$\mathrm{ZWV(S)} = [14, 1, 3, 0, 1, 3, 2, 0, 1, 1, 2],$$

where the sentence length in the number of tokens is added to the beginning of the Zipfian word vector as well. In Sect. 4 we examined the performance when different sentence length definitions for the Zipfian word vectors used in the system. Note that Zipfian word vectors contain information about anything that is recognized as a token after tokenization. We used the Penn Tree Bank [15] tokenizer, which was designed for English but still effective since all of the Eastern European languages we experimented with use the same punctuation characters with English.

The TCat concept [16] used for text classification is similar in its use of Zipfian distribution of words. While TCat is based on three levels of frequency (high, medium, and low frequency levels) we vary the length of the Zipfian word vector to increase the accuracy in the learning performance and adapt to the problem. Also, in TCat, each level of frequency behaves as a binary classifier, differentiating between positive and negative examples whereas each bin in our model behaves as a quantization of features to be used in learning.

3.3 Feature Representation

We assume that $\mathcal{S} = \{S_1, \ldots, S_i, \ldots, S_N\}$ and $\mathcal{T} = \{T_1, \ldots, T_i, \ldots, T_N\}$ where N is the total number of alignments and S_i and T_i correspond to the set of sentences involved in the ith alignment. For each set of sentences, that become a candidate for alignment within the sentence alignment algorithm, we create what we call the *Zipfian word matrix*. The Zipfian word matrix of a given set of sentences, S, is essentially the matrix we get when we concatenate the Zipfian word vectors surrounding S based on S's local context, which contains at most $2 \times w + 1$ rows for a given window size of w. Then the decision whether T is the translation of S is based on the two dimensional (2D) weight decaying Pearson correlation coefficient of their corresponding Zipfian word matrices.

Weight decaying is applied to the sentences that are far from S, which is the sentence according to which the context is calculated. Exponential decaying is applied with decaying constant set to 0.7. The use of weight decaying for 2D Pearson correlation coefficient does not improve statistically significantly, but it increases the accuracy and decreases the variance; hence giving us a more robust value.

3.4 Context in Sentence Alignment

The sentence alignment algorithm we have developed is context-based in the sense that features belonging to the sentences that come before and after the current sentence are also considered. We represent the local context of a given set of sentences as a pair, the number of sentences to consider before and after the given set of sentences.

We consider two options for context size selection: (i) static context size selection, which uses (w, w) for both the source and the target local contexts; (ii) symmetric local context selection, which uses (b, a) for both the source and the target local contexts, where b and a correspond to the number of sentences that come before and after a given sentence. For a given context window size limit, w, where $2w = b + a$, there can be w^2 symmetric local context selections for the pair S and T. For the Lithuanian-English pair on the first chapter when w=10 the maximum score attaining symmetric local context is found to be $(3.12, 3.35)$.

3.5 Sentence Alignment Algorithm

Our sentence alignment algorithm makes use of dynamic programming formulation with up to 6-way alignments with extensions to handle non-monotonic alignments. The algorithm is essentially a modified version of the Needleman-Wunsch sequence alignment algorithm [17] with gap penalty set to -0.5. Further discussion on dynamic programming methodology to solve sentence alignment problems can be found in [1] or in [5]. We use the assumption that the alignments are found close to the diagonal of the dynamic programming table to further speed up the alignment process. Another property of our system is its ability to model up to 6-way alignments.

Another benefit in using sequence alignment methodology is our ability to model not only constant gap costs in the alignments but also affine as well as convex gap costs (a good description for affine and convex gap costs is in [18]). However, as the dataset does not provide enough contiguous gaps, we have not tested this capability; yet it is likely that affine and convex gap costs model the gap costs in sentence alignment better.

3.6 Non-monotonic Alignments

Previous approaches assume that sentence alignment functions are monotonic. We relax this assumption and assume monotonicity in the flow of the arguments (semantic monotonicity) within a 4 sentence window from both the source and target corpus. Thus, we also allow non-monotonic alignments of type $([1,2],[1,2] \times [1,2],[1,2])$, where $[1,2]$ corresponds to 1 or 2 sentences that are selected and $\times$ means that $\cdot_1$ is aligned with $\cdot_4$ and $\cdot_2$ with $\cdot_3$ in $(\cdot_1, \cdot_2 \times \cdot_3, \cdot_4)$. Although the Multext-East dataset does not contain non-monotonic alignment examples, we have observed that by allowing non-monotonicity, in some cases our system is able to backtrack and recover from errors that were incurred from previous steps and therefore reducing noise.

4 Experiments

We used the George Orwell's 1984 corpus from Multext-East [3], which contains manually sentence split and aligned translations for English, Bulgarian, Czech, Estonian, Hungarian, Romanian, Latvian, Lithuanian, Serbo-Croatian, and Slovene, which are abbreviated as `en`, `bg`, `cs`, `et`, `hu`, `ro`, `lv`, `lt`, `sr`, and `sl` respectively. In all of our experiments, the corresponding language pair is chosen to be English. We compared the results of our system with that of hunalign [19] and Moore's system [6]. Without an input dictionary, hunalign makes use of the Gale and Church [1] algorithm which is based on sentence lengths, and builds a dictionary dynamically based on this alignment.

We calculated the Pearson correlation coefficient score for the sentence lengths in characters and in word tokens for the datasets in the full parallel corpus based on the correct alignments. The correlation coefficients for the sentence lengths in characters are found as: (bg, 0.9692), (cs, 0.9674), (et, 0.9738), (hu, 0.9585), (lt, 0.9719), (ro, 0.9730), (sr, 0.9694), (sl, 0.9805). The correlation coefficients for the sentence lengths in word tokens are found as: (bg, 0.9598), (cs, 0.9542), (et, 0.9489), (hu, 0.9440), (lt, 0.9554), (ro, 0.9578), (sr, 0.9575), (sl, 0.9669). The total number of alignments is 2733 and sentences is 5596 in the first chapter of Multext-East, which gives a moderately sized corpora, when all languages are combined. These numbers rise to 52594 for alignments and 106504 for sentences when the full dataset is used.

Our first couple of experiments are based on choosing appropriate parameters. Fig. 2 show the change in sentence alignment accuracy with varying window sizes in the first chapter of the corresponding corpora in Multext-East (the corresponding graph for the sentence alignment coverage has similar trends). Based on these graphs, we chose the window size w to be 7. To reduce the complexity of calculations to a manageable value, the value of b is chosen to be 10.

We have experimented with using different sentence length definitions for the Zipfian word vectors used in the system. The alternatives that we consider are: (i) no sentence length information, (ii) only the sentence length in the number of characters added, (iii) only the sentence length in the number of word tokens added, (iv) both added. The results show that when w=7, adding only the sentence length in the number of tokens perform better than other alternatives that we considered and the performance decreases in this order: (iii) $\gg$ (iv) $\gg$ (ii) $\gg$ (i).

In the first chapter of the dataset, our context-based sentence alignment algorithm (CBSA), CBSA when the average score for the symmetric local contexts is used (CBSAavg), and non-monotonic CBSA (nmCBSA) sentence alignment techniques reduce the sentence alignment accuracy error rate of hunalign by 1.3611, 1.2895, and 1.2149 times and of Moore by 1.5648, 1.4825, and 1.3967 times respectively. The results can be seen in Table 1. In terms of sentence alignment coverage, CBSA, CBSAavg, and nmCBSA reduce the error rate of hunalign by 1.5401, 1.4602, and 1.2827 and of Moore by 1.6022, 1.5190, and 1.3343 times respectively. The results can be seen in Table 3.

The results on the full dataset are presented in Table 2 and in Table 4. Our CBSA and nmCBSA sentence alignment techniques reduce the sentence alignment accuracy error rate of hunalign by 2.0243 and 1.4360 times and increase the error rate of Moore by 1.7404 and 2.4534 times respectively. In terms of sentence alignment coverage, CBSA

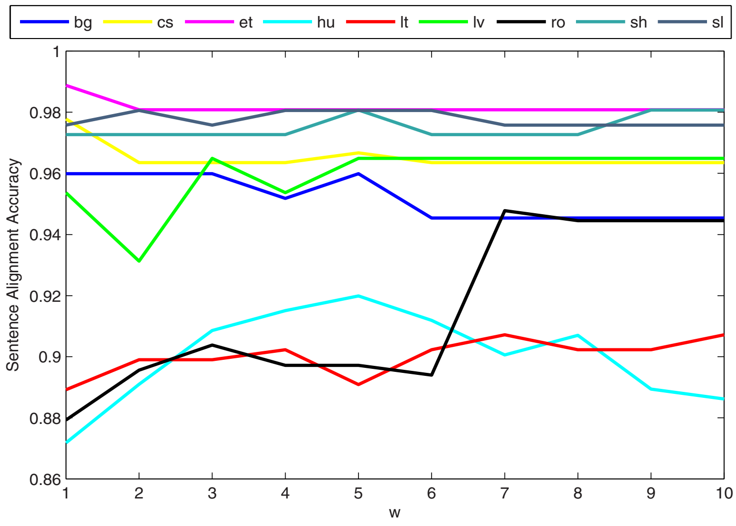


Fig. 2. Sentence Alignment Accuracy versus Window Size w

Table 1. Sentence alignment accuracy per English - language alignments on the first chapter

First Chapter	**Sentence Alignment Accuracy**				
Language	**hunalign**	**Moore**	**CBSA, w=7**	**CBSA, w=7, average**	**nmCBSA, w=7**
Bulgarian	**96.74 / 3.26**	96.09 / 3.91	95.44 / 4.56	**96.74 / 3.26**	95.11 / 4.89
Czech	96.14 / 3.86	95.82 / 4.18	**96.78 / 3.22**	**96.78 / 3.22**	**96.78 / 3.22**
Estonian	**99.68 / 0.32**	98.39 / 1.61	98.39 / 1.61	98.39 / 1.61	98.39 / 1.61
Hungarian	87.86 / 12.14	88.96 / 11.04	91.64 / 8.36	**92.98 / 7.02**	91.30 / 8.70
Latvian	95.71 / 4.29	92.74 / 7.26	**97.69 / 2.31**	**97.69 / 2.31**	**97.69 / 2.31**
Lithuanian	88.44 / 11.56	82.31 / 17.69	**92.52 / 7.48**	91.84 / 8.16	92.18 / 7.82
Romanian	89.86 / 10.14	95.27 / 4.73	**95.61 / 4.39**	91.22 / 8.78	92.23 / 7.77
Serbo-Croatian	**98.70 / 1.30**	97.08 / 2.92	97.73 / 2.27	97.73 / 2.27	97.73 / 2.27
Slovene	97.70 / 2.30	97.04 / 2.96	98.36 / 1.64	**98.68 / 1.32**	98.36 / 1.64

Table 2. Sentence alignment accuracy per English - language alignments on the full dataset

Full Dataset	**Sentence Alignment Accuracy**				
Language	**hunalign**	**Moore**	**CBSA, w=4**	**CBSA, w=7**	**nmCBSA, w=7**
Bulgarian	72.92 / 27.08	**98.77 / 1.23**	98.23 / 1.77	98.29 / 1.71	97.40 / 2.60
Czech	95.65 / 4.35	**97.92 / 2.08**	95.92 / 4.08	96.18 / 3.82	94.89 / 5.11
Estonian	96.88 / 3.12	**98.36 / 1.64**	97.02 / 2.98	97.13 / 2.87	95.53 / 4.47
Hungarian	94.77 / 5.23	**97.74 / 2.26**	95.91 / 4.09	96.25 / 3.75	94.47 / 5.53
Lithuanian	95.42 / 4.58	**97.12 / 2.88**	95.60 / 4.40	95.65 / 4.35	93.60 / 6.40
Romanian	92.50 / 7.50	**96.52 / 3.48**	92.39 / 7.61	92.55 / 7.45	90.75 / 9.25
Serbo-Croatian	97.21 / 2.79	**98.40 / 1.60**	97.53 / 2.47	97.54 / 2.46	96.06 / 3.94
Slovene	97.91 / 2.09	**98.96 / 1.04**	97.98 / 2.02	98.15 / 1.85	97.47 / 2.53

Table 3. Sentence alignment coverage per English - language alignments on the first chapter

First Chapter	**Sentence Alignment Coverage**				
Language	**hunalign**	**Moore**	**CBSA, w=7**	**CBSA, w=7, average**	**nmCBSA, w=7**
Bulgarian	95.34 / 4.66	94.86 / 5.14	94.54 / 5.46	**95.99 / 4.01**	90.54 / 9.46
Czech	94.92 / 5.08	95.24 / 4.76	**96.35 / 3.65**	**96.35 / 3.65**	**96.35 / 3.65**
Estonian	**99.52 / 0.48**	98.08 / 1.92	98.08 / 1.92	98.08 / 1.92	98.08 / 1.92
Hungarian	84.30 / 15.70	85.90 / 14.10	90.06 / 9.94	**91.51 / 8.49**	89.74 / 10.26
Latvian	92.65 / 7.35	90.26 / 9.74	**96.49 / 3.51**	**96.49 / 3.51**	**96.49 / 3.51**
Lithuanian	84.85 / 15.15	79.15 / 20.85	**90.72 / 9.28**	89.90 / 10.10	90.39 / 9.61
Romanian	86.79 / 13.21	93.64 / 6.36	**94.78 / 5.22**	89.72 / 10.28	90.54 / 9.46
Serbo-Croatian	**97.75 / 2.25**	96.46 / 3.54	97.27 / 2.73	97.27 / 2.73	97.27 / 2.73
Slovene	95.81 / 4.19	95.64 / 4.36	97.58 / 2.42	**98.06 / 1.94**	97.58 / 2.42

Table 4. Sentence alignment coverage per English - language alignments on the full dataset

Full Dataset	**Sentence Alignment Coverage**				
Language	**hunalign**	**Moore**	**CBSA, w=4**	**CBSA, w=7**	**nmCBSA, w=7**
Bulgarian	72.72 / 27.28	**98.65 / 1.35**	98.20 / 1.80	98.27 / 1.73	97.37 / 2.63
Czech	94.39 / 5.61	**97.52 / 2.48**	95.51 / 4.49	95.80 / 4.20	94.42 / 5.58
Estonian	95.67 / 4.33	**97.98 / 2.02**	96.44 / 3.56	96.56 / 3.44	94.86 / 5.14
Hungarian	93.60 / 6.40	**97.12 / 2.88**	95.41 / 4.59	95.77 / 4.23	93.89 / 6.11
Lithuanian	94.07 / 5.93	**96.50 / 3.50**	94.97 / 5.03	95.04 / 4.96	92.92 / 7.08
Romanian	89.85 / 10.15	**95.54 / 4.46**	91.19 / 8.81	91.35 / 8.65	89.43 / 10.57
Serbo-Croatian	96.60 / 3.40	**98.17 / 1.83**	97.37 / 2.63	97.38 / 2.62	95.83 / 4.17
Slovene	97.15 / 2.85	**98.72 / 1.28**	97.68 / 2.32	97.86 / 2.14	97.18 / 2.82

Table 5. Comparing the performance of different sentence alignment methods

	Alignment Method	**Alignment Mistakes**	**Alignments Per Mistake**	**Sentence Mistakes**	**Sentences Per Mistake**
	hunalign	147	18.55	422	13.26
First	**Moore**	169	16.17	439	12.75
Chapter	**CBSA, w=7**	**108**	**25.31**	**274**	**20.43**
	CBSA, w=7, average	114	23.97	289	19.37
	nmCBSA, w=7	121	22.58	329	17.02
	hunalign	3745	14.04	8788	12.12
Full	**Moore**	**1063**	**49.50**	**2635**	**40.42**
Dataset	**CBSA, w=7**	1850	28.44	4250	25.06
	nmCBSA, w=7	2608	20.16	5864	18.16

and nmCBSA reduce the error rate of hunalign by 2.0678 and 1.4986 times and increase the error rate of Moore by 1.6129 and 2.2254 times respectively.

The comparison of the performance of different sentence alignment techniques is presented in Table 5. The reason for the worse performance of CBSA than Moore's system in the full dataset can be explained by the contribution of the word translation models that the Moore's system builds. Thus, given a moderately sized corpus with which to work with, hunalign as well as Moore's system will likely to have a bad performance since the word translation probabilities and the dictionaries that their systems build respectively will be more error-prone. Therefore, it might be a good idea to incorporate word translation models to the CBSA system when working with large corpora.

5 Conclusion

We have developed a novel language-independent context-based sentence alignment technique given parallel corpora. We can view the problem of aligning sentences as finding translations of sentences chosen from different sources. Unlike current approaches which rely on pre-defined features and models, our algorithm employs features derived from the distributional properties of words in sentences and does not use any language dependent knowledge. The resulting sentence alignment methodology is language-independent; it can handle non-monotonic alignments; it does not require any stemming, dictionaries, or anchors; it handles deletions; and it extends the type of alignments available up to 6-way.

The main advantage of Moore's and Chen's methods are their employment of the word translation probabilities and their updates when necessary. It is a custom to feed previous alignment results back into the aligner to further improve on the results. This process is generally referred to as bootstrapping and there may be multiple passes needed until convergence. We can easily improve our model by making use of word translation models and bootstrapping.

We provide formalizations for sentence alignment task and the context for sentence alignment. We use the notion of Zipfian word vectors which effectively presents an order-free representation of the distributional properties of words in a given sentence. We define two dimensional weight decaying correlation scores for calculating the similarities between sentences.

We accept the context to be the frame in which the reasoning about sentence alignment is done. We can also further improve our model by using a pre-specified dictionary, by dynamically building a dictionary, by using stemming, by using a larger corpus to estimate frequencies and generating Zipfian word vectors based on them, by using larger window sizes to select the local context size from, or by using bootstrapping which makes use of the previously learned alignments in previous steps.

We evaluate the performance of our system based on two different measures: sentence alignment accuracy and sentence alignment coverage. We compare the performance of our system with commonly used sentence alignment systems and show that our system performs 1.2149 to 1.6022 times better in reducing the error rate in alignment accuracy and coverage for moderately sized corpora. The addition of word translation probabilities and models of word order to the CBSA system is likely to achieve better sentence alignment results when working with large corpora.

Acknowledgments

The research reported here was supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK). The author would like to thank Deniz Yuret for helpful discussions and for guidance and support during the term of this research.

References

1. Gale, W.A., Church, K.W.: A program for aligning sentences in bilingual corpora. Computational Linguistics 19, 75–102 (1993)
2. Kruskal, J.B.: An overview of sequence comparison. In: Sankoff, D., Kruskal, J.B. (eds.) Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison, pp. 1–44. Addison-Wesley, Reading (1983)
3. Erjavec, T.: MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In: Fourth International Conference on Language Resources and Evaluation, LREC 2004, Paris, ELRA, pp. 1535–1538 (2004), `http://nl.ijs.si/et/Bib/LREC04/`
4. Brown, P.F., Lai, J.C., Mercer, R.L.: Aligning sentences in parallel corpora. In: Proceedings of the 29th annual meeting on Association for Computational Linguistics, pp. 169–176. Association for Computational Linguistics, Morristown, NJ, USA (1991)
5. Chen, S.F.: Aligning sentences in bilingual corpora using lexical information. In: Proceedings of the 31st annual meeting on Association for Computational Linguistics, pp. 9–16. Association for Computational Linguistics, Morristown, NJ, USA (1993)
6. Moore, R.C.: Fast and accurate sentence alignment of bilingual corpora. In: Richardson, S.D. (ed.) AMTA 2002. LNCS (LNAI), vol. 2499, pp. 135–144. Springer, Heidelberg (2002)
7. Brown, P.F., et al.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics 19, 263–311 (1993)
8. Knight, K.: A statistical machine translation tutorial workbook (1999), `http://www.isi.edu/natural-language/mt/wkbk.rtf`
9. Yarowsky, D.: Decision lists for lexical ambiguity resolution. In: Hayes-Roth, B., Korf, R. (eds.) Proceedings of the Twelfth National Conference on Artificial Intelligence. American Association for Artificial Intelligence, AAAI Press, Menlo Park (1994)
10. Yarowsky, D., Florian, R.: Evaluating sense disambiguation across diverse parameter spaces. Natural Language Engineering 8, 293–310 (2002)
11. Wang, X.: Robust utilization of context in word sense disambiguation. In: Dey, A., et al. (eds.) Modeling and Using Context: 5th International and Interdisciplinary Conference, pp. 529–541. Springer, Berlin (2005)
12. Ristad, E.S., Thomas, R.G.: New techniques for context modeling. In: ACL, pp. 220–227 (1995)
13. Biçici, E.: Local context selection for aligning sentences in parallel corpora. In: Kokinov, B., et al. (eds.) CONTEXT 2007. LNCS (LNAI), vol. 4635, pp. 82–93. Springer, Heidelberg (2007)
14. Zipf, G.K.: The meaning-frequency relationship of words. The Journal of General Psychology 33, 251–256 (1945)
15. Treebank, P., Marcus, M.P., Marcinkiewicz, M.A.: Building a large annotated corpus of english: The penn treebank (2004)
16. Joachims, T.: Learning to Classify Text using Support Vector Machines. Kluwer Academic Publishers, Dordrecht (2002)
17. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarity in the amino acid sequences of two proteins. J. Mol. Biol. 48, 443–453 (1970)
18. Gusfield, D.: Algorithms on Strings, Trees, and Sequences. Cambridge University Press, Cambridge (1997)
19. Varga, D., et al.: Parallel corpora for medium density languages. In: Proceedings of the Recent Advances in Natural Language Processing 2005 Conference, Borovets, Bulgaria, pp. 590–596 (2005), Comment: hunalign is available at `http://mokk.bme.hu/resources/hunalign`

Bilingual Segmentation for Alignment and Translation

Chung-Chi Huang, Wei-Teh Chen, and Jason S. Chang

Information Systems and Applications, NTHU, HsingChu, Taiwan 300 R.O.C.
{u901571,weitehchen,jason.jschang}@gmail.com

Abstract. We propose a method that bilingually segments sentences in languages with no clear delimiter for word boundaries. In our model, we first convert the search for the segmentation into a sequential tagging problem, allowing for a polynomial-time dynamic-programming solution, and incorporate a control to balance monolingual and bilingual information at hand. Our bilingual segmentation algorithm, the integration of a monolingual language model and a statistical translation model, is devised to tokenize sentences more suitably for bilingual applications such as word alignment and machine translation. Empirical results show that bilingually-motivated segmenters outperform pure monolingual one in both the word-aligning (12% reduction in error rate) and the translating (5% improvement in BLEU) tasks, suggesting monolingual segmentation is useful in some aspects but, in a sense, not built for bilingual researches.

1 Introduction

A statistical translation model (STM) is a model that, relied on lexical information or syntactic structures of languages involved, decodes the process of human translation and that, in turn, detects most appropriate word correspondences in parallel sentences. Ever since the pioneer work of (Brown et al., 1993), the field of STMs has drawn myriads of attention. Some researchers exploited Hidden Markov models to approach relatively monotonic word-aligning problems in similarly-structured language pairs (Vogel et al., 1996; Toutanove et al., 2002) while, to make a STM more tolerant of the input of distantly-related language pairs, some integrated structural facets of languages, such as flattened source-language (SL, like English) parse tree (Yamada and Knight, 2001), inversion transduction grammar rules consisting of binary rewrite rules with word ordering preference on target-language (TL, like Chinese) end (Wu, 1997; Zhang and Gildea, 2005), and context-motivated dependency features (Cherry and Lin, 2003).

Determining the corresponding translation(s) of a word in sentence pairs is not straightforward. Still worse, a language with no obvious delimiter for word boundaries would further impose uncertainty on the quality of performance and increase the complexity of the problem. Conventionally, researchers resort to some segmenters to tokenize sentences in such language before sentences are fed into any STM. Nonetheless, those tools, developed monolingually, segment sentences without taking their translations into consideration. That is, resulted segmentations in this language might be under- or over-segmented for the corresponding words in another, which would degrade the word-aligning performance since the one-to-many or many-to-one relation of alignments is harder to learn and capture than one-to-one and, in turn, make more vulnerable the subsequent machine translation model utilizing these vaguely-acquired one-to-many, many-to-one and intertwined many-to-many translation pairs in that unevenly-segmented tokens for counterparts raise the chance of encountering unknown words or of producing inappropriate translations (translating too much or too little of information). As a result, segmentation does cast a whole other issue in bilingual applications if one of the two languages provides no direct information of word boundaries from the plain texts.

We are inspired by the idea that if the cases of one-to-one correspondences in both languages occur more often, the odds are the modeling would be simpler and that the performance could be further improved for alignment and translation. To make one-to-one linkages more frequent, the substrings in the sentences need to be determined whether or not they had contained enough information for counterparts in alignment, or had reached the basic translation unit of the target language. Insufficient is depending only on monolingual data to make those decisions. In this paper, monolingual together with bilingual information, supplied by a STM, is leveraged to segment sentences such that there exists more one-to-one word bonds and that the segmentations, in accordance with its partner language's words, would be more built for alignment and translation.

2 The Model

This section begins with an example illustrating tokens segmented monolingually are not proper for bilingual researches. Afterwards, a bilingually-oriented segmenting algorithm and a training process are described in detail.

2.1 An Example

For simplicity, the English-French notation is used throughout this paper. *E* and *F* denote the source (English) and the target (Chinese) language, respectively, and e_i stands for the i-th word in sentence *e* in language *E* and f_j for the j-th character in sentence *f* in *F*.

Figure 1(a) shows an English sentence *e* with its Chinese translation *f* segmented using some segmenting tool trained monolingually, while the same *e* with Chinese translation segmented manually according to the corresponding English words is illustrated in figure 1(b). In figure 1, spaces are used to discriminate boundaries of tokens in both languages and solid lines are the manually-done word alignments. The difference is, in figure 1(a), the monolingual segmenter tokenizes “一國兩制” and “鄧小平” as meaningful and understandable words in Chinese whereas smaller segments of these two in figure 1(b) (“一”, ”國”, ”兩”, ”制”, ”鄧” and ”小平”) lead to perfectly one-to-one word-aligning results over the sentence pair.

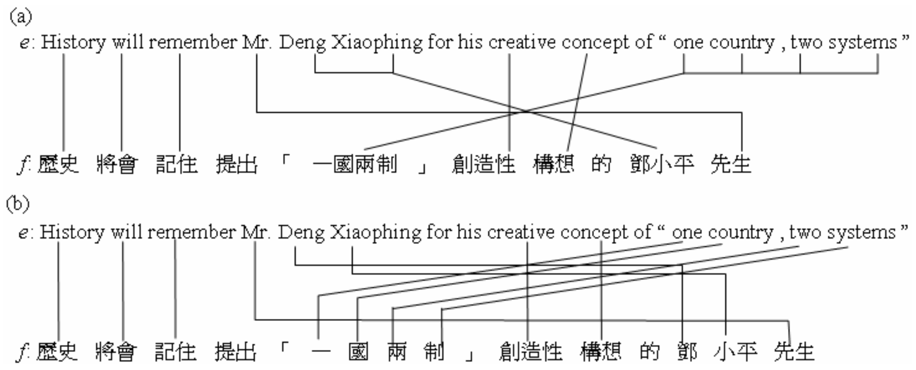

(a)
e: History will remember Mr. Deng Xiaophing for his creative concept of " one country , two systems "
f: 歷史 將會 記住 提出 「 一國兩制 」 創造性 構想 的 鄧小平 先生
(b)
e: History will remember Mr. Deng Xiaophing for his creative concept of " one country , two systems "
f: 歷史 將會 記住 提出 「 一 國 兩 制 」 創造性 構想 的 鄧 小平 先生

Fig. 1. A sentence pair with different segmentations in Chinese

In spite of the fact that the Chinese segmentation in figure 1(a) is tangible and accurate in the sense of Chinese, if word aligning is to be performed on the sentence pair, segments of figure 1(b) are more suitable since the translation probabilities (*t*) in table 1, from training on English and monolingually-segmented Chinese parallel texts, suggest the overall probability will decease rapidly whenever each word of the string "one country two system" is aligned to "一國兩制" (thus indicate word aligners might not align "一國兩制" to all of the words in "one country two system"). On the other hand, translation probabilities in table 2, learnt from English and one character per token Chinese sentence pairs, imply "一" tends to align "one", "國" to "country", "兩" to "two" and "制" to "system".

Table 1. Low-probability translation table

e_i	f_j[1]	$t(f_j \mid e_i)$
one	一國兩制	.0184
country	一國兩制	.0441
two	一國兩制	.0173
system	一國兩制	.0131

Table 2. High-probability translation table

e_i	f_j	$t(f_j \mid e_i)$
one	一	.453
country	國	.521
two	兩	.700
system	制	.279

In these two scenarios, the products of the translation probabilities for aligning "一國兩制" to "one country two system" vary enormously (1.8×10^{-6} versus 4.6×10^{-2}), which directly proves the idea that segmentation in figure 1(b) would more fit into the bilingual context of word alignment. Besides, when it comes to translation, the manually-segmented results of "一國兩制" would be much simpler for MT to determine the translation counterparts (with one character translated into an English word).

[1] Here, it stands for a word segment, instead of a character.

2.2 Bilingual Segmentation Algorithm

Based on target language's language model and bilingual STM, our model segments TL sentences such that these bilingually-oriented segmentations make easier finding the corresponding words in source language whether in word-aligning or translating task. In other words, given a sentence pair $(e,f)=(e_1,\cdots e_i\cdots,e_m,f_1,\cdots f_j\cdots,f_n)$, the model is dedicated to tokenize the TL sentence f into f_s^* with tokens most suited in bilingual sense, which means:

$$f_s^* = \arg\max_{f_s} \left\{ \mathrm{P}\left(f_s, \mathbf{A} \middle| e, f\right) \right\} \tag{1}$$

where f_s is any possible segmentation of the original sentence f and a hidden variable **A**, (bilingual) alignment information, is introduced to make the resulted segments in F explicitly dependent on counterparts in E.

Each segmentation result of the sentence, however, can be associated with a corresponding tag sequence, utilizing "B" to represent the starting character of a segment and 'I' the intermediate one(s), and each tag sequence associated with the sentence can help to convert it into a unique segmentation result. Take "一國兩制", the substring of the Chinese sentence above, for example. If the segmenting result of it is "一", "國", "兩" and "制", then the corresponding tag sequence is "BBBB" while if the tag sequence related to it is "BIBI", its segmentation is "一國" and "兩制". Consequently, segmentation modeling can be recast as a tagging problem as follows.

$$\mathbf{T}_f^* = \arg\max_{\mathbf{T}_f} \left\{ \mathrm{P}\left(\mathbf{T}_f, \mathbf{A} \middle| e, f\right) \right\} \tag{2}$$

where $\mathbf{T}_f$ stands for the tag sequence associated with f.

$\mathrm{P}(\mathbf{T}_f, \mathbf{A}|e,f)$ is factored into probabilities of monolingual language model (Chinese language model) with tag information and of bilingual information as:

$$\mathrm{P}\left(\mathbf{T}_f, \mathbf{A} \middle| e, f\right) = \mathrm{P}_{CLM}^{\lambda} \times \mathrm{P}_{BiInfo}^{1-\lambda} \tag{3}$$

where λ, between 0 and 1, is the control to adjusting monolingual segments to fit into bilingual applications. P_{CLM} and P_{BiInfo} are further decomposed as:

$$\mathrm{P}_{CLM} = \mathrm{P}_{CLM}\left(f_1, t_1\right) \prod_{j=2}^{n} \mathrm{P}_{CLM}\left(f_j, t_j \middle| f_{j-1}, t_{j-1}\right) \tag{4}$$

where t_j is the tag ("B" or "I") of the character f_j, and

$$\mathrm{P}_{BiInfo} = \prod_{j=1}^{n} t\left(f_j \middle| e_{a_j}\right) a\left(a_j \middle| j, m, n\right) \tag{5}$$

where translation (t) and alignment (a) probabilities provide the model with the bilingual insights and a_j describing the mapping between f_j and e_{a_j} is a position in e or zero implying f_j aligns to NULL.

From equation (3) to (5), P($\mathbf{T}_f$,$\mathbf{A}$|e,f) is

$$\mathrm{P}_{CLM}^{\lambda}\left(f_1,t_1\right)\prod_{j=2}^{n}\mathrm{P}_{CLM}^{\lambda}\left(f_j,t_j\left|f_{j-1},t_{j-1}\right.\right)\times\prod_{j=1}^{n}\left(t\left(f_j\left|e_{a_j}\right.\right)a\left(a_j\left|j,m,n\right.\right)\right)^{1-\lambda} \tag{6}$$

where tag sequence $\mathbf{T}_f$ comprises t_1, t_2, $\cdots$ and t_n, a_j lies between 0 and m, and e_0 denotes NULL. Nonetheless, the assumption that t_j is independent of the interaction within t_{j-1}, a_{j-1} and a_j is problematic in modeling bilingual segmentation.

A polynomial-time dynamic-programming algorithm, taking into account the tangled relationships among t_{j-1}, t_j, a_{j-1} and a_j, is described below to seek for $\mathbf{T}_f^*$, given a sentence pair. In this algorithm, δ (j,a_j,t_j) denotes the maximal probability of tagging the character f_j as t_j while f_j aligns to e_{a_j}, ε stands for a small probability of unlikely events, and Γ for the tag set (Γ={B,I}).

Bilingual Segmentation Algorithm

```
1. Initialization
   For $0 \leqq a_1 \leqq m$
     For $t_1$ in $\Gamma$
       $\delta(1,a_1,t_1)=\mathrm{P}_{CLM}^{\lambda}(f_1,t_1)\left(t(f_1|e_{a_1})a(a_1|1,m,n)\right)^{1-\lambda}$
2. Recursion
   For $2 \leqq j \leqq n$
     For $0 \leqq a_j \leqq m$
       For $t_j$ in $\Gamma$
         $\delta(j,a_j,t_j)=\max_{0\le a_{j-1}\le m,\ t_{j-1}\in\Gamma}\left\{\delta(j-1,a_{j-1},t_{j-1})\mathrm{P}_{CLM}^{\lambda}(f_j,t_j|f_{j-1},t_{j-1})\mathrm{P}_{BiInfo}^{1-\lambda}\right\}$
   where $\mathrm{P}_{BiInfo}$ is calculated as follows.
   Do Case
       Case ($t_{j-1}$,$t_j$) is (B,B) or (I,B)
         If $a_{j-1}=a_j$ do $\mathrm{P}_{BiInfo}=\varepsilon$
         Else do $\mathrm{P}_{BiInfo}=t(f_j|e_{a_j})a(a_j|j,m,n)$
       Case ($t_{j-1}$,$t_j$) is (B,I) or (I,I)
         If $a_{j-1}=a_j$ do $\mathrm{P}_{BiInfo}=t(f_j|e_{a_j})a(a_j|j,m,n)$
         Else do $\mathrm{P}_{BiInfo}=\varepsilon$
3. Segmentation
   By backtracking, $\mathbf{A}^*=(a_1^*,\cdots,a_n^*)$ and $\mathbf{T}_f^*=(t_1^*,\cdots,t_n^*)$, maximizing P($\mathbf{T}_f$ ,$\mathbf{A}$|$e$,$f$ ), are
   known and so is the $f_s^*$ after $f$ is segmented according to $\mathbf{T}_f^*$.
```

2.3 Training Procedure

In this subsection, we introduce the training procedure of acquiring probabilities related to TL language model and bilingual STM, exploited to segment TL sentences in our bilingual segmenting algorithm, from a sentence-aligned corpus **C**, a set of$(e,f)=(e_1,\cdots e_i\cdots,e_m,f_1,\cdots f_j\cdots,f_n)$. Recall that the target language is assumed to be Chinese and that e_i and f_j represent the i-th word in e and the j-th character in f respectively.

At the first stage of the training process, all TL sentences are tokenized by an existing segmenter so that understandable and meaningful compounds of characters in monolingual sense can be leant. By means of the tag set Γ, tag sequences are attached to the segmented TL sentences before probabilities of unigram, $P_{CLM}(f_j,t_j)$, and bigram, $P_{CLM}(f_j,t_j|f_{j-1},t_{j-1})$, TL language model accompanied with tagging information can be calculated by relative frequency.

At the second stage, IBM model 2 is utilized to estimate (bilingual) translation and alignment probability making up of our STM. Model 2, nonetheless, does not guarantee a proper convergence of probabilities. Hence, we follow Brown's (1993) idea of chaining IBM model 2 after IBM model 1 to obtain more satisfying probabilistic estimations.

3 Experiments

To examine the impact of bilingually-motivated Chinese tokenizations on word alignment and machine translation, experiments were conducted and evaluated by the metrics of alignment error rate (Och and Ney, 2000) and BLEU (Papineni et al.,2002) accordingly.

3.1 Training

We used the first 100,000 sentence pairs, 24 words for English and 35 characters for Chinese on average, of the news portion of Hong Kong Parallel Text (Hong Kong news for short) as our sentence-aligned corpus **C**. During training[2], English sentences are considered to be the source while Chinese sentences are the target, and Chinese sentences are segmented by Ketitool, an existing segmentation tool.

3.2 Tuning λ

Recall that, in our model, λ is a weight for combining monolingual and bilingual information. It must be chosen from the range of 0 to 1 beforehand.

After learning the Chinese language model with tag information and acquiring translation and alignment probabilities, with no prior prejudice, 39 sentence pairs, averaging 29 words in English and 44 characters in Chinese sentences, were set out as development data to tune λ. For selecting a fine-tuned λ, the alignments and segments of the development data produced by our segmentation algorithm were evaluated by the metrics of AER and the recall related to Chinese segments (the number of correct automatically-segmented Chinese tokens with respect to the number of Chinese tokens segmented manually according to the English counterparts). Balanced performance in the two was observed when λ was 0.88.

Table 3, where P is short for precision, R for recall of alignment, F[3] for f measure and RC for recall related to Chinese segmentations, shows the empirical results. For

[2] Iterations of IBM model 1 and 2 were set to be three.

[3] $F = 2\times P\times R / (P+R)$.

simplicity, throughout this section, BS stands for the proposed bilingual segmenter whereas MS for the monolingual one (Ketitool). For comparison, the result of IBM model 2, trained on the same corpus with one character per token in Chinese, was listed in the last row. Also note that the RC score collected from the tokenized Chinese sentences by Ketitool was 0.726, smaller than our model's 0.794, indicating our bilingual segmenter improved the work of Chinese segmentation in terms of (segment) proximity to English.

Table 3. Results on the developing data

	P	R	AER[4]	F	RC
BS	.720	.745	.268	.732	.794
IBM2	.606	.711	.346	.654	.700

3.3 Evaluations

In this subsection, we examine whether or not bilingually-oriented segmentation boosts the quality of word alignment and machine translation.

In the word-aligning task, two different sets of tokenized Chinese sentences paired up with corresponding English sentences were fed into the state-of-the-art word alignment system, GIZA++: one segmented by Ketitool and the other by bilingual segmentation algorithm of ours. In other words, Chinese sentences in corpus **C** were pre-tokenized monolingually and bilingually to inspect the performance of these two in the context of word alignment.

The test set, randomly selected from **C**, of this task consisted of 73 sentence pairs, on average 30 words in English and 43 characters in Chinese. The aligning results on the test data measured by word-to-character (English-to-Chinese) AER are summarized in table 4 where GIZA++ was run with default settings. Compared with MS, BS reduced 12% alignment error rate relatively, indicating the bilingual segmentations on Chinese end did reflect the English counterparts more appropriately.

Table 4. Results of the word-aligning task

	P	R	AER[5]	F
BS	.679	.742	.291	.709
MS	.638	.702	.332	.669

Table 5. Translating results

	BLEU
bigram_seg	.235
MS	.223

In the task of Chinese-to-English translation, on the other hand, 544 Chinese sentences of average 44 characters were randomly chosen from **C** as the test data and the reference translation set was made up of their English counterparts, that is, one reference translation per Chinese sentence. Moreover, the Chinese sentences were

[4] # of sure links / # of possible is 0.987.

[5] # of sure links / # of possible is 0.94.

tokenized using a bigram and BI-tagging segmenter, *bigram_seg*, (one using the bilingual segmentation algorithm in section 2.2, but without the bilingual part) trained on bilingually-motivated segmentations of **C** produced by our bilingual segmenter, or tokenized using Ketitool.

A publicly distributed translation system, pharaoh, was utilized to perform the translation. The target language model pharaoh needs was trained on the English part of the whole Hong Kong news, 739,919 sentences in total, using SRI language modeling toolkit, and we leveraged the alignment output of GIZA++ to yield the phrase translation table.

Table 5 illustrates the quality of translation, measured by case-insensitive matching up to 4-grams, on the test set. The simple segmenter, *bigram_seg*, trained on bilingual segments of our model's output achieved an increase of 1.2 absolute BLEU points compared with that of Ketitool, a segmenter equipped only with monolingual insights. The bilingually-motivated segmenter did segment sentences in such a way that the resulted tokens are more suitable for bilingual applications. Furthermore, if the Chinese sentences were segmented in accordance with their English counterparts using method in section 2.2 and pharaoh was provided with according phrase translation table, the BLEU score is 0.256, 15% relative improvement over MS. The high BLEU score makes us believe a more complicated segmenter will learn more from the bilingual segmenations and in turn do better in translating than *bigram_seg* did.

4 Conclusion and Future Work

A bilingual segmenter, leveraging monolingual language model with tag information, bilingual translation and alignment probabilities, has been described. Experiments indicate the control (λ) does adjust monolingual segmentations into ones more suitable in bilingual context, and that bilingually-motivated segmenats exhibit advantages over pure monolingual ones in word alignment and machine translation. For future work, we would like to explore the influence of various monolingual segmenters provided with our bilingual segmentations on performance of translation.

References

1. Brown, P.F., et al.: The mathematics of statistical machine translation: parameter estimation. Computational Linguistics, 263–311 (1993)
2. Cherry, C., Lin, D.: A probability model to improve word alignment. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, pp. 88–95 (2003)
3. Chiang, D.: A hierarchical phrase-based model for statistical machine translation. In: Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics, pp. 263–270 (2005)
4. Liu, Y., Liu, Q., Lin, S.: Tree-to-string alignment template for statistical machine translation. In: Proceedings of the Annual Conference of the Association for Computational Linguistics, pp. 609–616 (2006)
5. Och, F.J., Ney, H.: Improved statistical alignment models. In: Proceedings of the Annual Conference of the Association for Computational Linguistics, pp. 440–447 (2000)

6. Toutanova, K., Ilhan, H.T., Manning, C.D.: Extentions to HMM-based statistical word alignment models. In: Proceedings of the Conference on Empirical Methods in Natural Processing Language (2002)
7. Vogel, S., Ney, H., Tillmann, C.: HMM-based word alignment in statistical translation. In: Proceedings of the 16th conference on Computational linguistics, pp. 836–841 (1996)
8. Wu, D.: Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. Computational Linguistics, 377–403 (1997)
9. Yamada, K., Knight, K.: A syntax-based statistical translation model. In: Proceedings of the Annual Conference of the Association for Computational Linguistics (2001)
10. Zen, R., Ney, H.: A comparative study on reordering constraints in statistical machine translation. In: Proceedings of the Annual Conference of the Association for Computational Linguistics, pp. 144–151 (2003)
11. Zhang, H., Gildea, D.: Stochastic lexicalized inversion transduction grammar for alignment. In: Proceedings of the Annual Meeting of the ACL, pp. 475–482 (2005)

Dynamic Translation Memory: Using Statistical Machine Translation to Improve Translation Memory Fuzzy Matches

Ergun Biçici[1] and Marc Dymetman[2]

[1] Koç University (Istanbul, Turkey)
ebicici@ku.edu.tr
[2] Xerox Research Centre Europe (Grenoble, France)
marc.dymetman@xrce.xerox.com

Abstract. Professional translators of technical documents often use Translation Memory (TM) systems in order to capitalize on the repetitions frequently observed in these documents. TM systems typically exploit not only complete matches between the source sentence to be translated and some previously translated sentence, but also so-called *fuzzy matches*, where the source sentence has some substantial commonality with a previously translated sentence. These fuzzy matches can be very worthwhile as a starting point for the human translator, but the translator then needs to manually edit the associated TM-based translation to accommodate the differences with the source sentence to be translated. If part of this process could be automated, the cost of human translation could be significantly reduced. The paper proposes to perform this automation in the following way: a phrase-based Statistical Machine Translation (SMT) system (trained on a bilingual corpus in the same domain as the TM) is combined with the TM fuzzy match, by extracting from the fuzzy-match a large (possibly gapped) bi-phrase that is dynamically added to the usual set of "static" bi-phrases used for decoding the source. We report experiments that show significant improvements in terms of BLEU and NIST scores over both the translations produced by the stand-alone SMT system and the fuzzy-match translations proposed by the stand-alone TM system.

1 Introduction

Translation Memory (TM) systems [1,2] have become indispensable tools for professional translators working with technical documentation. Such documentation tends to be highly repetitive, due to several factors, such as multiple versioning of similar products, importance of maintaining consistent terminology and phraseology, and last but not least, simplification of the translation process itself. TM systems typically exploit not only *complete matches* between the source sentence to be translated and some previously translated sentence, but also so-called *fuzzy matches* [3], where the source sentence has some substantial commonality with a previously translated sentence. These fuzzy matches can be extremely useful as a starting point for a human translator, but the translator then needs to manually edit the associated TM-based translation in order to accommodate the differences with the source sentence under translation. Typically, the amount of editing required depends on the *fuzzy-match level*, which can be defined as the percentage of shared words between the new source sentence and the previously translated source sentence. Actually, some translation services use these fuzzy-match levels as a basis for estimating the cost of translations both to the freelance translators they employ and to their end-customers; below a certain fuzzy-match level (say 50%) the translation cost of a sentence depends only on the number of words, whereas for a sentence close to a complete match (∼100% fuzzy match), the cost is typically a small fraction of this price.

If part of the manual editing for the TM-based translations could be automated, then the cost of translation could be significantly reduced. This paper proposes such an automation, and describes a hybrid TM-SMT system that works along the following lines. We start by training a phrase-based Statistical Machine Translation (SMT) system on the bilingual corpus associated with the TM. Then, to translate a given source sentence S, the system first retrieves a fuzzy-match S' for S from the TM along with its associated target sentence T', and then uses these for *biasing* the translation candidates produced by the SMT system towards translations "compatible" with the pair (S', T').

Our approach capitalizes on the ability of the underlying SMT system to use *non-contiguous*, or *gapped*, phrases, where the fuzzy-match pair (S', T') is used to dynamically augment the collection of bi-phrases normally used by the SMT system for translating S, by adding a bi-phrase (P_S, P_T), where P_S is a (possibly non-contiguous) common subsequence of S and S', and where P_T is the (possibly non-contiguous) subsequence of T' which is detected to be aligned with P_S.

Using this approach, our hybrid translation system achieves significant improvements in BLEU and NIST scores over both the translation proposed by the stand-alone SMT system and the translation proposed by the stand-alone TM-based system. While these results do not necessarily translate directly into increased usability by the human translators, they do suggest important potential cost reductions for users of Translation Memory tools.

The remainder of the paper is organized as follows. In Section 2 we briefly describe the specificities of the underlying phrase-based SMT system that we use. In Section 3 we detail the ways in which we detect and use fuzzy-matches in order to construct a dynamic bi-phrase which is then added to the "static" biphrases normally used by the SMT system. In Section 4 we present our experimental results. In Section 5 we describe related research, and finally we conclude.

2 The SMT System MATRAX

The phrase-based SMT system MATRAX [4], developed at Xerox, was used in all experiments. MATRAX is based on a fairly standard log-linear model of the form:

$$\Pr(t, a|s) = 1/Z_s \exp \sum_{m=1}^{M} \lambda_m \phi_m(s, t, a),$$

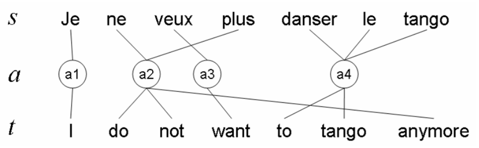


Fig. 1. A MATRAX translation making use of the set of bi-phrases $a = \{$*'Je'/'I'*, *'veux'/'want'*, *'danser le tango'/'to tango'*, *'ne ♢ plus'/'not ♢ ♢ ♢ anymore'* $\}$

where the notation is as follows: s is the source sentence to be translated, t a candidate target sentence, and a an "alignment" between the two, namely a decomposition of (s, t) into a set of "bi-phrases" (see Figure 1); the ϕ_m's are real-valued *features* each representing the assesment of a translation candidate (s, t, a) relative to some particular dimension, such as whether a is composed of high-probability bi-phrases (estimated on a large bilingual corpus), or whether the target sentence t is well-formed according to a certain language model, and so forth. The λ_m's are weights balancing the contributions of each aspect, trained from a small bilingual development corpus, and finally Z_s is a normalization factor. When given a test source sentence s to translate, the so-called *decoder* attempts to find the pair (t, a) which maximizes $\Pr(t, a|s)$ ["Viterbi decoding"], and outputs the translation t.

One original aspect of MATRAX is the use of non-contiguous bi-phrases. Most existing phrase-based models (see [5,6]) depend on phrases that are sequences of contiguous words on either the source or the target side (e.g. *'prendre feu'* / *'catch fire'*). By contrast, MATRAX considers pairs of non-contiguous phrases, such as *'ne ... plus'/'not ... anymore'*, where words in the source and target phrases may be separated by gaps, to be filled at translation time by lexical material provided by some other such pairs. One motivation behind this approach is that, basically, the fact that the source expression *'ne ... plus'* is a good predictor of *'not ... anymore'* does not depend on the lexical material appearing inside the source expression, an insight which is generally unexploitable by models based on contiguous phrases.

These bi-phrases are estimated on the basis of a large training corpus of aligned bi-sentences (s, t). As a first step in this process, the training procedure invokes GIZA++ [7], which has the effect of producing a matrix describing probable *word-level* alignments between the sentences s and t (see section 3.5 for details about this process). Then, through a certain technique of "non-negative matrix factorization" [8], words that show strong affinities are grouped together into bi-phrases. One particularity of this approach is that it inherently produces gapped bi-phrases. These bi-phrases are then stored in a bi-phrase database, that we will refer to in the sequel as the "static bi-phrase library", along with some intrisic features of each bi-phrase $(\tilde{s}, \tilde{t})$, the most important of which is ϕ_{phr}, which is an estimate of the conditional probability $\Pr(\tilde{t}|\tilde{s})$.

3 Dynamic Translation Memory (DTM)

3.1 Experimental Setup

Before going into the details of our hybrid SMT-TM system, *Dynamic Translation Memory (DTM)*, it will be convenient to introduce our experimental setup. We start from a Background Translation Memory (TMB) data base provided to us by Xerox's GKLS (Global Knowledge and Language Services), a branch of the company that produces translations for internal and external customers. The translation memory tool used is TRADOS [9] and the TMB contains around 400,000 English-to-French translation units (typically sentences) in the automotive domain.

We start by randomly partitioning the translation memory into two subsets TM_1 and TM_2, representing respectively 90% and 10% of the translation units in the TMB. TM_1 will from now on be our "effective" translation memory, while TM_2 will serve as a reference for testing the performance of the different systems.

More precisely, we will consider various subsets TM_{FML}, where TM_{FML} is the set of bi-sentences (S, T) in TM_2 such that S has a fuzzy-match *relative to* TM_1 at a certain Fuzzy-Match Level (FML) (e.g. TM_{90-95}). We will then take T as the reference translation, and compare the system performances relative to that reference, for different fuzzy-match levels.

3.2 DTM Overview

The overview of our hybrid system is given below:

1. We first take TM_1 as our bilingual corpus and train the SMT system MATRAX on it;
2. As a by-product of this training, we produce, for each bi-sentence (S', T') in TM_1, a word-alignment matrix where each word pair is associated with a number representing the strength of the association;
3. When given a test source sentence S to translate, we then:
   (a) Attempt to find a fuzzy-match S' for S among the source sentences in TM_1;
   (b) Compute a (possibly gapped) source phrase P_S such that P_S is a subsequence of both S and S';
   (c) Retrieve the word-alignment matrix precomputed for (S', T'), where T' is the target sentence associated with S' in TM_1;
   (d) Use the word alignment matrix to recover a (possibly gapped) target phrase P_T which: (i) is a subsequence of T', and (ii) is strongly aligned with P_S according to the word alignment matrix;
   (e) Start calling the MATRAX decoder on S, which has the consequence of retrieving from MATRAX's static bi-phrase library a collection of (typically small) candidate bi-phrases for decoding S;
   (f) Add to that collection the "dynamic" bi-phrase (P_S, P_T), and also assign to it a strong "weight" through its bi-phrase feature ϕ_{phr};
   (g) Resume the decoder call with the updated bi-phrase collection, and obtain a translation T.

3.3 Example

We will now detail some aspects of this process, using as an illustration the translation of the test sentence S: *'Install the upper arm front retaining bolt in three stages.'* While this sentence is not found literally in TM_1, there is a fuzzy-match for it there (at a fuzzy match level of 82%), namely S': *'Install the lower arm front retaining bolt in two stages.'*

Anticipating on the procedure that we will describe next, here are the translation results for S obtained by three systems: TRADOS (stand-alone TM-based translation system), MATRAX (stand-alone SMT system), and MATRAX-DTM (our hybrid system):

TRADOS *'Poser la vis de fixation avant de bras inférieur en deux passes.'*
MATRAX *'Poser le bras supérieur avant vis de fixation dans trois passes.'*
MATRAX-DTM *'Poser la vis de fixation avant de bras supérieur en trois passes.'*

Here, while the MATRAX-DTM translation is correct, the TRADOS translation is identical with the target T' associated with the fuzzy match S' in TM_1 and does not account for the differences between *'lower'*/*'upper'* (*'inférieur'*/*'supérieur'*) or *'two'*/*'three'* (*'deux'*/*'trois'*); as for the MATRAX translation, while it does get the translations of *'upper'* and *'three'* correctly, it confuses the order of several words and also uses the preposition *'dans'* while in this context the preposition *'en'* is called for.

3.4 Fuzzy-Matcher

While we initially thought of using TRADOS's internal fuzzy matcher to provide us with a fuzzy match S' for S, we finally preferred to implement our own matcher, for two main reasons: (i) this gives us greater flexibility, and in particular we use our matcher for producing a subsequence of the source sentence which is later useful for the dynamic bi-phrase extraction; (ii) TRADOS does not directly provide to the end-user the source sentences from the TM that it uses for producing approximate targets in case of fuzzy matches, but only the targets themselves (such as the one shown in the previous section); furthermore in certain cases TRADOS appears to be able to use two different translation units in order to produce the approximate targets, and in such cases it would not make sense to ask for *one* TRADOS fuzzy-match source S'.[1]

MATRAX-DTM initially indexes the translation units in TM_1 for faster retrieval, using an inverted word index. The candidates for the fuzzy match are selected among the first N (e.g. N = 100) source sentences in TM_1 having the largest number of words in common with the given source sentence, S, to be translated. However, while the first of these candidate sentences has by construction more words in common with S than any other sentence in the TM,

[1] This is an instance of a certain "sophistication" of commercial TM-based systems, that we could easily emulate in DTM, by considering several S' instead of one.

it may not have these words in the same order as S. So, in a second pass, we perform a LCS (Longest Common Subsequence) procedure between S and each of the N candidates, and retain the S' for which the corresponding subsequence is the longest. The LCS procedure is the standard dynamic program for performing this task, but we modify it in order to allow control of the maximal size of gaps that are allowed to occur in the subsequence (which is taken to be 2 in the current experiments). Note that if N was taken to be the number of units in TM_1, the procedure would guarantee finding the best match according to the LCS criterion, but that taking N = 100 is only a reasonable heuristic approximation to that.

In the case of our example S, this procedure finds S': *'Install the lower arm front retaining bolt in two stages.'*, along with its translation T': *'Poser la vis de fixation avant de bras inférieur en deux passes.'* The procedure also identifies as the longest common subsequence between S and S' the sequence *'Install the $\diamondsuit$ arm front retaining bolt in $\diamondsuit$ stages .'*, where the symbol $\diamondsuit$ indicates a gap. This gapped word sequence will be used as the source phrase P_S of the dynamic bi-phrase (P_S, P_T) that we are going to build next.

In order to complete the process (that is, in order to compute P_T), we now need to consider the word alignment matrix associated with the pair (S', T').

3.5 Word-Alignment Matrix

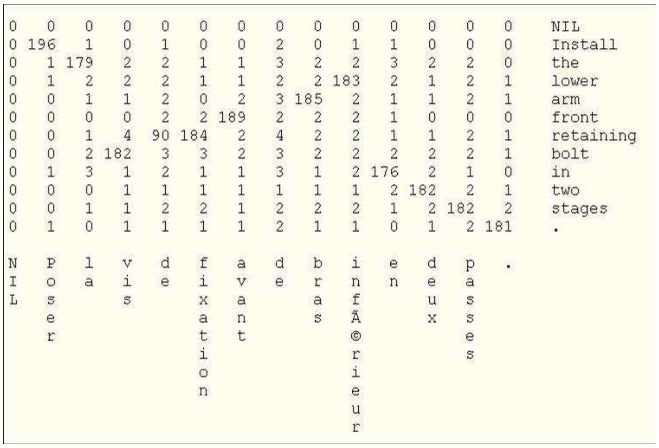
As we explained earlier, we started the whole process by training MATRAX on TM_1, considered as an aligned bilingual corpus.[2]

One of the steps in this training consists in using GIZA++ [7] for producing word alignments for all the bi-sentences (S', T') in TM_1. This is done in the following way:

– Use GIZA++ to produce the top 100 forward word alignments for (S', T'), as well as the top 100 backward word alignments for (S', T'); as is well known the forward (resp. backward) alignments are non-symmetrical, as they only allow 1-to-n alignment configurations,.
– Combine these alignments in a $|S| \times |T|$ matrix of counts; this joint word alignment matrix is now "symmetrical" relative to source and target, with each entry being an integer between 0 and 200.

The alignment matrices thus obtained as a by-product of MATRAX training are stored along with the TM_1 translation units. At translation time, when retrieving the fuzzy-match pair (S', T'), we then also retrieve the associated alignment matrix shown in Figure 2.

[2] One difference between a TM and a bilingual corpus is that the TM does not accurately represent the actual statistics of duplicate source sentences in the translations actually produced, and thus using the TM as a bilingual corpus might diverge somewhat from the corresponding statistics of the domain, but this does not appear to be a serious problem.



```
0    0    0    0    0    0    0    0    0    0    0    0    0    0     NIL
0  196    1    0    1    0    0    2    0    1    1    0    0    0     Install
0    1  179    2    2    1    1    3    2    2    3    2    2    0     the
0    1    2    2    2    1    1    2    2  183    2    1    2    1     lower
0    0    1    1    2    0    2    3  185    2    1    1    2    1     arm
0    0    0    0    2    2  189    2    2    2    1    0    0    0     front
0    0    1    4   90  184    2    4    2    2    1    1    2    1     retaining
0    0    2  182    3    3    2    3    2    2    2    2    2    1     bolt
0    1    3    1    2    1    1    3    1    2  176    2    1    0     in
0    0    0    1    1    1    1    1    1    1    2  182    2    1     two
0    0    1    1    2    2    1    2    2    2    1    2  182    2     stages
0    1    0    1    1    1    1    2    1    1    0    1    2  181     .

N    P    l    v    d    f    a    d    b    i    e    d    p    .
I    o    a    i    e    i    v    e    r    n    n    e    a
L    s         s         x    a         a    f         u    s
     e                   a    n         s    Ã         x    s
     r                   t    t              ©              e
                         i                   r              s
                         o                   i
                         n                   e
                                             u
                                             r
```

Fig. 2. Word alignment matrix for (S', T')

3.6 Bi-phrase Extraction

Our task is now the following. We are given a gapped source phrase P_S, along with a word alignment matrix for the pair (S', T'), and we need to extract from that matrix a gapped target phrase P_T "maximally" aligned with P_S. The algorithm we use to do that is related to the competitive linking method of Melamed [10], and works in the following way:

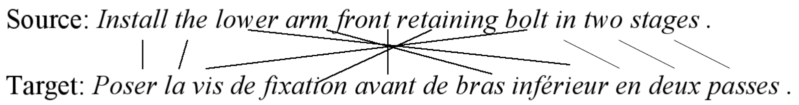
1. We initialize the set TargetWords to $\emptyset$;
2. We start by identifying the cell C with the maximal count in the matrix;
3. If C is associated with a source word belonging to P_S, then we include the corresponding target word occurrence in TargetWords, otherwise we don't;
4. We eliminate all cells belonging to the same line as C from further consideration, as well as all cells belonging to the same column as C;
5. We now identify the cell C' with maximal count among the cells still under consideration in the matrix, and iterate the process until all cells have been eliminated;
6. We finally collect all the word occurrences belonging to TargetWords in their sequential order to produce a gapped sequence P_T.

Performing this procedure on our example, we obtain the bi-phrase (P_S, P_T): (*'Install the ◇ arm front retaining bolt in ◇ stages .'*, *'Poser la vis de fixation avant de bras ◇ en ◇ passes .'*)

This "dynamic" bi-phrase is then added to the collection of bi-phrases retrieved by Matrax from its static bi-phrase library. In order to favor the use of the dynamic bi-phrase over that of other "standard" bi-phrases which may be in competition with it, we assign a strong (a priori defined) value to its associated feature ϕ_{phr}, which is an estimate of the conditional probability

$\Pr(P_T|P_S)$.[3] Standard decoding is then resumed with this extended collection of bi-phrases, producing the translation: *'Poser la vis de fixation avant de bras supérieur en trois passes.'*, which is seen by inspection to be composed from three bi-phrases: (P_S, P_T), plus the static bi-phrases (*'upper'*,*'supérieur'*) and (*'three'*,*'trois'*).

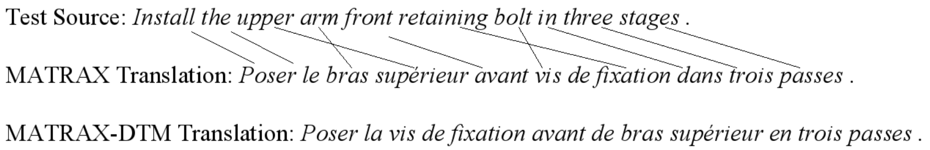
One interesting aspect of this example is the complexity of the reorderings at the level of word alignments in the translation unit (S', T'), as can be seen informally in Figure 3.


Source: *Install the lower arm front retaining bolt in two stages .*

Target: *Poser la vis de fixation avant de bras inférieur en deux passes .*

Fig. 3. A word aligned example bisentence

Phrase-based SMT systems such as MATRAX are usually reluctant to generate such complex reorderings, as typically one of the feature functions used strictly controls the amount of reordering between biphrases of the source and targets sentences (i.e. the reordering feature function used in MATRAX [4]). The result of this reluctance can be seen in Figure 4 where the corresponding translations of MATRAX and MATRAX-DTM for the source sentence is given. As it is apparent from the translations, MATRAX is only able to simulate local organization whereas MATRAX-DTM can exploit global organization. The amount of reorderings allowed in MATRAX is simply not enough in this case and it is likely that increasing it will degrade the quality of the translations as it will also open possibilities for a soup of words. In such cases MATRAX-DTM is likely to perform better than MATRAX, because it can rely on larger bi-phrases which encapsulate complex re-ordering constraints.


Test Source: *Install the upper arm front retaining bolt in three stages .*

MATRAX Translation: *Poser le bras supérieur avant vis de fixation dans trois passes .*

MATRAX-DTM Translation: *Poser la vis de fixation avant de bras supérieur en trois passes .*

Fig. 4. Corresponding translations for a given source sentence

[3] In the current implementation, the value of that feature is fixed arbitrarily, once for all for all dynamic bi-phrases, and the feature is not distinguished from the standard feature ϕ_{phr} used for static bi-phrases; A more principled approach would consist in having a specific $\phi_{phr'}$ feature for the dynamic bi-phrases, for which the associated log-linear weight would be trained independently from the weight of the standard feature.

4 Experimental Results

We present in Table 1 the results we have obtained for different fuzzy-match levels.

Table 1. Translation performance for different fuzzy-match levels

		Translation Systems		
Level of Fuzzies	**Score**	**MATRAX**	**MATRAX-DTM**	**TRADOS**
30 − 40	NIST	6.8033	6.9568	3.2830
	BLEU	0.1944	0.2118	0.0905
40 − 45	NIST	7.0222	7.2083	3.9443
	BLEU	0.2307	0.2527	0.1307
50 − 74	NIST	8.5695	9.4795	6.5578
	BLEU	0.3181	0.4015	0.2652
74 − 85	NIST	9.2462	10.9127	8.7225
	BLEU	0.3951	0.5504	0.4154
80 − 95	NIST	9.7141	12.3632	10.1338
	BLEU	0.4301	0.6597	0.5118
90 − 95	NIST	9.4954	12.2955	10.2812
	BLEU	0.4478	0.7140	0.5633
95 − 100	NIST	9.2833	12.7308	11.8645
	BLEU	0.4053	0.7106	0.6516
30 − 100	NIST	9.4000	11.8356	9.5668
	BLEU	0.3672	0.5572	0.4410

The results show that for all fuzzy-match ranges, the DTM system MATRAX-DTM performs markedly better than both the SMT system and the TM-based system. We plot the NIST [11] and BLEU [12] performances of the three systems for different fuzzy match levels respectively in the first and second panels of Figure 5.

In both of these panels, it is again visible that MATRAX-DTM performs much better than both MATRAX and TRADOS. We also observe that TRADOS' performance is inferior to MATRAX's on lower-order fuzzies but it catches up in roughly the range 50 − 80 and performs better for higher-order fuzzies. We also see that TRADOS is a little better than MATRAX for the overall range 30 − 100, in terms of both the NIST and the BLEU performance, and that the difference is larger for the BLUE scores; here the difference between BLEU and NIST may be due to the fact that MATRAX is optimized using the NIST scoring function, thus possibly advantaging NIST scores at test time.

In the third panel of Figure 5, we finally show a comparison of MATRAX-DTM with the other two systems, in terms of relative percentage gains in terms of BLEU and NIST. We see that with increasing fuzzy match levels, the difference between the performance of MATRAX-DTM and that of MATRAX goes up whereas it goes down relative to that of TRADOS.

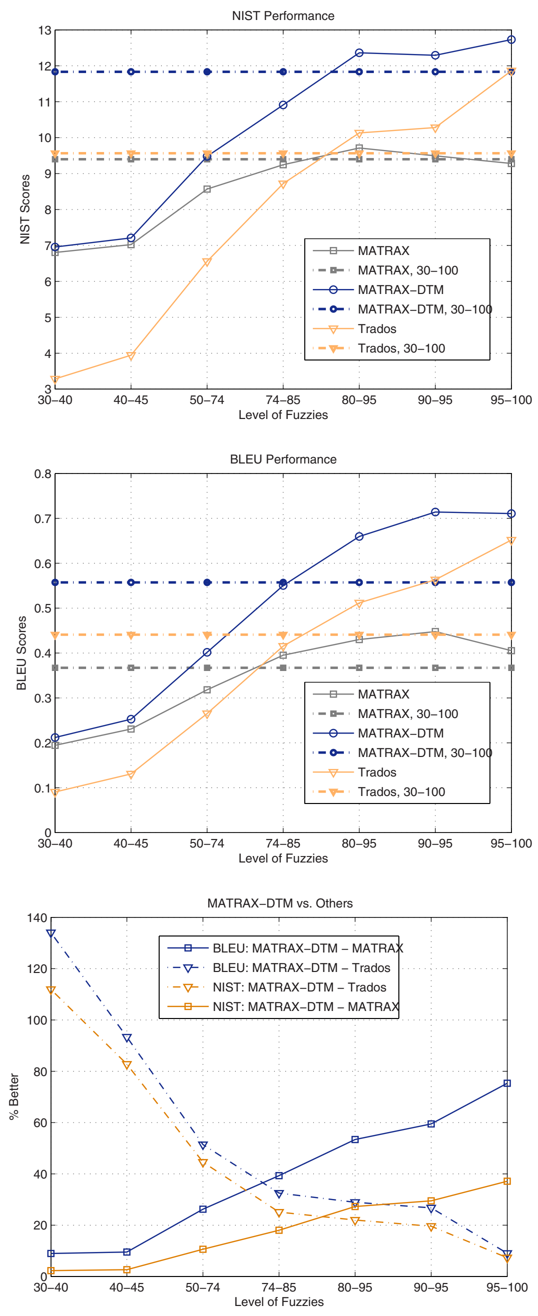


Fig. 5. The first two panels show NIST and BLEU performance of the three systems for different fuzzy match levels; the third panel compares the performance of MATRAX-DTM relative to each of the other two systems

5 Related Work

Several researchers have proposed ways to associate Translation Memories and Statistical Machine Translation systems. One of the early works in this area is [13], whose authors used an IBM-4 SMT model to *construct* a "Translation Memory" of source/target subsentential segments from a large bilingual corpus. Their adapted decoder then exploits these segments to produce better translations than available to the original word-based decoder. Today, this system would probably be recognized more as an early manifestation of phrase-based translation (the TM segments constructed in [13] would now be called "bi-phrases"!) than as an approach for improving the usability of standard Translation Memories. In a somewhat similar spirit, [14] also produce subsentential bi-segments, but this time using an Example-Based approach which attempts to detect useful source and target segments based on a syntactic chunker, and then using these bi-segments in a statistical decoder. The authors of a more recent article [15] experiment with different combinations of EBMT (Example-Based Machine Translation) and Phrase-Based SMT (PB-SMT) techniques, focussing on the potential gains of jointly using bi-segments obtained both by EBMT and by PB-SMT techniques, whether the translations are actually produced using an EBMT or a PB-SMT decoder. While by and large the papers just cited concentrate more on ways to extract reliable subsentential segments from a bilingual corpus, in [16], the authors are somewhat closer to the spirit of the present paper as they focus on improving the translation of close matches (one word edit-distance at most) between a test source sentence and a TM sentence, where the TM is a collection of short sentences in a "Basic Travel Expression Corpus". When such a close match is found, phrase-based techniques are used for computing the translation of the replacer-word as well as for identifying the subsegment of the TM target, corresponding with the source word to be replaced, which should be overwritten with that translation. When no such close match is found in the TM, then the test sentence is given to a standard PB-SMT system for translation. While the paper reports no significant improvement of the combined system relative to the PB-SMT system in terms of BLEU and NIST, it does report some improvements in terms of human evaluation.

6 Conclusions

We have presented the hybrid TM-SMT system DTM which is able to improve the translation of sentences having a TM fuzzy match by dynamically extracting from the fuzzy match a highly predictive bi-phrase which is added to the collection of static bi-phrases used by the underlying SMT system. The approach relies on the fact that the underlying SMT system handles non-contiguous phrases on a par with standard contiguous phrases, and so is able to accommodate the new dynamic bi-phrase, which is typically large and non-contiguous.

We describe experiments, performed on texts from a large industrial TM data base, that show that the DTM system considerably improves the BLEU and NIST scores compared to both the underlying SMT and TM-based systems, for all fuzzy-match ranges, with the SMT component of DTM being especially useful at low fuzzy-match levels, and its TM fuzzy-match component being especially useful at high fuzzy-match levels. We believe that these results suggest useful potential applications of this technology for professional translators in technical domains, for whom the use of Translation Memory tools is now commonplace.

Acknowledgments. Thanks to Brian Hudson and Felix Ustorf at GKLS for providing access to the translation memory used in the experiments and for their useful advice.

References

1. Zetzche, J.: Translation memory: state of the technology. Multilingual 18, 34–38 (2007)
2. Lagoudaki, E.: Translation memories survey 2006. Technical report, Imperial College London (2006)
3. Sikes, R.: Fuzzy matching in theory and practice. Multilingual 18, 39–44 (2007)
4. Simard, M., Cancedda, N., Cavestro, B., Dymetman, M., Gaussier, É., Goutte, C., Yamada, K., Langlais, P., Mauser, A.: Translating with non-contiguous phrases. In: HLT/EMNLP (2005)
5. Och, F.J., Ney, H.: The alignment template approach to statistical machine translation. Computational Linguistics 30, 417–449 (2004)
6. Lopez, A.: A survey of statistical machine translation. Technical report, University of Maryland technical report, UMIACS-TR-2006-47 (2007)
7. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics 29, 19–51 (2003)
8. Goutte, C., Yamada, K., Gaussier, E.: Aligning words using matrix factorisation. In: ACL 2004: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, pp. 502–509 (2004)
9. Trados: Trados translator's workbench (2007), `http://www.translationzone.com/en/products/sdltrados2007/`
10. Melamed, I.D.: Models of translational equivalence. Computational Linguistics 26, 221–249 (2000)
11. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: Proceedings of the second international conference on Human Language Technology Research, pp. 138–145. Morgan Kaufmann Publishers, San Francisco (2002)
12. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: ACL 2002: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics, Morristown, NJ, USA (2001)
13. Marcu, D.: Towards a unified approach to memory- and statistical-based machine translation. In: Proceedings of ACL 2001 (2001)
14. Langlais, P., Simard, M.: Merging example-based and statistical machine translation: an experiment. In: Richardson, S.D. (ed.) AMTA 2002. LNCS (LNAI), vol. 2499, Springer, Heidelberg (2002)
15. Groves, D., Way, A.: Hybrid data-driven models of machine translation. Machine Translation 19, 301–323 (2005)
16. Hewavitharana, S., Vogel, S., Waibel, A.: Augmenting a statistical translation system with a translation memory. In: Proceedings of EAMT 2005 (2005)

Identification of Transliterated Foreign Words in Hebrew Script

Yoav Goldberg and Michael Elhadad

Computer Science Department
Ben Gurion University of the Negev
P.O.B 653 Be'er Sheva 84105, Israel
{yoavg,elhadad}@cs.bgu.ac.il

Abstract. We present a loosely-supervised method for context-free identification of transliterated foreign names and borrowed words in Hebrew text. The method is purely statistical and does not require the use of any lexicons or linguistic analysis tool for the source languages (Hebrew, in our case). It also does not require any manually annotated data for training – we learn from noisy data acquired by over-generation. We report precision/recall results of 80/82 for a corpus of 4044 unique words, containing 368 foreign words.

1 Introduction

Increasingly, native speakers tend to use borrowed foreign terms and foreign names in written texts. In sample data, we found genres with as many as 5% of the word instances borrowed from foreign languages. Such borrowed words appear in a transliterated version. Transliteration is the process of writing the phonetic equivalent of a word of language A in the alphabet of language B. Borrowed words can be either foreign loan words with no equivalent in language B, or words from language A used as slang in language B. Identifying foreign words is not a problem in languages with very similar alphabets and sound systems, as the words just stay the same. But this is not the case in words borrowed from languages that have different writing and sound systems, such as English words in Japanese, Hebrew and Arabic texts.

Transliterated words require special treatment in NLP and IR systems. For example, in IR, query expansion requires special treatment for foreign words; when tagging text for parts of speech, foreign words appear as unknown words and the capability to identify them is critical for high-precision PoS tagging; in Machine Translation, back transliteration from the borrowed language to the source language requires the capability to perform the inverse operation of transliteration; in Named Entity Recognition and Information Extraction, the fact that a word is transliterated from a foreign language is an important feature to identify proper names.

We focus in this paper on the task of identifying whether a word is a transliteration from a foreign language – and not on the task of mapping back the transliteration to its source. In some languages, such as Japanese, identifying borrowed foreign words is easy – they are written in a special script (Katakana) used just for such purposes. Other languages don't have such luxury. In this paper we describe a method for identifying transliterated words in Hebrew, written in the Hebrew script. The method is unsupervised, uses easily acquired resources, and is not specific to Hebrew.

The Modern Hebrew writing system has properties that make it different from most Indo-european writing systems, and which make transliteration identification difficult (similar properties are shared with all Semitic languages):

- Vowels are not written in most cases, but in some cases, vowels are indicated by inserting specific letters (for the sounds *i* and *o*). The same letters can be used either as vowels or as consonants (*e.g.*, yod and vav).
- The sounds *p* and *f* are encoded by the same letter, similarly for the pairs *b* and *v* and *s* and *sh*. The sounds *th* and *j* do not exist in Hebrew.
- The letters *ayin, het, khaf* in Hebrew have no corresponding sounds in most European languages.
- In the written form, words are agglutinated with a sequence of prefixes and suffixes (corresponding to prepositions and pronouns).

In the rest of the paper, we first review previous work related to the task of transliteration identification. We then present our approach: we identify transliteration by comparing the letter structure of words with models trained in a way that captures the sound structure of the language – one in Hebrew and one in English, as written in the Hebrew writing system. The model for Transliterated English is trained on data automatically generated from an English corpora, using the CMU pronunciation dictionary and very simple phoneme to letter rules.

2 Related Work

There is a large body of work on transliteration [1, 2, 3] which mainly focuses on back-transliteration.

There is a growing body of research on automatic extraction of transliterated pairs [4, 5]. Sherif and Kondrak [5] use seed examples and a sentence aligned English/Arabic text to jointly learn a bilingual string distance function and extract transliterated pairs. While this work aims at complete alignment, our task is only the identification of transliterated candidates. Identification of transliteration candidates can help full alignment by relaxing the need for aligned text.

The task of identifying transliterated words has been less studied. Stalls and Knight [1] identified the problem – "...in Arabic, there are no obvious clues, and it's difficult to determine even whether to attempt a back-transliteration, to say nothing of computing and accurate one" – but don't deal with it directly.

Oh and Choi [6] studied identification of transliterated foreign words in Korean text, using an HMM on the word syllable structure. They used a corpus of about 1,900 documents in which each syllable was manually tagged as being either Korean or Foreign, and achieved impressive results. However, beside requiring a large amount of human labor, their results are not applicable to Hebrew (or Arabic) as these languages' syllables structure is not clearly marked in writing, and even the vowels are not available in most cases.

Nwesri *et al.* [7] dealt with the identification of transliterated foreign words in Arabic text in the setting of an information retrieval system. They tried several approaches: using an Arabic lexicon (everything which is not in the lexicon is considered foreign), relying on the pattern system of Arabic morphology, and two statistical ngram models, the better of which was based on Cavnar and Trenkle's rank order statistics [8], traditionally used for language identification. For the statistical methods, training was done on manually constructed lists of a few thousands Arabic and foreign words written in Arabic script. They also augmented each of the approaches with hand written heuristic rules.

They achieved mediocre results on their training set, somewhat lower results for their test set, and concluded that the best approach for identification of transliterated foreign words is the lexicon-based method enhanced by hand written heuristics, by which they achieved a precision of 68.4% and recall of 71.1% on their training set, and precision of 47.4% and recall of 57.2% on their test set.

Another related field of research is that of language identification, in which documents are classified according to their language. The problem of finding transliterated foreign words can be regarded as performing language identification on individual words instead of documents or sentences. Algorithms that rely on letter-ngram statistics can be relevant to the foreign words identification task. Two notable works are [8] and [9], both based on letter-level ngram statistics. Canvar and Trenkle use rank order statistics, and Dunning use Naive-Bayes classification with a trigram language model, and add-one smoothing.

Language identification can be considered a 'solved problem', with success rates of above 99.8%. However, such systems usually require a minimum of about 50 bytes of text in order to perform well. This data is not available when working on a single word. Indeed, Nwesri *et al.* [7] report low success rates using a modification of Cavnar's method for Arabic foreign words identification. Qu and Grefenstette [10] used Cavnar's method for the more limited task of language identification of names, to distinguish English, Japanese and Chinese names in Latin script. Training on 88k English names, 83k Japanese names and 11k Chinese names, they achieved accuracies of 92% (Japanese), 87% (Chinese) and 70% (English).

3 Our Approach

We concentrate on identifying borrowed and transliterated words in Hebrew text. As most borrowed words come from English or European sources, we concentrate on finding borrowed words from such origins. We also focus on a context-free approach, which works on words alone, without requiring their context. Our intuition is that Hebrew/Arabic words sound different than English/Indo-European words, and as letters are closely related to sounds, this should be reflected in their writing. Therefore, we believe that letter level ngram approaches should be applicable to this problem, given sufficient data and suitable language models, even at the word level.

Naive-Bayes Classification. At its core, our method is a Naive-Bayes classifier for choosing the best generative language model for a given word: Assuming two generative language models, M_n for generating native (Hebrew) words and M_f for generating foreign words, and an observed word w, we would like to find the model M_i that maximizes $P(M_i|w)$. As we don't know $P(M_i|w)$, we use Bayes Formula, and get:

$$P(M_i|w) = P(w|M_i)P(M_i)/P(w)$$

$P(w) = 1$ (the word is given), and we assume both models are equally probable ($P(M_n) = P(M_f)$), so we are left with the problem of evaluating $P(w|M_i)$. By normalizing the results:

$$P'(M_f|w) = \frac{P(M_f|w)}{P(M_f|w) + P(M_n|w)}$$

$$P'(M_n|w) = \frac{P(M_n|w)}{P(M_f|w) + P(M_n|w)}$$

we can get not only a decision, but a probability for each category. This is a standard construction (see [9, 11]). We differ from earlier work in the model parameters estimation, and by the data from which we learn.

Unsupervised Approach. Our model is unsupervised, in that it does not require any manually tagged data for training. Our approach needs data to learn and estimate the parameters $P(w|M_i)$. Unfortunately, there is no large corpus of transliterated foreign words in Hebrew script available, and manually creating one is labor intensive. For training their language model of Foreign words in Arabic script, [7] created a list of 3046 foreign words, and achieved rather low results. We identify two problems with this approach:

1. They learn from a set of words, without taking into account the frequencies of these words in the language, which results in a poor approximation of the language 'sound-patterns'.
2. The list of words constructed manually is just too small to be useful – most machine learning approaches will not get enough data to reach reliable results from such a quantity.

Instead of using a small amount of quality data, we opted for a large quantity of noisy data. For estimating the native language model we use a collection of Modern Hebrew texts from a period of about 100 years ago, which are assumed to have a relatively low frequency of foreign words. For estimating the foreign language model, we use over-generation, and create a large and noisy corpus. This noisy data performs much better than using just a small amount of hand-made clean data.

The Over-Generation Process. At the heart of the over-generation mechanism is a procedure *pron*− > *nat* which, given a foreign word in foreign script, outputs many possible transliterations of that word in the native script. These transliterations are meant for machines to learn from, and need not be 'correct' by human standards. Indeed, most of them will not be accepted as proper transliterations of the word by humans. They do, however, seem to capture the 'essence' of the foreign language in native writing.

The algorithm we use is as follows:

1. find a phonemic representation of the word
2. list all common ways of writing each phoneme in the native script
3. output all the combinations of writing produced in stage 2

For step 1, we look up the word in the CMU pronunciation dictionary (http://www.speech.cs.cmu.edu/cgi-bin/cmudict).[1] We ignore stress information, and distinguish phonemes at the start of words from the rest of the phonemes[2]. We also convert the 'T S' phoneme sequence into a new 'TS' phoneme, because this sequence can be represented in Hebrew by the single letter '**צ**'. We also change all 'K' phonemes into a new 'KH' phoneme if the word contains the 'ch' letter sequence and the pronunciation does not have a 'CH' phoneme.

Step 2 is achieved by a simple table lookup. The table is easy to construct, and presented below (Table 1).

As a concrete example, in Hebrew, the English word 'county' has two pronunciations in the cmudict: 'K AW1 N T IY0' and 'K AW1 N IY0'. These are converted to: '*K AW N T IY' and '*K AW N IY' ('*' marks a phone at the beginning of a word). The translation table indicates that:

'*K'	'ק'
'AW'	'או', 'אוו', 'ואו'
'N'	'נ'
'T'	'ת', 'ט'
'IY'	'י', 'אי', ϵ

and so the resulting 26 transliterations are (only 3 of them, underlined, would be accepted by humans):

קאונתי קאונתאי קאונת קאונטי קאונטאי קאונט קאוונתי קאוונתאי קאוונת קאוונטי
קאוונטאי קאוונט קואונתי קואונתאי קואונת קואונטי קואונטאי קואונט קאוני קאונאי
קאונ קאווני קאוונאי קאוונ קואוני קואונאי קואונ

[1] Our approach does not require any new and task specific lexicons, and crucially no lexicon is needed for the source (*e.g.*, Hebrew) language. Furthermore, for target languages with simpler letter-to-sounds rules, such as Spanish or Italian, a more direct, lexicon free approach could be taken.

[2] Because the Hebrew writing system does not include vowels, vocalized phonemes are transliterated differently when they are word-initial: word initial vowels are always represented in the written Hebrew form, while non word-initial vowels might be dropped and not appear in the written form.

Table 1. Pronunciation to Hebrew Translation Table

Phoneme	Possible Hebrew Rendering	Phoneme	Possible Hebrew Rendering
AA	או , א , ו	*AA	או , א
AH	א , ϵ	*AH	א
AW	או , אוו , ואו	*AW	או , אוו , אאו
AE	א , ϵ	*AE	א
AO	או , אוו , ו	*AO	או , אוו
AY	אי , יי	*AY	אי
B	ב	CH	צ'
D	ד	DH	ד , ת , ת'
EH	א , ϵ	*EH	א
ER	אר , ר	*ER	אר
EY	אי , יי , י	*EY	אי
F	פ	G	ג
HH	ה	IH	י , אי
*IH	אי	IY	י , אי , ϵ
*IY	אי	JH	ג'
K	ק	L	ל
M	מ	N	נ
NG	נג	OW	או , אוו , ו , וו
*OW	או , אוו	OY	אוי , וי
*OY	אוי	P	פ
R	ר	S	ס
SH	ש	T	ת , ט
TH	ת' , ד	UH	ו
*UH	או	UW	ו
*UW	או	V	ב , ו , וו
W	וו	Y	י
Z	ז	ZH	ז , ז'
TS	צ , טס	KH	ק , כ
N*	ן	M*	ם
H*	ך	F*	ף

We then apply this technique to a large body of foreign (English) text, to produce a large corpus for training (6MB of the Brown Corpus provided by the NLTK toolkit [12] resulted in 126MB of 'foreign Hebrew' data).

Probabilistic Model. Now that we have a fair amount of training data for both languages, we should decide on the specific probabilistic model. As a baseline, we used the estimation Dunning [9] used for language identification. The setting is a generative Markov Model, in which every letter is assumed to be dependent only on the n previous letters[3]. Dunning used a model of depth 2 (a trigram model), we tried models of depth 2 and 3.

[3] Word boundaries are also considered as letters, and we treat each of the Hebrew ז' צ' ג' as one letter as well (צ' , ז' and ג' are used in Modern-Hebrew script to represent the sounds tch, dj, and j, found in many foreign words. These sounds are not available in traditional Hebrew, and so don't have unique characters.)

In this model, the probability of a word w (made of letter $w_1 \dots w_N$) is:

$$\prod_{n \leq i \leq N} P(w_i | w_{i-1}, \dots, w_{i-n})$$

The most natural estimation for $P(w_i | w_{i-1}, \dots, w_{i-n})$ is:

$$P(w_i | w_{i-1}, \dots, w_{i-n}) = \frac{C(w_i w_{i-1} \cdots w_{i-n})}{C(w_{i-1} \cdots w_{i-n})}$$

Where $C(.)$ are the counts collected from the training data, and we set $C(.) = 0$ for every value that appears less than K times (we set $K = 5$), as these counts are considered to be unreliable, esp. when dealing with such a small alphabet. For smoothing, Dunning used Laplace's add-1 method, which seems reasonable for small alphabets (and performs very well in practice for language identification).

Table 3 lists the results of using trigram and 4-gram models with Laplace smoothing (Dun3 and Dun4), on the data described in Section 4. The trigram model performs better, achieving a precision of 58.7% and recall of 64.9%.

The results are fine (much better than was previously achieved for the Arabic case[4] [7] who reported precision/recall of 47%/57% on the test data using a lexicon), but still far from perfect. We suggest below several modifications to improve these results.

Non-traditional Smoothing. Laplace's add-one smoothing seems too simplistic for this task. We note that using a higher-order model yields somewhat worse results in terms of precision (58.7% for the trigram model vs. 55.6% for the 4gram model) while only slightly improving recall (from 64.9% to 65.7%). Using more advanced smoothing methods, such as these used by [13] didn't improve the results. An interesting observation is that our small alphabet together with our practically unlimited training set data means that smoothing will hardly be needed for unigram or bigram models. However, we would like to have the extra discrimination power we can get from 3- and 4-gram models, when available. To achieve this, we create 4 different models for each language ($M_{i1} \dots M_{i4}$, which are unigram, bigram, trigram and 4gram models) and linearly combine them:

$$P(w|M_i) = \lambda_1 P(w|M_{i1}) + \lambda_2 P(w|M_{i2}) + \lambda_3 P(w|M_{i3}) + \lambda_4 P(w|M_{i4})$$

If we keep $\sum_{i \in 1 \dots 4} \lambda_i = 1$, $P(w|M_i)$ stays a proper probability. One can view this combination as a weighted voting scheme.

Note that this differs from traditional back-off estimation based smoothing, which is also linear combination of lesser degree ngrams, but on the ngram level, whereas we work on the model level.

Setting the best values of λ_is is yet to be decided, for the current experiment we set them all to an equal value of $1/4$, which appears to work well empirically, as can be seen in Table 3 (Vot).

[4] The results are not directly comparable, but we believe the differences in performance are big enough to be indicative of the improved performance of our method.

Note that this scheme does leave a structural 0 in the model: for example the overgeneration procedure for Hebrew does not produce the letters ח or ע, as these sounds rarely occur in foreign words, if at all. As a result, even after our 'smoothing' words with these letters will have a 0 probability of being foreign, even in the unigram model. But this is fine, as it results in the right decision. In some respects, this is a good feature of this smoothing method.

Backward Model. When working with such a short words (the average word in our corpus is about 5.5 letters long), we cannot afford to miss clues that can be ignored by methods working on larger samples. In addition to the forward-moving Markov process, in which every letter is dependent on the n previous ones, we hypothesise a backward moving process in which every letter is dependent on the n following ones, $P(w_1, \ldots, w_N) = \prod p(w_i | w_{i+1}, \ldots, w_{i+n})$. These probabilities are in many ways similar to the forward moving ones, but there are slight variations in some cases. We add 3 such models (bigram, trigram, 4gram) to our voting scheme:

$$\begin{aligned} P(w|M_i) = \lambda_1 P(w|M_{i1}) + \lambda_2 P(w|M_{i2}) + \lambda_3 P(w|M_{i3}) + \lambda_4 P(w|M_{i4}) \\ + \lambda_5 P(w|M_{i2back}) + \lambda_6 P(w|M_{i3back}) + \lambda_7 P(w|M_{i4back}) \end{aligned} \quad (1)$$

The addition of the backward models to the vote (Vot+Back in Table 3) results in improved precision (from 76.3 to 80.4), and slightly hurt recall (from 71.7 to 71.4).

All Foreign Words Are Not Created Equal. Examining foreign words in Hebrew text reveals that a very big proportion of them are proper names. Although many English words function also as proper names, the sound patterns of proper names are somewhat different than those of regular English words. As they represent much of the data we want to identify, they deserve a special treatment. This special treatment is achieved by adding another 'language' to our identification scheme – the language of foreign proper names, and building a model M_{fn} for that language.

The foreign names corpus is brutally constructed by creating a sub-corpus of the English one, containing all words that appear only in capitalized form, and then over-generating the corresponding native version. This is noisy in several respects: some of the capitalized words in the Brown corpus are not really proper names but rather plain nouns or modifiers[5], and the representation of proper names in cmudict is not very complete. For this reason, the resulting model can not reliably decide between 'English' and 'English name', but it does succeed in approximating some of the properties of names which get lost in the mass of all English words.

We then apply our NaiveBayes classifier on the word, and get 1 of 3 possible decisions: $native$, $foreign$, $foreign\_name$. We treat either $foreign\_name$ or $foreign$ as $foreign$. This results in our best model so far, in terms of recall (82 vs. 71), even if somewhat hurting precision (80.1 vs. 80.4).

[5] Note that we consider only words for which **all** occurrences are capitalized, thus we rule out most of this kind of noise.

Adding a Lexicon. Having achieved reasonable precision and recall by using statistical information alone and no explicitly annotated data, we turn to assess the effect of adding knowledge from a manually created lexicon of Hebrew words. For our lexicon, we use the word-list of the Hebrew spell checker HSPELL [14]. This lexicon contains 404,640 unique words.

As a baseline (HSPELL in Table 4), we consider all the words appearing in the lexicon as Hebrew, and all the other words as Foreign. This results in our best recall so far (85%), coupled with very poor precision (13%). Next, we combine the lexicon with the statistical models by considering every word appearing in the lexicon as Hebrew, and using the statistical model for deciding the rest of the words. This is done for the voted forward and backward model (Vot+Back+HSPELL) as well as for the model considering proper-names (Vot+Back+Names+HSPELL).

The lexicon increases the precision of each model by about 11%, while dropping the recall by about 10%. We note that the effects of incorporating the proper-names model are orthogonal to the addition of lexicon knowledge. The decrease in recall is due to very common borrowed words (*e.g.*, “Blues”, “Internet”, “Single”), and some common foreign names (*e.g.*, “Madonna”, “Paul”, “Glenn”) which are included in the lexicon.

Comparison with a Supervised Method. In Section 3, we claim that our method of learning with a lot of noisy data is better than learning with a small amount of quality hand annotated data. To support this claim, we perform a 5-fold cross validation experiment, in which we use some of our hand annotated data to train the trigram, voted, and voted+backward models, and test on the rest of it.

The results are summarized in Table 2. As can be seen, the models trained on the automatically generated noisy data (Table 3) consistently outperform the supervised models trained on the small amount of data. Contrary to the case with the generated noisy data, there is no real difference between the voted and voted+back models for the small supervised data.

4 Evaluation

Experiment Settings. For training the Hebrew model, we use the prose and mass sections of the ‘Ben Yehuda Project’ (the Hebrew equivalent of ‘Project Gutenberg’, containing copyright-free modern-Hebrew writings (prose, poetry and essays) of 26 authors, all of which died more than 70 years ago), overall 28MB of text.

For the foreign models we use text over-generated from 6MB portion of the Brown Corpus distibuted with the NLTK toolkit [12]. For testing on the Hebrew case, we use words from 50 articles from the ‘gossip’ section of Ynet[6], a total of 9618 words, 4044 of them unique. We manually removed the prefixes (prepositions) from all words, and tagged them as either H(ebrew), F(oreign) or FN(foreign name). Overall, there are 3608 Hebrew words, 368 foreign words of

[6] http://www.ynet.co.il – a popular Israeli news portal.

which 251 are foreign proper names, and another 68 words which are ambiguous (these words either have a clear foreign origin but became so common that they sound Hebrew, or they have two readings, one Hebrew and one borrowed). The data is available at (http://www.cs.bgu.ac.il/∼yoavg/ForeignWordsId).

For the cross validation experiments described in Section 3, we took for each fold a different 20% of the Hebrew words and 20% of the Foreign words for testing, and the rest for training.

Results. The following tables summarize the results of all the experiments described above.

Table 2. Results on Ynet Gossip Section, Supervised via 5-fold Cross Validation

Experiment	Precision (%)	Recall (%)
Dun3(CV)	17.75	60.42
Vot(CV)	59.72	60.81
Vot+Back(CV)	59.70	60.76

Table 3. Results on Ynet Gossip Section, using Purely Statistical Models

Experiment	Precision (%)	Recall (%)
Dun3	58.7	64.9
Dun4	55.6	65.7
Vot	76.3	71.7
Vot+Back	80.4	71.4
Vot+Back+Names	80.1	82

Table 4. Results on Ynet Gossip Section, using Statistical Models together with a Lexicon

Experiment	Precision (%)	Recall (%)
HSPELL	13	85
Vot+Back+HSPELL	91.86	61.41
Vot+Back+Names+HSPELL	91.19	70.38

Precision is the ratio between the number of correctly identified foreign words and the total number of words identified as foreign. Recall is the ratio between the number of identified foreign words, and the real number of foreign words. The 68 ambiguous words were excluded from the calculations.

5 Open Issues and Future Work

Segmentation. In both Hebrew and Arabic, some prepositions are appended as prefixes to words. All the results presented here ignored that fact and were on words without prefixes. This is suitable for IR settings, but not good enough for the general NLP application, in which text appear unsegmented. This prefixation can make otherwise 'foreign' words to get tagged as 'Hebrew'. Although recent morphological disambiguators for Hebrew [15, 16] perform such segmentation with reasonable accuracy (above 98%), they all rely on a Hebrew lexicon. By its very nature, such lexicon is bound not to be complete in its coverage of transliterated foreign words, and so the above-mentioned disambiguators unfortunately fall on the wrong side of the pipeline – it is precisely for improving such systems that foreign words identification is needed. Dealing with unsegmented data is an interesting problem, left for future work.

Context. Our method is context free, and assigns probabilities to words. But context does matter in some cases (for example, some words can be either foreign or native, depending on the reading). Context can also help in the segmentation problem. In addition, integrating our method with the output of a POS tagger is also a promising direction.

Arabic. The Arabic language is similar to Hebrew in terms of the sound patterns and writing system. Therefore, we expect that given the proper training material, our method will perform well for Arabic as well.

Different Kinds of Borrowings. This work focuses on identification of foreign words, yet no real distinction was made between cognates, borrowings and "real" transliterations. For most parts we can indeed ignore this distinction: "heavily incorporated" cognates and borrowing, which we consider as native words, tend to adopt the Hebrew pronunciation (and writing), and are considered Hebrew also by our algorithm, while less incorporated cases of borrowing exhibit inconsistent writing patterns, and demand special treatment similar to that of "pure" transliterations. However, finding an algorithmic way of separating these groups is an interesting research question.

6 Conclusions

We presented a method for identification of borrowed words and transliterated names in Hebrew text. The method is very loosely supervised, does not require annotated data, and achieves satisfactory results (precision/recall of 80%/82% with a purely statistical method, and 91%/70% when combined with a Hebrew lexicon). We verified that our approach of learning from a large amount of automatically generated data greatly outperforms learning with a small amount of manually annotated data. Giving specific care to identification of proper nouns was shown to improve performance.

References

[1] Stalls, B., Night, K.: Translating Names and Technical Terms in Arabic Text. In: Proc. of the COLING/ACL Workshop on Comp. Approaches to Semitic Languages (1998)
[2] Al-Onaizan, Y., Knight, K.: Machine Transliteration of Names in Arabic Text. In: Proc. of ACL Workshop on Comp. Approaches to Semitic Languages (2002)

[3] Yoon, S.Y., Kim, K.Y., Sproat, R.: Multilingual transliteration using feature based phonetic method. In: Proc. of ACL (2007)
[4] Klementiev, A., Roth, D.: Named entity transliteration and discovery from multilingual comparable corpora. In: Proc. of NAACL (2006)
[5] Sherif, T., Kondrak, G.: Bootstrapping a stochastic transducer for arabic-english transliteration extraction. In: Proc. of ACL (2007)
[6] Oh, J., Choi, K.: A statistical model for automatic extraction of korean transliterated foreign words. Int. J. of Computer Proc. of Oriental Languages 16 (2003)
[7] Nwesri, A.F., Tahaghoghi, S., Scholer, F.: Capturing out-of-vocabulary words in arabic text. In: Proc. of EMNL 2006 (2006)
[8] Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. In: Proc. of SDAIR-1994 (1994)
[9] Dunning, T.: Statistical identification of language. Technical report, Computing Research Lab, New Mexico State University (1994)
[10] Qu, Y., Grefenstette, G.: Finding ideographic representations of japanese names written in latin script via language identification and corpus validation. In: Proc. of ACL (2004)
[11] Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
[12] Bird, S., Loper, E.: NLTK: The natural language toolkit. In: The Comp. Vol. to the Proc. of ACL (2004)
[13] Thede, S.M., Harper, M.P.: A second-order hidden markov model for part-of-speech tagging. In: Proc. of ACL (1999)
[14] Har'el, N., Kenigsberg, D.: HSpell - the free Hebrew spell checker and morphological analyzer. Israeli Sem. on Comp. Ling (2004)
[15] Adler, M., Elhadad, M.: An unsupervised morpheme-based hmm for hebrew morphological disambiguation. In: Proc. of the ACL (2006)
[16] Shacham, D., Wintner, S.: Morphological disambiguation of Hebrew: A case study in classifier combination. In: Proc. of EMNLP-CoNLL (2007)

Innovative Approach for Engineering NLG Systems: The Content Determination Case Study

Marco Fonseca, Leonardo Junior, Alexandre Melo, and Hendrik Macedo

Departamento de Computação, Universidade Federal de Sergipe, 49100-000,
São Cristóvão/SE, Brazil
marcos_ufs@yahoo.com.br, leonardobsjr@yahoo.com.br,
asmelo10@hotmail.com, hendrik@ufs.br

Abstract. The purpose of Natural Language Generation (NLG) systems is that of automating the production of linguistically correct texts from a data source. Generators are usually built using ad-hoc software engineering practices, lacking a well-defined development process, standard software architecture, and the use of worldwide programming languages. This paper describes a new development approach that leverages the most recent programming languages and standards of modern software engineering to enhance the practical use of NLG applications. To show the practicability of the proposal, a content determination system, which accepts as input wrapped Web data regarding soccer championship results, was developed.

1 Introduction

Natural Language Generation (NLG) [13] is a conceptually consolidated technology. Past research has clarified many fundamentals issues and conceived solutions that are robust and scalable enough for practical use. Furthermore, opportunities for practical applications have multiplied with the information inundation from relevant Web content sources.

Unfortunately, NLG techniques remain virtually unknown and unused by mainstream and professional computing. This situation is probably due mainly to the fact that until recently, NLG was built using *ad-hoc* software engineering practices with no explicit development process and no standard software architecture. Reliance on special-purpose esoteric modeling and implementation languages and tools is another NLG issue. Every system is designed and implemented following specific domain complexities and needs and little has been done to change the portrayed situation. A good example is surface realization activity. Many realization components have been built based on different grammatical formalisms and theories used to describe NLG [8].

This work proposes an innovative approach to the development of NLG systems, in which the pipeline of text generation tasks work as a set of consecutive rule base for model transformation. Such methodology for building applications by applying transformations on models in different levels of abstraction was recently popularized as a new software engineering paradigm, the Model Driven Architecture (MDA) [12].

This paper shows how to adapt MDA to knowledge-driven software such as NLG systems. In particular, we show how the task of content determination could be achieved by means of MDA.

A wrapper was built to extract real soccer-related data from Web given an URL and turn them into instances of the soccer domain model previously specified. Both domain model and instances are used by the content determination component to generate the appropriate set of messages in the NLG sense.

The rest of the paper is organized as follows. Section 2 makes a brief description of common NLG tasks. Section 3 presents the proposed approach with emphasis on the content determination task. The application developed is described in section 4 and some related work is presented in section 5. Finally, in section 6, we present some concluding remarks.

2 Text Generation in Natural Language

NLG systems produce texts in natural language from rough data that represent information in a specific domain, and that are organized in conventional databases, knowledge bases, or even being produced as result of some application processing.

The natural language generation process is traditionally seen as a goal-driven communication process. As a consequence, the final text, being written or spoken, just a single-clause or a multi-paragraph document, is always an attempt to address some communicative goal. Starting from a communicative goal, the generator decides which information from the original data source should be conveyed in the generated text. During the generation process, the communicative goal is refined in more specific sub-goals and some kind of planning takes place to progressively convert them together with the original data to a well-formed and linguistically correct final text.

The whole generation process is traditionally organized in three specific tasks (layers): (1) content determination, (2) content organization, and (3) surface realization.

Content determination is the task of deciding which chunks of content, collected from the input data source, will make up the final text. Each chunk of content represents an indivisible information unit. These content units are usually grouped in a semantic unit of higher complexity for a given application domain. A semantic unit is called *message*. Considering for instance a system that generates soccer reports, the sentences “Brazilian soccer team has beat the Argentines last Sunday” and “Sunday soccer report: Victory of Brazil over Argentine” represent different linguistic constructions for the same kind of message: “Victory”.

Content organization groups the generated messages appropriately as units for each level of linguistic hierarchy: the paragraph, the sentence and the phrase. In addition, it defines element ordering within a group for each respective level. Finally, it is in charge of specifying coordination and subordination dependencies between these groupings.

Surface realization is the task of choosing the appropriated term (word) and the syntactic construction for each content unit. This choice is constrained by lexical and grammatical rules of the language. Punctuation symbols are defined at this stage as well.

3 The Model Transformation Approach for NLG

In this section, we present an innovative approach to the modeling and implementation of NLG systems that considers programming languages and standard tools for software development. The approach uses the concept of model transformation, popularized with the advent of a recent software engineering paradigm and software architecture at the same time, called Model Driven Architecture.

The key principles of the proposal are: (1) consider the input and output for each NLG task as data model instantiations previously specified in some modeling language, (2) characterize the data model types for each transformation layer as domain data models, domain message models, data organization structure models, and data formatting specific models, and (3) automate the software development process steps by applying generic transformation rules between these model types.

The languages and modeling tools that are at the core of MDA, such as UML, OCL, MOF, XMI and QVT [9], are conceptually very close to the specialized languages traditionally used in NLG, with the great advantage of being standards in mainstream software development, highly supported and platform neutral. OCL is a semi-formal textual language to annotate UML diagrams with constraints among various model elements. It provides an intuitive object-oriented syntax for set theory, predicate logic and algorithmic constructs. It is central to all new usages of UML. MOF is a slight variant of the UML class diagram designed for defining data and language meta-models in an MDA context. A MOF meta-model can be made very precise through OCL annotation in the same way than an UML model. XMI provides a standard way to serialize an UML model or MOF meta-model as an XML document by providing an XML tag for each UML modeling construct. QVT (query-View-Transformations) is an OMG initiative to standardize the model transformation language.

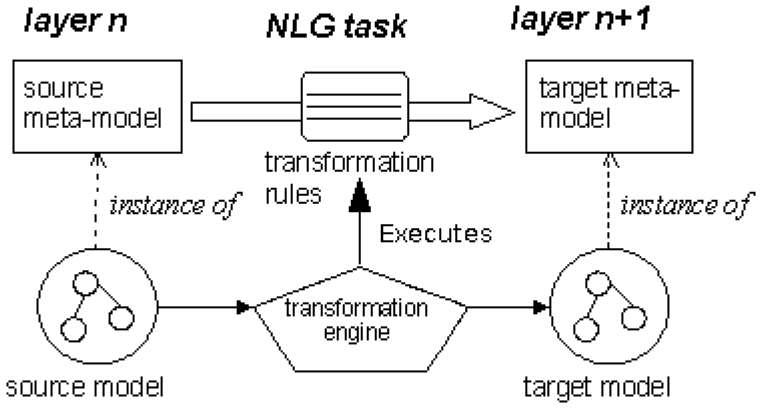


Fig. 1. Model transformation between subsequent layers of a generation pipeline

The definition of generic transformations between the different kinds of models is central to MDA. A transformation definition is a set of rules, written in a transformation definition language, which defines mappings between elements of a source meta-model and a target meta-model, both specified in MOF. It usually reuses OCL as sub-language for identifying the elements of these meta-models that are related or mapped by these transformations. Given as input a model instance of the source meta-model, a transformation engine applies the defined rules and outputs an instance of the target meta-model.

In our proposal, data models and data meta-models are specified by UML/MOF artifacts and the transformation rules between subsequent data meta-models in the generation pipeline are specified in a QVT-like language. This proposal is represented at figure one. This proposal is represented at figure one.

3.1 Content Determination

Content determination task consists in a collection of rules which should be applied to input data in order to produce a set of appropriated messages. These rules should model system's domain expert knowledge, specifying conditions and circumstances under which specific set of messages should be taken into account or not. The communicative goal is one of such circumstances. The type of messages produced with a communicative goal of reporting the winning teams of last championship round are completely different from those that should be produced by the communicative goal of reporting the variation in the championship classification table, for instance.

The rules may be specified using QVT. In that case, the source meta-model is the application domain data model (ADDM) whereas the target meta-model consists of the message data model (MDM) of the application.

3.2 Domain Modeling

The approach uses UML/MOF to specify the ontology of the generation system. Recent work considers the usage of UML as modeling language for ontologies [4]. Actually, UML models have characteristics that are often associated to the paradigm of declarative knowledge representation, such as the abstract nature of the modeling language that is not tied to any specific type of application, and the easiness with which an UML model can be modified without affecting other model's characteristics.

3.3 Message Definition

Generating texts regarding a specific domain requires the information to be expressed as an particular organization of elements that belongs to an ontology. We call it “organization by message.” The “Victory” message type illustrated previously, could be modeled as a relation between two entities of the class “team” and an entity of the class “game” with some properties that represent the number of goals for each team, for example. The system can thus express a clause indicating the winner of the match as a natural place.

The definition of the type of messages that should compose the second meta-model is not a trivial process. In the proposed approach the type of messages is defined based on a deep analysis of a corpus of natural texts. The methodology process for message definition consists of: (1) identifying individual sentences in each text present in the corpus, (2) identifying, for each sentence, the constituent phrases that will correspond to individual messages, (3) grouping identified messages as message classes and, finally, (4) modeling these classes as MOF meta-models.

Once the two set of MOF meta-models are specified - the ADDM and the MDM -, the transformation rules in a QVT-like language that provides the mapping between them, and the system input data as an instantiation of ADDM, the transformation engine is in charge of execute the rules and generate as processing output a coherent instantiation of MDM.

4 Generation of Soccer Reports

To show the feasibility of the proposal we have implemented a system that extract data concerning Brazilian soccer championship results from websites and generate the appropriated set of messages, which correspond to the first task of a NLG classical pipeline.

The application domain consists of twenty soccer teams that face one another twice. The winner of a match gets three points and the looser gets no point; in case of draw, both teams get one point. The team with the greater number of points after 38 rounds is the champion. Figure 2 illustrates a possible generated report.

"Hoje ocorreram 4 jogos da 15ª Rodada do 2º Turno do Campeonato Brasileiro. Às 16:00h aconteceram os confrontos entre São Paulo(3) e (2)Ponte Preta, Vasco(0) e (0)Figueirense, São Caetano(2) e (1)Paraná, e às 18:00h o confronto entre Grêmio(1) e (3)Fortaleza.

Com esses resultados o Santos continua líder com 56 pontos. Subiram na tabela: Vasco(+1->10º), São Paulo(+2->3º), São Caetano(+1->11º) e Fortaleza(+1->8º). Desceram na tabela: Paraná(-1->13º), Grêmio(-2->12º). Permanecem na mesma situação: Figueirense(7º) e Ponte Preta(5º)."

Fig. 2. Example of a soccer report in natural text (Portuguese)

4.1 Models and Meta-models

The ADDM is composed of ten conceptual entities. These concepts were modeled as UML/MOF artifacts as illustrated in the UML class diagram of figure 3.

In order to appropriately compose the MDM, we provided a corpora of natural language soccer reports. Each corpus concerns a distinct communicative goal. The text shown in figure 2, for instance, is the result of a generation process which addresses two different communicative goals: (1) reporting the results of the last championship round (first paragraph), (2) reporting the classification changes after the last championship round (second paragraph).

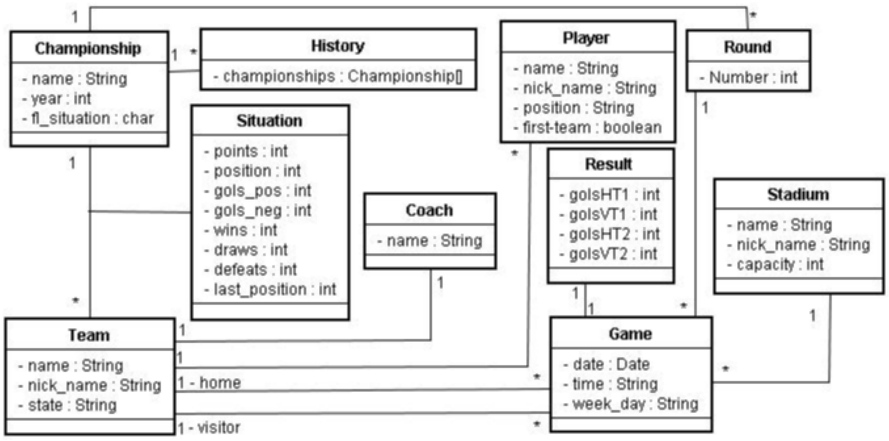


Fig. 3. UML/MOF domain concepts

From the analysis of the corpus, sentences and phrases, and to attend these two communicative goals in particular, the following message classes were defined:

(1) MsgGameResult: to inform the result of a match;
(2) MsgGameTime: to inform the time of a match;
(3) MsgNumberGames: to inform the number of matches in a specific day;
(4) MsgPoint: to inform the score of a team;
(5) MsgPositionChange: to inform the variation on classification table;
(6) MsgRound: to inform the current round;
(7) MsgTeamPosition: to inform the team classification in the championship.

Fig. 4. MsgPositionChange message meta-model

Figure 4 shows the UML/MOF MDM for MsgPositionChange. The UML artifacts in white color represent concepts belonging to ADDM and the OCL constraints formally define the relationship between the concepts of both layers.

4.2 Model Transformations

The transformation rules that map ADDM to MDM were implemented using a QVT-like language called ATL (ATLAS Transformation Language) [2]. Actually, ATL is both a transformation specification language and a transformation implementation language. The ATL rule is declarative and consists in a set of syntactically typed variables and OCL constraints working as filters that access the ADDM instances and transform them into MDM instances according to the MDM specification.

The ATL rules defined map ADDM attributes into MDM attributes. For each message class, a set of transformation rules was specified. Figure 5 illustrates the ATL transformation rule to generate MsgNumberGames messages.

```
helper def:countGames(dia:Integer,mes:Integer,year:Integer):
   Integer=Domain!Game.allInstances()->asSequence()
        ->iterate(game; sum:Integer=0|
            if (game.day=dia and game.month=mes and game.year=year)
                  then sum+1
                  else sum endif );
rule MsgNumberGames {
  from a : Domain!Game
  to q : Message!MsgNumberGames
   (nGames<-thisModule.countGames(6,2,2007))
}
```

Fig. 5. ATL transformation rule

4.3 Implementation

4.3.1 Wrapper

The input to the soccer report generator is a set of ADDM instances in XMI format. A wrapper was implemented to extract Brazilian soccer championship data from Web and instantiate the ADDM classes accordingly. The input to the wrapper system is an ordinary URL, from which it is in charge of navigating through hyperlinks related to the application domain till find relevant data, like a championship classification table, for instance. The wrapper reasoning component was developed using production rules implemented with JEOPS, a first-order forward-chaining Java-based inference engine. Details on the wrapper system is subject of a further paper.

Figure 6 illustrates the set of ADDM XMI instances as the result of a extraction process.

```
<Championship xmi:id="Campeonato2007" name="Campeonato Brasileiro
2007" year="2005" fl_situation="r" rounds="Rodada10"  teams=" Vasco
Flamengo"/>
<Round xmi:id="Rodada10" number="01" games = "Vasco_X_Flamengo"/>
<Game xmi:id="Vasco_X_Flamengo" day="17" month="02" year="2007"
time="16:00" week_day="Domingo" teamH="Vasco" teamV="Flamengo"
stadium="Maracana" round="Rodada10" result="ResultadoVasFla"/>
<Result xmi:id="ResultadoVasFla" golsHT1="1" golsVT1="1" golsHT2="2"
golsVT2="0"/>
<Stadium xmi:id="Maracana" name="Jornalista Mario Filho"
nick_name="Maracana" capacity="90000"/>
<Team xmi:id="Flamengo" name="Flamengo" nick_name="Mengo" state="Rio
de Janeiro" championship="Campeonato2007"
situation="Situacao_Mengo"/>
<Team xmi:id="Vasco" name="Vasco" nick_name="Vascao" state="Rio de
Janeiro" championship="Campeonato2007" coach="RenatoGaucho"
players="Romario" />
<Coach xmi:id="RenatoGaucho" name="Renato Portaluppi"/>
<Situation xmi:id="SituacaoVasco" points="6" position="1"
gols_pos="7" gols_neg="2" wins="2" draws="0" defeats="0"
last_position="2"/>
<Player xmi:id="Romario" name="Romario de Souza Faria"
nick_name="Romario" position="Atacante" first_team="true"/>
```

Fig. 6. ADDM XMI instances as the result of wrapper execution

Figure 7 shows the corresponding visual representation of these instances as an UML object diagram.

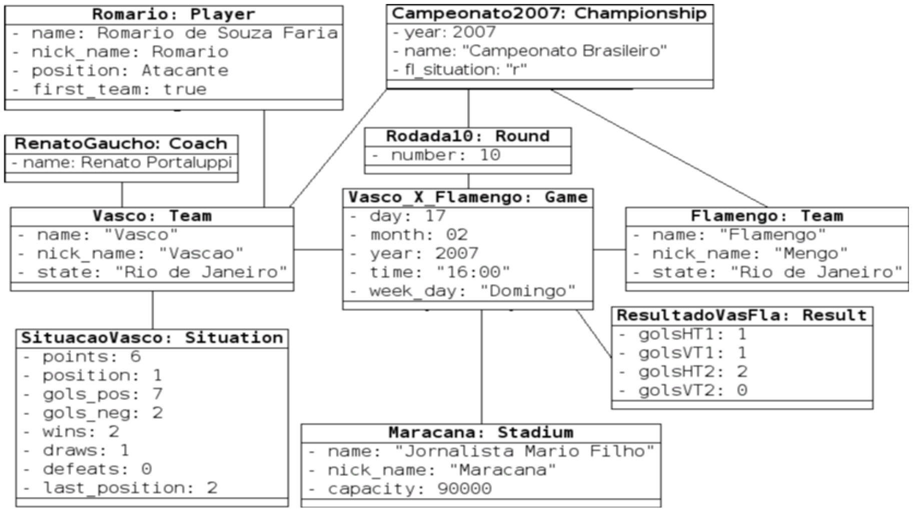


Fig. 7. ADDM instances as UML object diagram

4.3.2 Execution Environment

The development and execution environment consists of two Eclipse plug-ins: ADT and EMF. ADT allows for execution of ATL transformations. EMF is responsible for the codification of models and meta-models in XMI format. ADT also provides a simpler textual notation to the specification of EMF meta-models, the KM3, and automatically generates XMI for a meta-model defined in KM3.

As described so far, ADDM instances are provided by the wrapper already in XMI format. The ATL transformation engine (inference engine) applies the transformation rules specified between ADDM and MDM, generating the MDM instances as results in XMI format (figure 8).

```
<MsgPoint team="Flamengo" points="6"/>
<MsgTeamPosition team="Flamengo" position="1"/>
<MsgGameResult teamH="Flamengo" teamV="Vasco" scoreH="1" scoreV="3"/>
<MsgGameTime time="10:30" teamH="Flamengo" teamV="Vasco"/>
<MsgPositionChange team="Flamengo" change="1"/>
```

Fig. 8. MDM XMI instances as the result of content determination execution

5 Related Work

Recently, NLG researchers began a standardization attempt of natural language generation architectures. One of the most thorough attempts to date is the Reference Architecture for Generation, project RAGS [3]. In this project, the focus was placed on the set of applied natural language generation systems, defined by particular criteria set by the RAGS project itself. The RAGS model also proposes a collection of tasks within a generation system, although these tasks draw not on a specification of the linguistic problems faced when creating text, but rather on modules that have commonly been assumed in applied NLG systems - in particular modules arising from Reiter's proposals for a consensus pipeline architecture for NLG systems. This is nevertheless useful, since it can be also used to classify a significant proportion of the NLG literature and there have been recent attempts to relate a number of generation systems to the RAGS-model. These early attempts to apply the RAGS architecture have shown that the functional tasks identified in RAGS are often distributed broadly over different components in any particular generation system. This can be taken as a sign that either these tasks really are so distributed or that the definitions of the tasks have less theoretical/practical integrity than hoped. The truth lies probably between these extremes.

Another issue is about the development process. In fact, NLG applications are usually engineered in an ad-hoc manner. Classical practices of NLG software lack a precisely defined development methodology and process. They are at odd with the domain-independent practices, languages and standards of modern software engineering that drive technological advances and training of mainstream computing.

An initial step towards the standardization of language and development process for NLG has been taken: making NLG by means of XML tree transformations is subject of recent academic investigation. Wilcock [15] discusses a number of ways in which XML can be used in natural language generation, in particular, for spoken dialogue systems. XtraGen [14] is a Java-based software system for real-time generation of natural language. It adopts a particular grammar formalism based on extended XML-based templates that is both influenced by the ideas found in XSL and in other Lisp-based systems. Its application interface allows the setting of the grammar, the XML input document and some control parameters, that must be compiled in advance. Barrutieta *et al* [1] discuss how RST can be expressed through XML annotations and then used to generate personalized documents in a learning environment for the web. In such work RST is simplified in the sense that the granularity of discourse segments does not transcend the boundaries of the sentence. A DTD is used to declare the names and properties of all the relations that will make up the logical structure of the document (discourse).

6 Conclusion

In this paper we have shown an innovative approach to the development of Natural Language Generation systems. We argue that the most advanced mainstream open standards in software architecture, modeling languages and processing tools such as MDA, UML, OCL, and XMI are conceptually very close to the special purpose ones used in NLG software and that they could and should be used to develop NLG software. We argue that a generic approach to NLG application development that considers domain–independence, languages and standards of modern software engineering, would enhance practical use of NLG technology. The feasibility of this proposal was illustrated with the implementation of a soccer report generator. A wrapper system was implemented to extract real soccer related data from websites and instantiate the application domain data meta-model (ADDM). The content determination component of the generator takes the ADDM instances as input and produce message data meta-model instances (MDM) as output. The results of experiments have shown that the prototype developed in prepared execution environment works satisfactorily for the task of content determination. Currently, the prototype is being extended to cover other tasks of the NLG pipeline.

References

1. Barrutieta, G., Abaitua, J., Díaz, J.: An XML/RST-based Approach to Multilingual Document Generation for the Web. Procesamiento del Lenguaje Natural 29, 247–253 (2002)
2. Bézivin, J., Dupé, G., Jouault, F., Rougui, J.: First experiments with the ATL model transformation language: Transforming XSLT into XQuery. In: Proceedings of the OOPSLA 2003 Workshop on Generative Techniques in the Context of the MDA (2003)
3. Cahill, L., Doran, C., Evans, R., Kibble, R., Mellish, C., Paiva, D., Reape, M., Scott, D., Tipper, N.: Enabling Resource Sharing in Language Generation: an Abstract Reference Architecture. In: Proceedings of the 2nd International Conference on Language Resources and Evaluation, Athens, Greece (2000)
4. Cranefield, S., Purvis, M.: UML as an Ontology Modelling Language. In: Proceedings of the Workshop on Intelligent Information Integration. 16th International Joint Conference on Artificial Intelligence (IJCAI 1999) (1999)
5. Dale, R., Moisi, H., Somers, H. (eds.): Handbook of Natural Language Processing. Marcel Dekker, New York (2000)
6. Fensel, D.: Ontologies: the Silver Bullet for Knowledge Management and Electronic Commerce. Springer, Heidelberg (2003)
7. Elhadad, M.: Types in Functional Unification Grammars. In: Proceedings of the 28th. Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 157–164 (1990)
8. Elhadad, M.: Using argumentation to control lexical choice: a unification-based implementation. PhD thesis, Computer Science Department, Columbia University (1992)
9. Eriksson, H.E., Penker, M., Lyons, B., Fado, D.: UML 2 Toolkit. John Wiley & Sons, Chichester (2003)
10. Favero, E., Robin, J.: HYSSOP: Natural Language Generation Meets Knowledge Discovery in Databases. In: IIWAS 2001, Linz, Austria (2001)
11. Mittendorfer, M., Niklfeld, G., Winiwarter, W.: Evaluation of Intelligent Component Technologies for VoiceXML Applications. Technical Report. Software Competence Center Hagenberg (SCCH) and The Telecommunications Research Center Vienna (FTW) (2001)
12. Object Management Group. Model Driven Architecture (MDA). OMG Document ormsc/2001-07-01 edition (2001)
13. Reiter, E., Dale, R.: Building applied natural language generation systems. Natural Language Engineering 3, 57–87 (1997)
14. Stenzhorn, H.: XtraGen: A Natural Language Generation System Using XML- and Java-Technologies. In: Proceedings of the 2nd Workshop on NLP and XML (NLPXML) (2002)
15. Wilcock, G.: Pipelines, Templates and Transformations: XML for Natural Language Generation. In: Proceedings of the first NLP and XML Workshop; Workshop session of the 6th Natural Language Processing Pacific Rim Symposium, Tokyo (2001)

Comparison of Different Modeling Units for Language Model Adaptation for Inflected Languages

Tanel Alumäe

Institute of Cybernetics at Tallinn University of Technology,
Akadeemia tee 21, Tallinn, 12618, Estonia
tanel.alumae@phon.ioc.ee

Abstract. This paper presents a language model adaptation framework for highly inflected languages that use sub-word units as basic units in a language model for large vocabulary speech recognition. The proposed adaptation method uses latent semantic analysis based information retrieval to find documents similar to a tiny adaptation corpus. The approach enables to use different language units for modeling document similarity. The method is tested on an Estonian broadcast news transcription task. We compare words, lemmas and morphemes as basic units for similarity modeling. We observe a drop in speech recognition error rate after building adapted language model for each news story. Morpheme-based adaptation is found to give significantly larger improvement than word and lemma-based adaptation.

1 Introduction

Language model adaptation is a task of building a language model (LM) for speech recognition that is better suited for the given domain than a general background model, given a small adaptation corpus. In recent years, *latent semantic analysis* (LSA) has been successfully used for integrating long-term semantic dependencies into statistical language models [1]. The LSA-based approach gradually adapts the background language model based on the recognized words by boosting the unigram probabilities of semantically related words, using co-occurrence analysis of words and documents.

However, this approach cannot be efficiently directly used for highly inflective and/or agglutinative languages, such as Estonian, Finnish, Turkish, Korean and many others. In such languages, each word-phrase can occur in a large number of inflected forms, depending on its syntactic and semantic role in the sentence. In addition, many such languages are also so-called compounding languages, i.e., compound words can be formed from shorter particles to express complex concepts as single words. The compound words can again occur in different inflections. As a result, the lexical variety of such languages is very high and it is not possible to achieve a good vocabulary coverage when using words as basic units for language modeling. In order to increase coverage, subword units, such as morphemes, are used as basic units in language modeling. Subword units may be found using morphological analysis, as has been proposed among many others for Korean [2] and Estonian [3], or discovered automatically in a data-driven manner [4]. Given the abundance of compound words, it is questionable wheather morphemes carry enough information to make them appropriate for topic adaptation. For example, an Estonian word form

> *ajakirjandusõppejõududega*, 'with lecturers of journalism'

can be decomposed into four compound particles and two inflectional suffixes:

> *aja kirjandus õppe jõudu de ga* 'time literature study force plural commitative'.

When we look at the morpheme decomposition of the word as an unordered *bag of morphemes*, it gives us much less information about the semantics of the phrase than the original inflected compound word. On the other hand, the high variety of different word inflections and the resulting sparseness of word-document occurrences makes the performance of word-based term-document analysis questionable. One way to decrease the lexical variety of the language would be to lemmatize all words, i.e., to use a morphological analyzer to find canonical forms or *lemmas* for all words before mapping them into the LSA space. Lemmatization increases the coverage of the LSA model vocabulary, and should act as an additional smoothing measure, since all different inflections of the same words are mapped to the same vocabulary item.

Based on these observations, this paper investigates a LSA-based language model adaptation approach for highly inflected languages. We compare three different sets of units – words, lemmas and morphemes – as basic units for modeling document similarities using the LSA technique. Performance of different basic unit sets is measured on Estonian speech recognition experiments.

Similar LSA-based adaptation methods have also been investigated recently, e.g. [5,6]. In [7], a somewhat similar method was proposed to select a subset of training corpus for fast marginal adaptation, however, training set perplexity minimization was used as a measure for selecting the documents from the corpus. There have been previous attempts in language model adaptation for inflected languages. In [8], inflected words are clustered according to fuzzy string comparison rules, and language model is adapted by only using the texts from the closest topic from a predefined topic set for training. Naive Bayes classifier and standard centroid-based classifier (TDIDF) is used for topic detection. Another related work [9] uses morpheme-like units and lemmas for LSI-based Finnish spoken document retrieval.

The presented approach differs from previous ones in dividing the adaptation process into three steps: LSA-based similar document retrieval, topic-specific unigram estimation from the retrieved documents and adaptation of the background language model using the estimated topic unigram. This enables us to use a different vocabulary for document similarity modelling and retrieval than is used for language modelling.

The paper is organized as follows: in section 2, we provide an overview of Latent Semantic Analysis. In section 3, we describe how in-domain documents are selected from the training corpus and how fast marginal adaptation (FMA) [10] is used for language model adaptation. In section 4, we describe the experiments and the results, followed by a conclusion in section 5.

2 Latent Semantic Analysis

Latent Semantic Analysis (LSA) [11] is a mathematical technique for extracting and representing the semantic similarity of words and documents by analysis of large document corpora. The task of LSA is to define a mapping between the vocabulary $\mathcal{V}$ of M words, the document set $\mathcal{T}$, comprising N articles, and a vector space so that each word in $\mathcal{V}$ and each document in $\mathcal{T}$ is represented by a vector in this space. This is done by first constructing a word-document matrix W, where each element W_{ij} is a weighted count of word w_i in document d_j. The weighted count expresses both the word's importance in the particular document as well as the degree to which the word carries information in the domain of discourse in general. A suitable expression for W_{ij} as proposed in [1] is

$$W_{ij} = G_i L_{ij} \tag{1}$$

where G_i is the global weight, indicating the overall importance of w_i in the corpus, and L_{ij} is a local value, describing the normalized frequency of w_i in d_j.

The global weight G_i is calculated via normalized term entropy E_i as

$$G_i = 1 - E_i \tag{2}$$

Term entropy reflects the indexing value of the word w_i and can be calculated as

$$E_i = -\frac{1}{log(N)} \sum_{j=1}^{N} \frac{c_{ij}}{t_j} \log \frac{c_{ij}}{t_j} \tag{3}$$

where c_{ij} is the number of times w_i occurs in d_j, $t_i = \sum_j c_{ij}$ is the total number of times w_i occurs in $\mathcal{T}$. Thus, words distributed across many documents in the corpus (e.g. function words) receive a high term entropy value, while words present in relatively few specific documents receive a low entropy value.

The local value L_{ij} is a transformed version of c_{ij}, normalized for document length and dampened by applying a logarithm in order to reduce the effect of large differences in document length:

$$L_{ij} = log_2(1 + \frac{c_{ij}}{n_j}) \tag{4}$$

where n_j is the length of document d_j.

Next, LSA applies rank-R singular value decomposition (SVD) to the word-document matrix W:

$$W \approx \hat{W} = USV^T \tag{5}$$

where U is the $(M \times R)$ matrix of left singular vectors u_i, S is the diagonal matrix of R singular values and V is the $(N \times R)$ matrix of right singular vectors v_j. Matrix $\hat{W}$ is the best rank-R approximation to the original W. Rank R is the order of decomposition, $R \ll M (\ll N)$. The vectors u_i represent the word w_i in the new LSA space and the vectors v_j represent the document d_j in the same space.

The main benefit of SVD for our work is that it eliminates the sparseness issue, by reducing the dimensions of word and document vectors which isolates the most characteristic components of W and ignores the higher order effects that are unreliable and can be considered noise. This means that two words that do not necessarily co-occur in the original space $\mathcal{T}$ could still be close in the LSA space if they consistently tend to co-occur with a common set of words.

3 Language Model Adaptation

For language model adaptation, we apply fast marginal adaptation using an unigram model trained by weighting the units counts of the in-domain document set. The in-domain documents are retrieved by selecting L documents that are closest to the adaptation data in the LSA space.

3.1 Selecting Adaptation Documents

To find the closest documents to the given adaptation data in the LSA space, we first convert the adaptation data to pseudo-document representation $\tilde{d}_p$ by using the weighted counts (1) with $j = p$. According to [1], the representation of the adaptation data in the LSA space can be given then as

$$\tilde{v}_p = \tilde{d}_p^T U S^{-1} \tag{6}$$

Next, we calculate the "distance" between pseudo-document representation $\tilde{v}_p$ and every training document representation v_i by finding the the angle between $\tilde{v}_p S$ and $v_i S$:

$$K(\tilde{v}_p, v_i) = \angle(\tilde{v}_p S, v_i S) = \arccos \frac{\tilde{v}_p S^2 v_i^T}{\|\tilde{v}_p S\| \, \|v_j S\|} \tag{7}$$

In this way, the training documents are ranked by their distance measure and the top L documents can be selected for use as adaptation data. Of course, if we only need the closest documents and are not interested in the actual distance values, the cosine calculation can be discarded and the ranking be inverted. However, in the next section we explain how we use the distance values for improving the unigram compilation.

3.2 Weighted Unigram Counts

Before constructing the unigram models, we apply count weighting, depending on the closeness of the training document to the adaptation data. This approach is very similar to the relevance-based n-gram count weighting method proposed in [12].

Given a set to document identities found in the previous document selection step $\mathcal{T}_{Adap}$, we calculate the total weighted count $c_{Adap}(i)$ of the ith word as

$$c_{Adapt}(i) = s * \sum_{j \in \mathcal{T}_{Adap}} \left(1 - \frac{K(\tilde{v}_p, v_j)}{\pi}\right) * c_{ij} \tag{8}$$

where c_{ij} is the number of times w_i occurs in document d_j. The distance measure $K(\tilde{v}_p, v_j)$ is calculated as given in (7) and lies in $[0, \pi]$. The scaling factor s is calculated as

$$s = \frac{c}{\sum_{i=1}^{M} \sum_{j=1}^{L} \left(1 - \frac{K(\tilde{v}_p, v_j)}{\pi}\right) * c_{ij}} \tag{9}$$

where c is the total number of words in the adaptation data and M the number of words in the vocabulary. The scaling factor ensures that the sum of weighted counts is equal to the sum of unweighted counts.

This approach enables to emphasize the counts in documents that lie closer to the adaptation data. Using the resulting fractional counts, we apply Ristad's natural discounting [13] to estimate unigram models. This discounting method was chosen because it gave good results on the development set and can be applied to fractional counts.

3.3 Fast Marginal Adaptation

Fast Marginal Adaptation [10] is a method to quickly adapt a given language model to in-domain text characteristics. It uses the trigram trained on the background corpus as the initial language model. The background model is adapted so that its marginal is the unigram trained on the adaptation data. It turns out that this can be reformulated as a scaling of the background LM:

$$P_{Adap}(w_i|h) = \frac{\alpha(w_i) P_{BG}(w_i|h)}{Z(h)} \tag{10}$$

where $P_{Adap}(w_i|h)$ is the adapted probability of the word w_i, given the history h, $P_{BG}(w_i|h)$ the word probability according to the background model and $Z(h)$ a normalization factor that guarantees that the probability sums to unity. The scaling factor $\alpha(w_i)$ is usually approximated as follows:

$$\alpha(w_i) \approx \left(\frac{P_{Adap}(w_i)}{P_{BG}(w_i)}\right)^{\beta} \tag{11}$$

where $P_{Adap}(w_i)$ is the unigram probability of w_i based on in-domain corpus, $P_{BG}(w_i)$ the background unigram probability and β a tuning factor between 0 and 1. The task of $\alpha(w_i)$ is to scale the probability of w_i up or down, depending on its relative frequency in the adaptation corpus with respect to the background corpus. The normalization factor $Z(h)$ can be calculated using the approximated scaling factor by summing over the set of all words w:

$$Z(h) = \sum_{w} \alpha(w_i) P_{BG}(w_i|h). \tag{12}$$

4 Experimental Evaluation

4.1 Task Description

Language model adaptation was applied to an Estonian broadcast news transcription task. The material consists of studio-recorded hourly short radio news broadcasts provided by the Estonian national radio. We randomly selected some broadcasts and hand-transcribed them. The broadcasts were also manually segmented into stories and sentences. Audio segments that contained non-speech, such as music signatures, were removed. For testing, we used around 21 minutes of audio (193 sentences). For tuning the parameters, we used around 11 minutes of audio (101 sentences). The number of stories in the test and development set is 44 and 20, respectively. The average number of sentences per story was 4.6.

4.2 Acoustic Modeling

Since we have not got enough Estonian transcribed broadcast news data, we trained the acoustic models for recognition experiments on the Estonian SpeechDat-like phonetic database [14]. The database was collected from volunteer speakers over telephone. Each recording session contains read sentences from a handout sheet, answers to simple questions, short utterances, etc. The number of different speakers is 1332. The number of acceptable utterances is 177 793. This totals in about 241.1 hours of audio data. The speech data is recorded at 8kHz sampling rate and coded using 8-bit mono A-law.

The open source SphinxTrain toolkit was used for training the acoustic models. Models were created for 25 phonemes, the five filler/noise types and silence. For acoustic features, MFCC coefficients were used. The coefficients were calculated from a frequency band of 130 Hz - 3400 kHz, using a preemphasis coefficient of 0.9. The window size was 0.0256 seconds and the frame rate was 100 frames/second. A 512-point FFT was used to calculate 31 filter banks, out of which 13 cepstral coefficients were calculated. All units are modeled by continuous left-to-right HMMs with three emitting states and no skip transitions. The output vectors are 39-dimensional and are composed of 13 cepstral coefficients, delta and double delta coefficients. The final tied-state triphone models have 8000 shared states in total. Each state is modeled by 8 Gaussian mixture components. The final models were adapted via MLLR, using around 30 minutes of hand-transcribed broadcast news data from various speakers.

The pronunciation dictionary was created from word orthography using a set of context sensitive grapheme-to-phoneme rewrite rules and a small and incomplete set of foreign name pronunciation rules [15].

4.3 Language Modeling

For training the background language model, we used the following subset of the Mixed Corpus of Estonian [16], compiled by the Working Group of Computational Linguistics at the University of Tartu: daily newspaper "Postimees" (33M words), weekly newspaper "Eesti Ekspress" (7.5M words), weekly newspaper "Maaleht" (4.3M words), Estonian original prose (4.2M words), academic journal "Akadeemia" (7M words), transcripts of the Estonian parliament (13M words), weekly magazine "Kroonika" (0.6M words). In addition, we compiled a corpus of online newspaper articles from a website of a daily "Eesti Päevaleht" (93M words) and a corpus of news stories from an online news site *etv24.ee* (4.8M words).

The SRILM toolkit [17] was used for selecting language model vocabulary and compiling the language model. The language model was constructed by first processing the text corpora using the Estonian morphological analyzer and disambiguator [18]. The analyzer uses a rule-based approach, a statistical method based on hidden Markov models is used for disambiguation. Using the information from morphological analysis, it's possible to split compound words into particles and separate morphological suffixes from preceding stems. Language model vocabulary was created by selecting the most likely 60K units from the mixture of the corpora, using the transcripts from the speech recognition development set as handout text. Using the resulting vocabulary of 60K particles, a trigram language model was estimated for each training corpus subset. Trigrams occurring only once in the corpus were not included in the models. A modified version of Kneser-Ney smoothing as implemented in SRILM was applied. Finally, a single LM was built by merging the eight models, using interpolation coefficients optimized on the development set transcripts.

The morpheme-level (unnormalized) perplexity of the language model against the speech recognition development and test set transcripts is 131 and 128, respectively. The corresponding out-of-vocabulary (OOV) rates are 0.8% and 0.5%.

4.4 LSA Estimation

The word-based, lemma-based and morpheme-based LSA models were created based on the word statistics in the three above-mentioned newspaper corpora, the online newspaper corpus and the *etv24.ee* news corpus. The corpora were divided into documents according to the markers as provided by the corpus creators. The number of different documents in the resulting corpus was approximately 0.5M. For creating the lemma-based LSA model, words in the training data were first replaced with their respective lemmas using the morphological analyzer/disambiguator. The same tool was used for splitting the words into morphemes for creating the morpheme-based model.

The LSA models were constructed using a vocabulary of 60 000 most frequent units. The vocabulary of the morpheme-based model was taken from the trigram language model. The initial word-document matrix contained about 83.7 million nonzero entries, the lemma-document matrix had about 78.9 million nonzero entries and the morpheme-document matrix about 117.2M nonzero entries. The SVD transforms of co-occurrence matrices were calculated using PROPACK[1]. We used rank 200 SVD since this is a rough average of what has given the best results in LSA research before.

The OOV rate of the words in the recognized sentences of the development and test set against the word-based LSA model vocabulary is 13.7% and 11.4%. The corresponding values for the lemma-based model are 8.4% and 5.9%. The OOV-rate of the morpheme-based model is the same as that of the N-gram language model – 0.8% and 0.5%.

4.5 Recognition Procedure

The CMU Sphinx 3.6.3 decoder was used for recognition experiments. For each utterance, 1000 best sentence hypotheses were generated. The highest ranked hypotheses were converted to the suitable form (i.e., lemmatized or morphologically segmented, depending on the used LSA model) and the units from each story were used for calculating the story vector in the LSA space. The out-of-vocabulary units were discarded. Next, the semantically closest adaptation documents were found, together with their distance measures. The resulting set of documents was used for estimating a new unigram distribution for each story. The unigrams were then used for fast marginal adaptation of the background language model. The score combination weights of the updated language models and the acoustic model, as well as word insertion penalty, were optimized on the development set, using the word error rate of the word-level posterior probability maximizer as the minimization function. The simplex-based "Amoeba" search strategy implemented in SRILM was applied. Finally, the scores were combined and the final sentence hypothesis was selected using the SRILM-implementation of the ROVER algorithm. The scores of the test set were combined using the optimized weights from the development set. Note that the baseline results reported below confirm to ROVER-based N-best rescoring using the unadapted model, not the initial 1-best results, thus the impacts of weight optimization and ROVER algorithm were cancelled in the comparison of baseline and adapted models.

4.6 Results

We measured letter (including inter-word space) error rate (LER) of the test set before adaptation and after rescoring with the adapted language models. Letter error rate was used instead of the more traditional word error rate (WER) because we feel it is a more accurate, sensitive and useful measure of speech recognition quality for agglutinative and compounding languages. A common error that the recognizers of such languages tend to make is to recognize a stem of an inflected word correctly but confuse the morpheme suffix with another

[1] `http://sun.stanford.edu/~rmunk/PROPACK/`

similar suffix. When using WER as a quality measure, we would not take into account the severeness of word recognition errors: totally misrecognized words and words having tiny recognition errors in suffixes would both be counted as a single word error. This is in contradiction with human perception of recognition quality and with the usefulness of the recognition hypotheses for further automatic processing. Furthermore, in highly compounding languages, such as Estonian, it is common for a recognizer to make a 'compounding error', i.e., write a compound word as two separate words, or vice versa. In fact, this error is also very common in human-written Estonian texts. However, such compounding errors would be counted as two recognition errors by the WER measure (a substitution error and a deletion or an insertion error) which does clearly not reflect the situation adequately.

The recognition results of the baseline system and the three adapted systems are listed in Table 1. We analyzed the differences in recognition quality between various systems by comparing the LER of corresponding news stories, and applied the Wilcoxon signed rank test of statistical significance. The test showed that all systems using adapted models performed significantly better than the system without any adaptation. The morpheme-based adapted system performed significantly better than the other two adapted systems. There was no significant difference between word-based and lemma-based adaptation.

Table 1. Letter error rates before and after adaptation using the word-based *vs.* lemma-based *vs.* morpheme-based LSA model, together with relative improvement over unadapted system.

System	LER
No adaptation	7.1
Word-based adaptation	6.7 (-6%)
Lemma-based adaptation	6.6 (-7%)
Morpheme-based adaptation	**6.4 (-9%)**

4.7 Discussion

Experimental results showed that morpheme-based adaptation gives significantly better improvements in recognition accuracy than similar word-based and lemma-based models. This is in contradiction with our initial hypothesis that doubted in the ability of the bag-of-morphemes document represention to carry semantic content. However, the result can be explained by many circumstances.

First, when using morpheme-based adaptation, we are able to set the vocabulary of the LSA model to the same inventory of morphemes as the speech recognition language model. This means that the effective coverage of the morpheme-based model against recognition hypotheses is 100%, since the recognizer can only recognize morphemes in its vocabulary. Lemma-based and especially word-based adaptation, on the other hand, suffer from a substantial vocabulary mismatch.

Second, many corpus linguists have recently emphasized the importance of inflectional word forms in contrast to lemmas when analyzing corpora. Lemmatization presents a serious danger of potentially incorrect over-generalization which is criticized by linguists [19]. Morpheme-based adaptation at least partially avoids such over-generalization, since many common noun inflections are realized in Estonian via root alternation, and this information is retained in the morpheme-based representation.

Third, representation of documents using the bag-of-morphemes approach may give more information about the semantic content of the document than it seems at the first sight. It is true that the bag-of-words representation loses any information about the context and sequence of the morphemes. Thus, compound words with high semantic content are broken up into morphemes and the information about the compound word is lost. However, documents that systematically use the same compound words are likely to lie close in the morpheme-based LSA space, since LSA uses a linear combination of SVD-transformed term-document matrices.

5 Conclusion and Future Work

We presented a statistical language model adaptation method that is especially suitable for agglutinative and highly inflected languages that use sub-word units as basic units for N-gram language modelling. The method relies on a separate independent LSA model to retrieve documents from a large corpus that are semantically similar to a small "adaptation seed" and applies fast marginal adaptation of background N-gram models based on the resulting adaptation corpus. We compared three different basic units – words, lemmas and morphemes – for modelling document similarity.

Preliminary experiments carried out on a relatively small set of audio files show that morpheme-based adaptation provide significantly larger improvements in speech recognition accuracy than similar word-based and lemma-based adaptation. The results were somewhat surprising, since we initially doubted in the ability of morphemes to carry semantic content. The results could be useful for other tasks where a bag-of-words approach is traditionally used, such as document clustering.

One of the focuses of future work is integrating fast marginal adaptation directly into the decoder. An efficient implementation has been described in [20]. Also, we wish to replace current manual sentence and story segmentation with an automatic segmentation system.

There are many details in the adaptation scheme that could be improved. The confidence scores of the recognized words could be used for weighting when building the topic vector in LSA space. Further improvements can be expected when word and document clustering that enable to apply smoothing in the LSA space is implemented into the framework.

Acknowledgments

This work has been partially funded by the Estonian National Programme for Language Technology.

References

1. Bellegarda, J.R.: A multispan language modeling framework for large vocabulary speech recognition. IEEE Transactions on Speech and Audio Processing 6, 456–467 (1998)
2. Kwon, O.W., Park, J.: Korean large vocabulary continuous speech recognition with morpheme-based recognition units. Speech Communication 39, 287–300 (2003)
3. Alumäe, T.: Large vocabulary continuous speech recognition for Estonian using morpheme classes. In: Proceedings of ICSLP 2004 - Interspeech, Jeju, Korea, pp. 389–392 (2004)
4. Siivola, V., Hirsimäki, T., Creutz, M., Kurimo, M.: Unlimited vocabulary speech recognition based on morphs discovered in an unsupervised manner. In: Proceedings of Eurospeech, Geneva, Switzerland, pp. 2293–2296 (2003)
5. Chen, B.: Dynamic language model adaptation using latent topical information and automatic transcripts. In: IEEE International Conference on Multimedia and Expo, Amsterdam, The Netherlands, pp. 97–100 (2005)
6. Tam, Y.C., Schultz, T.: Unsupervised language model adaptation using latent semantic marginals. In: Proceedings of Interspeech 2006 - ICSLP, Pittsburgh, PA, USA, pp. 2206–2209 (2006)
7. Klakow, D.: Language model adaptation for tiny adaptation corpora. In: Proceedings of Interspeech 2006 - ICSLP, Pittsburgh, PA, USA, pp. 2214–2217 (2006)
8. Maučec, M.S., Kačič, Z., Horvat, B.: A framework for language model adaptation for highly-inflected Slovenian. In: Proceedings of ITRW on Adaptation Methods for Speech Recognition, Sophia Antipolis, France, pp. 211–214 (2001)
9. Turunen, V., Kurimo, M.: Using latent semantic indexing for morph-based spoken document retrieval. In: Proceedings of Interspeech, Pittsburgh, USA, pp. 341–344 (2006)
10. Kneser, R., Peters, J., Klakow, D.: Language model adaptation using dynamic marginals. In: Proceedings of Eurospeech, Rhodes, Greece, vol. 4, pp. 1971–1974 (1997)
11. Landauer, T.K., Foltz, P.W., Laham, D.: Introduction to latent semantic analysis. Discourse Processes 25, 259–284 (1998)
12. Iyer, R., Ostendorf, M.: Relevance weighting for combining multi-domain data for n-gram language modeling. Computer Speech and Language 13, 267–282 (1999)
13. Ristad, E.S.: A natural law of succession. Technical Report TR-495-95, Computer Science Department, Princeton University (1995)
14. Meister, E., Lasn, J., Meister, L.: Development of the Estonian SpeechDat-like database. In: Proceedings of Eurospeech, Geneva, Switzerland, vol. 2, pp. 1601–1604 (2003)
15. Alumäe, T.: Methods for Estonian large vocabulary speech recognition. PhD thesis, Tallinn University of Technology (2006)
16. Kaalep, H.J., Muischnek, K.: The corpora of Estonian at the University of Tartu: the current situation. In: The Second Baltic Conference on Human Language Technologies: Proceedings, Tallinn, Estonia, pp. 267–272 (2005)

17. Stolcke, A.: SRILM – an extensible language modeling toolkit. In: Proceedings of ICSLP, Denver, USA, vol. 2, pp. 901–904 (2002)
18. Kaalep, H.J., Vaino, T.: Complete morphological analysis in the linguist's toolbox. In: Congressus Nonus Internationalis Fenno-Ugristarum Pars V, Tartu, Estonia, pp. 9–16 (2001)
19. Čermák, F.: Some of the current problems of corpus and computational linguistics or fifteen commandments and general truths. In: The Third Baltic Conference on Human Language Technologies, Kaunas, Lithuania (2007)
20. Federico, M.: Language model adaptation through topic decomposition and MDI estimation. In: Proceedings of ICASSP, Orlando, FL, vol. 1, pp. 773–776 (2002)

Word Distribution Analysis for Relevance Ranking and Query Expansion

Patricio Galeas and Bernd Freisleben*

Dept. of Mathematics and Computer Science, University of Marburg,
Hans-Meerwein-Str. 3, D-35032 Marburg, Germany
{galeas,freisleb}@informatik.uni-marburg.de

Abstract. Apart from the frequency of terms in a document collection, the distribution of words plays an important role in determining the relevance of documents for a given search query. In this paper, *word distribution analysis* as a novel approach for using descriptive statistics to calculate a compressed representation of word positions in a document corpus is introduced. Based on this statistical approximation, two methods for improving the evaluation of document relevance are proposed: (a) a relevance ranking procedure based on how query terms are distributed over initially retrieved documents, and (b) a query expansion technique based on overlapping the distributions of terms in the top-ranked documents. Experimental results obtained for the TREC-8 document collection demonstrate that the proposed approach leads to an improvement of about 6.6% over the term frequency/inverse document frequency weighting scheme without applying query reformulation or relevance feedback techniques.

1 Introduction

In a typical information search process, results are obtained by literally matching terms in documents with those of a query. However, due to *synonymy* and *polysemy,* lexical matching methods are likely to be inaccurate when they are used to meet a user's information need [1].

One way to address this problem is to consider contextual information [2]. In fact, several search engines make use of contextual information to disambiguate query terms [3]. Contextual information is either derived from the user, the document structure or from the text itself by performing some form of statistical analysis, such as counting the frequency and/of distance of words.

In this paper, we present an information retrieval approach that incorporates novel contextual analysis and document ranking methods. The proposed approach, called *word distribution analysis*, is based on a compressed statistical description of the word positions in a document collection, represented through their measures of *center* and *spread.* As a complement to the term frequency/inverse document frequency (*tfidf*) metric, we propose the *term density*

* This work is partially supported by the Deutsche Forschungsgemeinschaft (DFG, SFB/FK 615).

distribution (TDD) measure to estimate a document's relevance. Furthermore, a new query expansion algorithm is proposed. It is based on overlapping the distributions of query terms in the top-ranked documents. Experimental results obtained for the TREC-8 document collection demonstrate that the proposed approach is superior to the *tfidf* weighting scheme without applying query reformulation or relevance feedback techniques.

The paper is organized as follows. Section 2 outlines related work on contextual information retrieval. In section 3, the word distribution analysis approach is presented. Implementation issues are described in section 4. Section 5 discusses experimental results for evaluating the performance of the proposed approach. Section 6 concludes the paper and outlines areas for future research.

2 Related Work

According to Huang [2], contextual information can be interpreted in two ways: the text surrounding the search terms in the document corpus, or the context obtained from the user (i.e. *personalization*). There are approaches that utilize the query history of users [4] or the text surrounding the query [5,6] to build augmented queries (i.e. *query expansion*) for improving the performance of interactive retrieval systems.

Relevance feedback is the most popular *query expansion* strategy [7,8,9]. Here, the expanded terms are typically extracted from the retrieved documents and judged as relevant in a previous retrieval iteration. As demonstrated in several experimental studies, relevance feedback systems are quite effective [10,11]. However, the browsing process required to determine the relevance of a document has been widely recognized as a significant limitation by the information retrieval research community.

To overcome the intervention of the user in the relevance feedback process, two basic types of strategies have been proposed: *automatic global analysis* and *automatic local analysis*. In automatic global analysis, all documents of the collection are used to determine a *thesaurus*-like structure, defining term-to-term relationships within the document corpus. In general, global analysis techniques are limited to small database applications, where doubtful improvements have been observed [12].

In automatic local analysis, the system is able to estimate the relevance of the first retrieved documents without user intervention. The main idea is to consider the *top-n* initially retrieved documents as relevant, and using statistical heuristics, identify query related terms [13,14]. *Noise* and *multiple topics* are two major negative factors for expansion term selection [15]. To deal with these problems, traditional clustering methods have been proposed [16], and the experiments performed by Fan et al. [17] confirm that highly-tuned ranking offers more high quality documents at the top of the hit list.

With respect to the document structure, it is difficult to determine correlated terms inside the document body, because "good" terms do not necessary co-occur very frequently with the original query terms. In fact, it is common to have unrelated words co-occurring with query terms very frequently [18]. To address this problem, page segmentation strategies have been suggested [15,19]. They provide a better document partition at the semantic level and reduce the probability to carry irrelevant terms to the query expansion process. In general, an important drawback of automatic local analysis strategies is the considerable amount of computation, which represents a substantial problem for interactive systems [20].

Katz [21] analyzes the distribution of content-bearing words in technical documents. Important concepts supporting word occurrences models, such as *inter-/within-document* relationships, *topicality* and *burstiness* are proposed. The author concentrates on the modeling of the *inter-document* distributions of content words, while our work focuses on the *within-document* relationships applied to relevance evaluation in the information retrieval process. Another interesting approach on this subject has been proposed by Fernández et al. [22], where words appearing in a similar syntactic context are used for lexical and syntactic disambiguation in a natural language parsing process.

3 Word Distribution Analysis

The previous studies corroborate that, independent of the applied technology, the analysis of relationships between words in a document collection is a significant way to improve relevance estimation in information retrieval systems. On the other hand, the analysis of word distributions in a text reveals that content-bearing words are likely to repeat in close proximity to each other [23].

In this section, we propose a novel approach to obtain a *compressed representation* of the relationships between words using simple methods of descriptive statistics applied to the word positions in documents. One naive method is to calculate the distance between all word pairs. Applying this procedure, one can expect that, if two words are near to each other in a set of documents, some semantic relationship is likely to exist between these two words. Unfortunately, such a method is computationally costly due to (a) the excessive increase of the dimensions of the index matrix which contains information about each word position in the document collection, and (b) the distance computations between the query and the document terms.

Our proposed method permits us to obtain semantic information about the relationships between words based on only two statistical parameters describing the positions of words in the corresponding document: the statistical measures of *center* and *spread*. The most common measures of center and spread are the mean (μ) and the standard deviation (σ). Because the mean and standard deviation suffer from the influence of extreme observations, resistant measures of center and spread, such as the *median* and *percentiles* will be used to better deal with outliers and common irregularities in the data. As shown in the next subsection, these statistical values can be easily incorporated in standard index structures (i.e. an inverted index), extending the capabilities to recognize relevant documents in a set of retrieved documents.

3.1 Descriptive Statistics and Document Semantics

In Table 1, we see a linear representation of a document d, where several instances of the words "a" and "b" are located in their respective positions along the document body. The idea is to analyze how these words are distributed within the document and to find a reduced representation that permits to compare their context.

Table 1. Positions of words a and b in a linear representation of document d

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a			a					a						a			
					b		b		b		b		b				

Table 2 shows the interquartile range [24] for the positions of a and b in document d.

Table 2. Interquartile range for the position of a and b in the document d

Word	Positions					1st quartile(Q_{25})	median (Q_{50})	3rd quartile (Q_{75})
a	1	4	9	15		1.75	6.5	13.5
b	6	8	10	12	14	7	10	13

Using the calculated parameters, their interquartile ranges are shown in Fig. 1. It can be observed that the instances of word a are mainly situated in the first half of the document with $Q_{25} = 1.75$ and $Q_{75} = 13.5$, while the instances of word b are distributed approximately in the middle of the document with $Q_{25} = 7$ and $Q_{75} = 13$. This statistical representation of the word positions gives a concrete picture of their dispersion within the document, such that the distributions of two or more words can be compared.

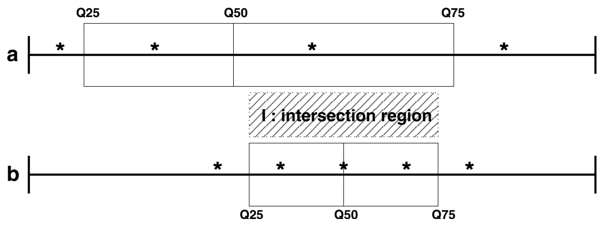


Fig. 1. Interquartile range for the positions distribution of a and b

The interquartile range allows us to reach some conclusions about specific scores in our distribution. Approximately 50% of the instances of word a are located in its corresponding interquartile range $r_a = [Q_{a_{25}}, Q_{a_{75}}]$, and about 50% of the instances of word b are located in the range $r_b = [Q_{b_{25}}, Q_{b_{75}}]$. Using the document length $|L|$, the equations can be normalized: $0 \leq R_a \leq 1$ and $0 \leq R_b \leq 1$, where $R_a = \frac{r_a}{|L|}$ and $R_b = \frac{r_b}{|L|}$. Thus, if the range of a word in the document is near to 1, the instances of this word are widely distributed within the document body. Similarly, if the *intersection range* (I) of two words is determined, we can expect that the word instances situated in this common document region are close to each other: $I(a,b) = R_a \cap R_b$. Based on this information, we propose two approximations described in the following subsection.

3.2 The Document Relevance Estimator

Let R_{a_d} be the distribution range of word a in document d, and ρ_{a_d} the word density (number of occurrences) of word a in document d. Then, the *Term Density Distribution (TDD)* is defined as an estimator for the relevance of word a in document d:

$$TDD(a)_d = R_{a_d} \cdot (1 + log\,\{\rho_{a_d}\}) \tag{1}$$

A wide range and a high frequency of word a imply that word a is regularly distributed within the document body, and it could be considered as a relevant key to describe the document content. For example, considering the documents of Fig. 2 and the query $q = \{a\}$, then the document *1* will be more relevant than the document *2*, because $TDD(a)_1 > TDD(a)_2$.

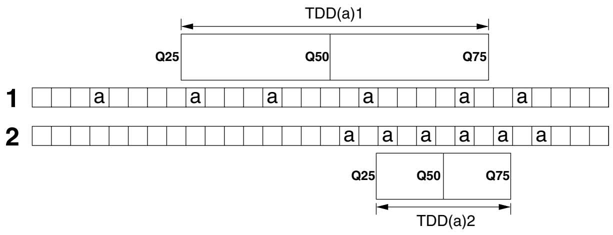


Fig. 2. Dispersion of the word a in two arbitrary documents

3.3 The Semantic Distance Estimator

By calculating the *intersection range* between two words $I(a,b)$ and their *word densities* in document d, one can estimate their *semantic statistical distance* $(\delta_{a,b})$ in the document:

$$\delta_{a,b} = \frac{n}{\sum_{d=1}^{n} I(a,b)_d \cdot (1 + log\,\{\rho_{(a,b)_d}\})} \tag{2}$$

where $I(a,b)_d$ is the *intersection range* between the words in document d, $\rho_{(a,b)_d}$ is the *word density* of *(a+b)* in document d, and n is the number of documents in the collection containing word a and word b.

The higher the *intersection range* between two words and the higher the *word density* in the same range, the closer their position is in the document, implying some *semantic connection* between them.

4 Implementation Issues

4.1 Index, Search and Ranking

In our implementation of the proposed approach, the statistical computations are performed simultaneously with the indexing process. To achieve this task, we utilize *Lucene*[1], an open source search engine library written in Java. We have extended the components of *Lucene* to calculate, store, and apply the word distribution information along the retrieval process.

Using the statistical information contained in the extended index structure, we propose a new search algorithm. First, an initial approximation of relevant documents based on the *tfidf* criterion and the query q is retrieved. Then, a procedure consisting of the following two simultaneous tasks is started.

Ranking Optimization. Selecting the first k documents from the initial *tfidf* ranking and applying the *TDD* estimator of equation (1), we calculate $D = \{d_1, d_2, d_3, \ldots, d_k\}$, an optimized document list based on the dispersion of the query q in the top-ranked documents.

The *TDD* value for each document d_i is obtained by applying equation (3), and the final ranking value is computed using a weighted combination of *TDD* and *tfidf* as shown in formula (4). An optimal value for the weighting coefficient w is estimated experimentally.

$$TDD(q)_{d_i} = R_{q_{d_i}} \cdot (1 + log(\rho_{q_{d_i}})) \tag{3}$$

$$FinalRanking = w \cdot TDD + (1 - w) \cdot tfidf \tag{4}$$

Query Expansion. Using the term frequency values provided by the *Lucene* index, we first calculate $T = \{t_1, t_2, t_3, \ldots, t_m\}$, the m-most frequently occurring terms in D. Then, our δ estimator in equation (5) is applied to compute the *semantic distance* between the query q and the words in T. Finally, the semantic distance threshold ϵ is used to build the term list Q_e representing both the *semantic neighborhood* of the query in the retrieved documents and the candidate terms to expand the query (equation (6)).

$$\delta_{t_i,q} = \frac{d}{\sum_{j=1}^{d} I(t_i,q)_j \cdot (1 + log\left(\rho_{(t_i,q)_j}\right))}\ ,\ i = 1 \ldots t \tag{5}$$

$$Q_e = \{t_1, t_2, t_3, \ldots, t_k\} \tag{6}$$

where $t_j \in Q_e \iff \delta_{t_j,q} \leq \epsilon$, and ϵ is a semantic distance threshold.

[1] http://lucene.apache.org

From the initial results, the necessary information to accomplish two tasks is obtained: (a) estimating the query terms distribution for immediate ranking optimization and (b) calculating the query neighborhood, giving the possibility to incorporate these new terms in a query refinement process.

5 Experimental Results

The TREC-8 document collection has been used to compare the performance of our approach with the *tfidf* weighting scheme. The goal of this evaluation is to determine how well our approach is able to identify relevant documents in the collection.

The evaluation framework consists of the following components: (a) *The Ad Hoc Test Collection* containing 556,077 documents (2.09 Gigabytes) corresponding to the Tipster disks (3 and 4), (b) *The Topics and Relevance Judgments* (qrels), (c) *Our algorithm* consisting of 4 java modules for indexing, search, graphical evaluation and tuning tasks, and (d) *The results analysis* where the effectiveness of our approach will be estimated.

To evaluate the performance of the word distribution and semantic distance concepts, we have accomplished two groups of experiments, consisting of 28 and 14 runs, respectively.

5.1 The Dispersion Runs

In the first group of experiments, the effectiveness of our word dispersion indicator is measured. Based on a short query (*qrel* title) and running the *tfidf* algorithm, we obtain a preliminary group of relevant documents (*tfidf* results). Subsequently, applying the word dispersion criterion, we estimate how the query terms are distributed in the retrieved documents and use this information to optimize the ranking (dispersion results).

Equation (7) represents the dispersion considering a query q having one or more terms and an arbitrary document d from the *tfidf*-ranking. The performed runs have been divided into four groups delimited by the dispersion models executed in the initial ranking:

$$\begin{aligned}
&\text{OTD model}: disp(i)_d = iqr(i)_d/length_d\\
&\text{LIN model}\ \ : disp(i)_d = tf(i)_d \cdot iqr(i)_d/length_d\\
&\text{SQR model}: disp(i)_d = \sqrt{tf(i)_d} \cdot iqr(i)_d/length_d\\
&\text{LOG model}: disp(i)_d = log(tf(i)_d) \cdot iqr(i)_d/length_d
\end{aligned}$$

Applying equation (8), we tested each model with six different dispersion weighting schemes: $w = \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$.

$$Disp(q)_d = \sum_{i \in q} disp(i)_d \tag{7}$$

$$DRank(d)_w = w \cdot Disp(q)_d + (1 - w) \cdot tfidf(q)_d \tag{8}$$

where q are the query terms, $iqr(i)_d$ is the interquartile range of term i in document d, $length_d$ is the length of document d, $tf(i)_d$ is the frequency of term i in document d, $disp(i)_d$ is the dispersion of term i in document d, $Disp(q)_d$ is the dispersion of query terms q in document d, $tfidf(q)_d$ is the tfidf of document d by query q, $DRank(d)_w$ is the dispersion ranking of document d using a weighting scheme w.

Based on this scenario, a total of 28 runs using the previously generated *index* and the *title-tag* of the TREC-topics as a query were performed. For each run, the corresponding *results_file* using the *trec_eval* program was generated, obtaining the *map* and *R-Precision* values for *tfidf* and the different weighting schemes of the dispersion ranking. From Fig. 3, the performance gain (*map* and *R-Precision*) of the dispersion ranking over *tfidf* is evident practically for all models, obtaining the OTD, LOG and SQR models as the best results.

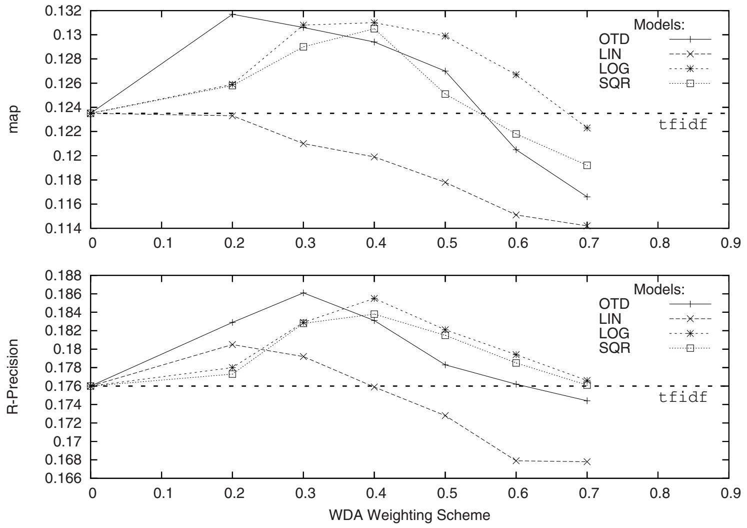


Fig. 3. Comparing the performance of all WDA models vs. *tfidf* using *map* and *R-Precision* values

By comparing the *map* values of *tfidf* and the dispersion ranking for all queries derived from the TREC-8 qrels, our approach outperforms the *tfidf* ranking by 6.6%.

Factors Influencing the Relevance Increase in the Dispersion Runs. In the following figures, the query-words distribution for the top ranked documents is presented, based on topic 430 for the *tfidf* and *dispersion* algorithms as an example. From Fig. 4, a clear difference in the document ranking positions and how the query terms are distributed in the document body can be observed. The document ranking positions change once the dispersion criterion is applied. For example, the document LA080389-0111 holding the first place in

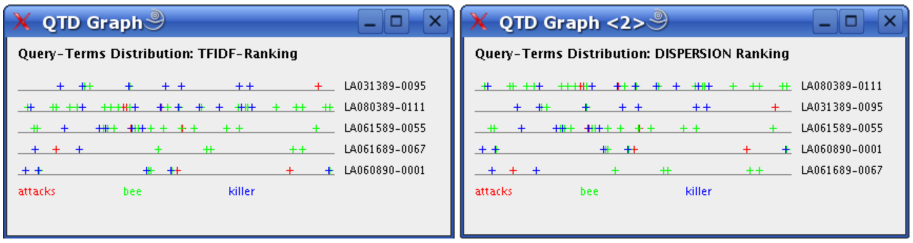


Fig. 4. Tfidf (left) and Dispersion Ranking (right) for Topic 430

the dispersion ranking (right) partially presents (per term) a more distributed term position than the first ranked document in the *tfidf* ranking. Furthermore, analyzing this particular result with *trec_eval*, a *map* gain of 17.2% (*tfidf*:0.1944, *dispersion*:0.2278) is achieved.

Usually, a slight positive variation of the *map* value generates an important difference in the distribution of query terms in the "new" top-ranked documents. We observe that the document *LA080389-0111* which ascends to the first position in the dispersion ranking, presents a more extensive description of the document *LA031389-0095* which previously occupied the first position in the *tfidf* ranking. Furthermore, in *LA080389-0111* a wider distribution of all query terms along the document body than in document *LA031389-0095* can be observed. Thus, documents where all query terms are regularly distributed will be favored in the ranking obtained from applying the dispersion criterion. This avoids two unfavorable situations: (a)*query term agglutination*, i.e. high frequency terms allocated in a small document fragment, and (b) *query term predominance*, i.e. the disproportioned effect of high frequency single query terms over low frequency ones. In our example, the influence of the high frequency term "bee" (*freq*=16) compensates the low frequency of the term "attack" (*freq*=1), nevertheless this document is placed in the first position of the *tfidf*-ranking.

Comparing our results with 13 participants of the ad hoc retrieval task who utilize an analog evaluation framework (based on the topic title), we observe that the *median* improvement over the baseline achieved by these participants is about 11.3%, with lower and upper quartiles of 3.5% and 13%, respectively. The performance gain of our approach inside the inter-quartile range is evident; as already mentioned, a performance gain of about 6.6% over *tfidf* is obtained. Compared to results of the selected TREC-8 participants, this value corresponds to about 58% of the participants's performance using relevance feedback techniques. In contrast, our results are achieved by applying the dispersion model only (without query reformulation).

5.2 The Query Expansion Runs

Query expansion (or term expansion) is a process of supplementing the original query with additional terms, with the aim of improving retrieval performance [25,26]. It should be emphasized that our query expansion experiments are based on the search results only. No internal/external knowledge structure was used to leverage the re-ranking procedure. In the group of runs described in the following, the proposed query expansion model based on the *Semantic Distance Estimator* δ is evaluated. For the top-n ranked documents, the *query-nearest-terms* to expand the query is computed and the ranking is recalculated.

The query reformulation and ranking procedure consists of the following stages:

1. Calculate the expanded query terms (eT) using the top-n documents from the dispersion ranking.
2. Get the top-r relevant terms from the expanded query $eT(r)$.
3. Reformulate the original query (oQ) adding the terms from $eT(r)$ using formula (9).
4. Execute *tfidf*-search using $eQuery$:

$$eQuery = oQ \wedge 0.5 \times (eT_1 \vee eT_2 \vee \ldots \vee eT_r) \tag{9}$$

where: $eQuery$ represent the expanded query, oQ is the original query and eT_i is the expanded term i.

For the top-5 ranked documents (n=5), 14 different query expansion schemes were applied: $r = \{1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 30, 40, 50, 60\}$.

The results are depicted in Fig. 5, where the averaged *map* for all topics and each query expansion scheme are illustrated. Approximately from the 15^{th} expanded term, our approach improves the *tfidf* results.

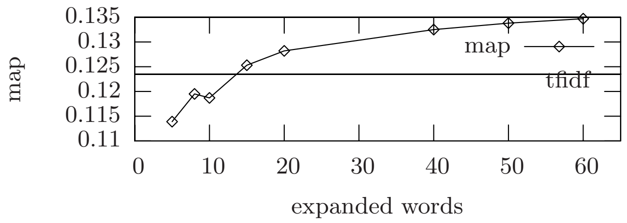


Fig. 5. Query expansion results: the averaged *map* for each query expansion scheme

6 Conclusions

In this paper, we have proposed word distribution analysis (WDA) as a novel methodology to improve the document relevance evaluation in information retrieval applications. WDA is based on a compressed representation of word positions in a document collection, based on two statistical parameters: the measures of *center* and *spread*, which reduce the index size compared to full term position index structures. By analyzing the distributions of query terms in the initial search results, the ranking can be optimized without any relevance feedback cycle. Furthermore, we have extended our distribution model with the *semantic* *distance* concept to develop a query expansion methodology supporting the user in the query refinement process.

An evaluation of WDA using the TREC-8 collection has exhibited a performance gain of 6.6% over the usual *tfidf* weighting scheme without applying query reformulation methods. This improvement represents 58% of the TREC-8 participants's performance improvements implementing relevance feedback techniques. Further analyses have shown that WDA promotes documents having a wider query term distribution and thus minimizes term *agglutination* and *predominance* effects in the top-ranked documents.

There are several issues for future work. For example, it would be interesting to study whether the use of specific distributions improves the ranking evaluation for particular document categories such as medical, juridical, scientific papers, etc. Furthermore, extending our approach to multiple distribution schemes based on Fourier analysis is a topic to be investigated. This enhancement should improve the precision of the model without significantly increasing index size and search performance.

References

1. Berry, M., Dumais, S., O'Brien, G.: Using linear algebra for intelligent information retrieval. Technical Report UT-CS-94-270, SIAM Review (1994)
2. Huang, X., Huang, Y.: Using contextual information to improve retrieval performance. In: Proceedings of 2005 IEEE International Conference on Granular Computing, July 2005, Beijing, China (2005)
3. Lawrence, S.: Context in web search. IEEE Data Engineering Bulletin 23(3), 25–32 (2000)
4. Shen, X., Zhai, C.: Exploiting query history for document ranking in interactive information retrieval. In: 26th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp. 377–378. ACM Press, New York (2003)
5. Finkelstein, L., et al.: Placing search in context: the concept revisited. ACM Transactions on Information Systems 20(1), 116–131 (2002)
6. Razek, M.A., Frasson, C., Kaltenbach, M.: Context - based information agent for supporting intelligent distance learning environment. In: Proc. of the Twelfth International World Wide Web Conference, WWW 2003, Budapest, Hungary, p. 968. Springer, Heidelberg (2003)
7. Salton, G., Buckley, C.: Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science 41(4), 288–297 (1990)
8. Efthimiadis, E.: Interactive query expansion and relevance feedback for document retrieval systems. PhD thesis, City University, London UK (1992)
9. Buckley, C., Salton, G., Allan, J.: The effect of adding relevance information in a relevance feedback environment. In: 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, London, July 1994, pp. 292–300 (1994)
10. Robertson, S., Jones, K.S.: Relevance weighting of search terms. American Society for Information Sciences 27(3), 129–146 (1976)
11. Buckley, C., et al.: Automatic query expansion using smart: TREC-3. In: Overview of the 3rd Text Retrieval Conference, pp. 69–80. NIST Special Publication (1995)

12. Attar, R., Fraenkel, A.: Experiments in local metrical feedback in full-text retrieval systems. Information Processing and Management 17(3), 115–126 (1981)
13. Efthimiadis, E., Biron, P.: Ucla-okapi at TREC-2: Query expansion experiments. In: Proceedings of the 2nd Text Retrieval Conference (TREC-2), pp. 279–290. NIST Special Publication 500-215 (1994)
14. Xu, J., Croft, W.: Improving the effectiveness of information retrieval with local context analysis. ACM Transactions on Information Systems 18(1), 79–112 (2000)
15. Yu, S., et al.: Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In: Proceedings of the 12th International Conference on World Wide Web, Budapest, pp. 11–18. ACM Press, New York (2003)
16. Hearst, M., Pedersen, G.: Reexamining the cluster hypothesis: scatterlgather on retrieval results. In: Proceedings of International ACM SIGIR Conference on Research and Development in IR, New York, pp. 76–84. ACM Press, New York (1996)
17. Fan, W., et al.: Tuning before feedback: combining ranking discovery and blind feedback for robust retrieval. In: Sheffield (ed.) Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in information Retrieval, July 2004, United Kingdom (2004)
18. Sun, R., Ong, C.H., Chua, T.S.: Mining dependency relations for query expansion in passage retrieval. In: SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 382–389. ACM Press, New York (2006)
19. Cai, D., et al.: Block-based web search. In: SIGIR 2004: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 456–463. ACM Press, New York (2004)
20. Li, J., Guo, M., Tian, S.: A new approach to query expansion. Machine Learning and Cybernetics, 2302–2306 (August 2005)
21. Katz, S.M.: Distribution of content words and phrases in text and language modelling. Natural Language Engineering 2(1), 15–59 (1996)
22. Fernandez, M., Villemonte de La Clergerie, E., Vilares, M.: Knowledge acquisition through error-mining. In: Proc. of International Conference on Recent Advances in Natural Language Processing (RANLP 2007), Borovets, Bulgaria, pp. 220–229 (2007)
23. Bookstein, A., Klein, S., Raita, T.: Clumping properties of content-bearing words. Journal of the American Society for Information Science 49(2), 102–114 (1998)
24. Tukey, J.W.: Exploratory Data Analysis. Series in Behavioral Science. Addison-Wesley, Reading (1977)
25. Efthimiadis, E.: Query expansion. Annual Review of Information Science and Technology (ARIST) (2), 121–187 (1996)
26. Billerbeck, B., et al.: Query expansion using associated queries. In: CIKM 2003: Proceedings of the 12th Int. Conference on Information and Knowledge Management, pp. 2–9. ACM Press, New York (2003)

Hybrid Method for Personalized Search in Scientific Digital Libraries

Thanh-Trung Van and Michel Beigbeder

Centre G2I/Département RIM
Ecole Nationale Supérieure des Mines de Saint Etienne
158 Cours Fauriel, 42023 Saint Etienne, France
{van,mbeig}@emse.fr

Abstract. Users of information retrieval systems usually have to repeat the tedious process of searching, browsing, and refining queries until they find relevant documents. This is because different users have different information needs, but user queries are often short and, hence, ambiguous. In this paper we study personalized search in digital libraries using user profile. The search results could be re-ranked by taking into account specific information needs of different people. We study many methods for this purpose: citation-based method, content-based method and hybrid method. We conducted experiments to compare performances of these methods. Experimental results show that our approaches are promising and applicable in digital libraries.

1 Introduction

Search in digital libraries is usually a boring task. Users have to repeat the tedious process of searching, browsing, and refining queries until they find relevant documents. This is because different users have different information needs, but user queries are often short and, hence, ambiguous. For example, the same query "java" could be issued by a person who is interested in geographical information about the Java island or by another person who is interested in the Java programming language. Even with a longer query like "java programming language", we still do not know which kind of document this user wants to find. If she/he is a programmer, perhaps she/he is interested in technical documents about the Java language; however, if she/he is a teacher, perhaps she/he wants to find tutorials about Java programming for her/his course.

From these examples, we can see that different users of an information retrieval system have different information needs. Furthermore, a person can have different interests at different times. A good information retrieval system have to take into account these differences to satisfy its users. This problem could be solved if the system can learn some information about the interests and the preferences of the users and use this information to improve its search results. This information is gathered in *user profiles*. Generally, a user profile is a set of information that represent interests and/or preferences of a user. This information could be collected by implicitly monitoring the user's activities [1,2] or by directly requesting the users [3]. User profiles could be used not only for personalized search [4], but also for different tasks like information filtering [5] or personalized visualization of search results [6]. In the frame of digital libraries, user profiles could be collected from the papers that the users read in this library, from users search histories, from users' browsing histories or be explicitly specified by user etc.

Our works focus on personalized search in digital libraries: the users' search results are re-ranked using similarities between documents in the search results and the user profile. Unlike most other personalized information retrieval systems that use only content-based methods to build users' profiles and to represent the documents in order to compute the similarities between them, we also use citation-based methods and hybrid method for this purpose.

The rest of this paper is organized as follow. In the next section, we present some related work. Then in the section 3 we present our approaches for personalized search in digital libraries. Experiments and results are presented in the section 4. Finally, conclusions and future work are presented in the section 5.

2 Related Work

The work of Amato et al. [7] presents a user profile model that can be applied to digital libraries. In this model, information about a user is classified in five data categories: i) the *personal data category* which contains the user's personal identification data ii) the *gathering data category* which collects preferences and restrictions about the documents the user is looking for iii) the *delivering data category* that are specifications about delivery modes of information iv) the *actions data category* which contains the recording of the user's interaction with the retrieval systems and navigation data v) the *security data category* which is a collection of user preferences establishing the conditions under which the data represented in the user profile may be accessed.

In [8], the authors propose some approaches for re-ranking the search results in a Digital Library that contains digitized books. They consider two kinds of search: search for books by querying on the metadata of books (Metadata Search, MS) and search for informations in the pages of book by querying using keywords (Content Search, CS). They use two different profiles corresponding to these two kinds of search: MS-profile and CS-profile. A MS-profile is built from the ratings of the books that the user provides explicitly. A CS-profile is built from the content of the pages that have been judged as relevant by the user. Metadata search results and content search results are re-ranked using these profiles.

Torres et al. [9] present many algorithms for recommendation of research papers: collaborative methods, content-based methods and hybrid methods. The user profile represents short-term interests and consists of only one paper. The authors did many offline and online experiments to compare the performances of these methods and found many interesting results.

The CiteSeer digital library [10] that contains scientific papers uses a heterogenous profile to represent the user interests. If there is a new available paper, CiteSeer will try to decide if this paper would be interesting to the user using his/her user profile. If so, then the user can be alerted about this paper. CiteSeer uses two methods for determining paper relevance: i) *constraint matching* and ii) *feature relatedness*. The former method allows a user to describe what is an interesting paper by specifying contraints (e.g. keyword). In the latter method, the user specifies a set of papers that are interesting and CiteSeer tries to find papers that are related to this set using content-based method and citation-based method.

3 Our Approaches for Personalized Search in Digital Libraries

Our work focus on personalized search in digital libraries of scientific papers. Like in the CiteSeer system [10], the user profile is represented by a set of paper that are interesting to the user. Each time the user issues a query, the first **n** documents[1] will be re-ranked using the original score computed by the search engine and the similarity between the document and the user profile. The document-profile similarity is computed using two methods: a content-based method and a citation-based method. The personalized search process is illustrated in Figure 1.

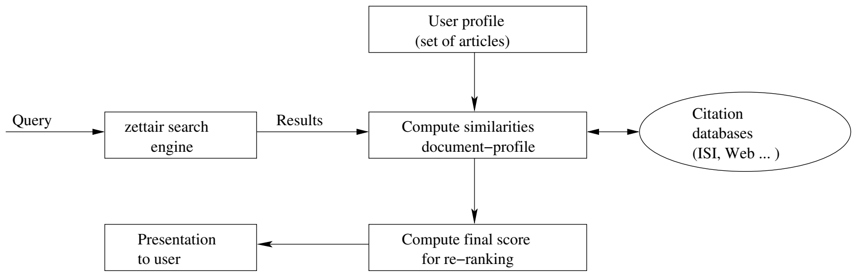

User profile (set of articles)
Query
zettair search engine
Results
Compute similarities document−profile
Citation databases (ISI, Web ...)
Presentation to user
Compute final score for re−ranking

Fig. 1. Re-ranking of search results using user profile

3.1 Computing Similarity Document-Profile

The similarity between a document and a profile is the sum of the similarity between this document and each document in the user profile:

$$similarity(d, p) = \sum_{d' \in p} similarity(d, d') \tag{1}$$

The document-profile similarity is computed using two methods: a content-based method and a citation-based method. We use the **zettair**[2] search engine to compute the content-based similarity (under the vector-space model). To compute the content-based similarity between a document and other documents in

[1] In our experiments n = 300.
[2] `http://seg.rmit.edu.au/zettair/`

the collection, we use the **zettair** search engine to index the collection and submit the document as the query to **zettair** and note the returned similarities. Content-based methods are widely used to compute document-profile similarity in personalized information retrieval systems. However, one of the main problems with this method is that it favors only papers that are similar in content with the papers in the user profile, and not papers that may be different in content (e.g. using different terms) but related with them.

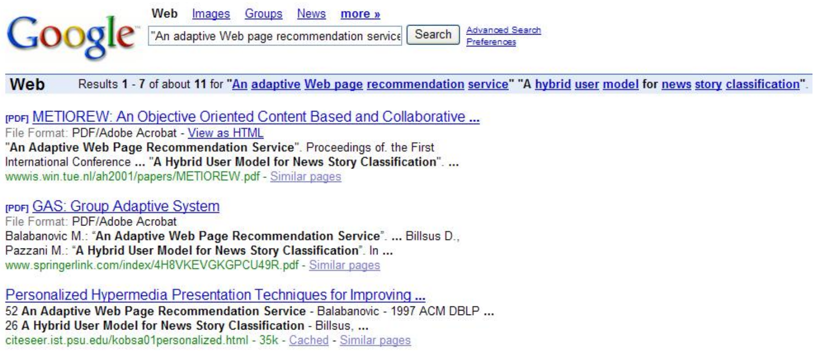
An important characteristic of scientific papers in a digital library is that they have bibliographical relationships between them. We can use these relationships to find the similarity between scientific papers. Content-based methods and citation-based methods are complementary to find relatedness between scientific papers. The citation-based similarity that we use is based on the principle of the co-citation method [11]. In this method, the relatedness between two papers is based on their *co-citation frequency.* The co-citation frequency is the frequency that two papers are *co-cited.* Two papers are said to be *co-cited* if they appear together in the bibliography section of a third paper. However if we want to know this citation information, we have to extract the *citation graph* from the actual library or to get this information from a *citation database*[3]. Both methods are usually limited; i.e. we can only know citing papers of a paper **A** if the citing papers exist within the same digital library or citation database with the paper **A**. Many works [12,13] showed that if the size of a digital library or a citation database is not big enough, then the performance of this method will be limited.

Recently, a new method for citation analysis called Web citation analysis begins attracting the scientometrics community. Web citation analysis finds citations to a scientific paper on the Web by sending the query containing the title of this paper (as phrase search using quotation marks) to a Web search engine and analyze returned pages [14]. Because a Web search engine can index many kinds of document in many different formats, the notion of "citation" used here is a "relaxation" in comparison with the traditional definition. Vaughan and Shaw [14] used this method with the Google search engine and compared the method with the traditional bibliographic method using ISI database. Given an article, they classified Web documents that cite this article into 7 different categories: Journal (site of correspondence journal); Author (author, co-author, or one of their employers lists the articles in their pages); Service (a Web bibliographic service lists the article); Class (bibliography/reading list for a course); Paper (a scientific paper that is posted on the Web); Conference (conference announcement, report or summary/description); Other (cited in another way). Kousha and Thelwall [15] used a similar strategy called URL citations to find citations to articles of open access journals. However, in their work the URL citation of a Web page is the mentions of its URL in the text of other Web page (and not its title).

In our work, we propose to use the Web as a citation database to find the similarity between scientific papers. Our method is called *Web co-citation* method.

[3] A citation database is a system that can provide bibliographic information about papers.

In the Web co-citation method, we compute the co-citation similarity of two scientific papers by the frequency that they are "co-cited" on the Web. The notion of "co-citation" used here is also a "relaxation" in comparison with the traditional definition. If the Web document that mentions two scientific papers is another scientific paper then these two papers are normally co-cited. However, if this is a table of content of a conference proceeding, we could also say that these two papers are co-cited and have a relation because a conference normally has a common general theme. If these two papers appear in the same conference, they may have the same general theme. Similarly, if two papers are in the reading list for a course, they may focus on the same topic of this course. In summary, if two papers appear in the same Web document, we can assume that they have a (strong or weak) relation. The search engine used in our experiment is the Google search engine. To find the number of time that a paper is "cited" by Google we send the title of this paper (as a phrase search using quotation marks) to Google and note the number of returned hits. Similarly, to find the number of times that two papers are "co-cited", we send the titles of these two papers (as a phrase search and in the same query) to Google and note the number of returned hits. This is valid because Google default is to use automatic "AND" queries. This idea is illustrated in Figure 2. In this example, we are looking for the co-citation frequency of two papers, the title of the first paper is "An adaptive Web page recommendation service" and the title of the second paper is "A hybrid user model for news story classification". Here the co-citation frequency is 11. In our experiments, we use a script to automatically query Google instead of manually using a Web browser.


Google
Web Images Groups News more »
"An adaptive Web page recommendation service
Search
Advanced Search
Preferences

Web Results **1** - **7** of about **11** for "**An adaptive Web page recommendation service**" "**A hybrid user model for news story classification**".

[PDF] METIOREW: An Objective Oriented Content Based and Collaborative ...
File Format: PDF/Adobe Acrobat - View as HTML
"**An Adaptive Web Page Recommendation Service**". Proceedings of. the First International Conference ... "**A Hybrid User Model for News Story Classification**". ...
wwwis.win.tue.nl/ah2001/papers/METIOREW.pdf - Similar pages

[PDF] GAS: Group Adaptive System
File Format: PDF/Adobe Acrobat
Balabanovic M.: "**An Adaptive Web Page Recommendation Service**". ... Billsus D., Pazzani M.: "**A Hybrid User Model for News Story Classification**". In ...
www.springerlink.com/index/4H8VKEVGKGPCU49R.pdf - Similar pages

Personalized Hypermedia Presentation Techniques for Improving ...
52 **An Adaptive Web Page Recommendation Service** - Balabanovic - 1997 ACM DBLP ...
26 **A Hybrid User Model for News Story Classification** - Billsus, ...
citeseer.ist.psu.edu/kobsa01personalized.html - 35k - Cached - Similar pages

Fig. 2. Illustration of the Web co-citation method

We use a variant of the formula presented in [16] to compute the co-citation similarity between two papers:

$$cocitation\_similarity(d, d') = ln\left(\frac{cocitation(d, d')^2}{citation(d) + citation(d')}\right) \qquad (2)$$

In Equation 2, $cocitation(d, d')$ is the number of times that these two papers are co-cited[4], $citation(d)$ and $citation(d')$ are respectively the citation frequency that papers d and d' receive. Note that in the Web co-citation method, the document-profile similarity (cf. formula 1) has a negative value, we convert it into a positive value with same variation by this formula:

$$similarity'(d, p) = \frac{1}{|similarity(d, p)|} \tag{3}$$

3.2 Re-ranking Search Results

The final score that is used for re-ranking is a combination between the following scores: i) the original score computed by the search engine ii) the document-profile similarity computed by the Web co-citation method iii) the document-profile similarity computed by the content-based method. The combination formulas are the two following formulas:

– Linear formula:

$$final\_score = \sum_i \alpha_i \times score_i \tag{4}$$

– Product formula:

$$final\_score = \prod_i score_i \tag{5}$$

In the formula 4, α_i are positive coefficients that satisfy the condition $\sum_i \alpha_i = 1$. We tried many different combinations to find the best coefficients. The scores are normalized (divided by the correspondent maximal value) to have the values in the range from 0 to 1.

We conducted experiments to evaluate the performance of different combination methods. The experiments are presented in the following section.

4 Experiments and Results

The search engine that we use is the **zettair** search engine, the default model used in **zettair** is the *Dirichlet-smoothing* model [17]. The test collection that we use is the collection used in INEX 2005[5]. This collection has 17000 XML documents extracted from journals and transactions of *IEEE Computer Society* published between 1995 and 2004. Thus this collection could be used as a medium-size digital library of Computer Science. This collection includes not only scientific papers but also other elements like *tables of content*, *editorial boards*, etc. Because we are interested only in scientific papers, we have to remove these elements from the collection. After this process, the collection contains 14237 documents. Then we extract necessary information for our experiments from these documents. There are also many topics with relevance assessments

[4] If two papers are not co-cited, we assign a small constant to avoid the zero value.

[5] `http://inex.is.informatik.uni-duisburg.de/2005/`

distributed with the collection, each topic represents an information need. Two types of topics were used in INEX 2005 [18]:

- CAS topics (Content And Structure) which allow users to use structure of documents in their queries. They contain explicitly references to the XML structure, and explicitly specify the contexts of the user's interests and/or the contexts of certain search concepts.
- CO+S topics (Content Only + Structure) which do not contain structure of documents (however, it contains also an optional CAS title field which represents the same information need but including additional knowledge in the form of structural hints).

In our experiments we only use the CO+S topics to build the queries. There are 29 assessed CO+S topics but only 26 topics are used. The ignored topics are those that contain too few relevant document or are not typical queries in digital libraries (e.g. search for "call for paper"). The following topics are used in our experiments: 202 203 205 206 207 208 209 210 212 213 216 217 218 221 222 223 227 228 229 230 232 235 236 237 239 241.

Our work simulates personalized search using user profiles. We consider that each topic represents a different information need of one person. The user profile is built from the documents which are judged as relevant (participants in the TREC filtering task [19] use similar strategies to build user profiles). We use a k-fold cross-validation approach [20] for the evaluation. In this approach, the relevant documents of each topic are partitioned into **k** subsets (in our experiments, $\mathbf{k} = 5$). The documents in a subset are used as test documents and the documents in the other $\mathbf{k} - 1$ subsets are used as the user profile. The experiment is repeated **k** times, each time a different subset is used as test subset. The evaluation metric is precision at **n** (with $\mathbf{n} = 5\ 10\ 15\ 20\ 30$) and mean average precision (MAP). The precision at n is the percentage of retrieved docs that are relevant after n documents (whether relevant or nonrelevant) have been retrieved. The mean average precision is the mean of the average precision values of the set of queries. We use the **trec_eval**[6] program to compute these precision values.

Because there are **k** different experiments, hence there are **k** different MAP values and with each value of **n** there are **k** different precisions. Therefore, we have to compute the average values as follows:

$$Average\_of\_precisions\_at\_n = \frac{\sum_{i=1}^{k} precision\_at\_n_i}{k} \tag{6}$$

$$Average\_of\_MAPs = \frac{\sum_{i=1}^{k} MAP_i}{k} \tag{7}$$

Results are presented in Table 1 and Table 2. With each table, the second column is the original results of the **zettair** search engine. The third column is the results of the re-ranking method using two scores: the original score of

[6] `http://trec.nist.gov/trec_eval/`

Table 1. Average of precisions at 5, 10, 15, 20, 30 documents

	Result of **zettair**	Web Co-citation	Content-Based	Hybrid Approach
5 docs	0.2892	0.3108 (p) (+7.5%) 0.3185 (l) (+10.1%)	0.3185 (p) (+10.1%) 0.3462 (l) (+19.7%)	0.3369 (p) (+16.5%) 0.3631 (l) (+25.6%)
10 docs	0.2123	0.2446 (p) (+15.2%) 0.2477 (l) (+16.7%)	0.2362 (p) (+11.3%) 0.2715 (l) (+27.9%)	0.2661 (p) (+25.3%) 0.2869 (l) (+35.1%)
15 docs	0.1672	0.1944 (p) (+16.3%) 0.1974 (l) (+18.1%)	0.1959 (p) (+17.2%) 0.2174 (l) (+30.0%)	0.2159 (p) (+29.1%) 0.2221 (l) (+32.8%)
20 docs	0.1473	0.1600 (p) (+8.6%) 0.1639 (l) (+11.3%)	0.1677 (p) (+13.8%) 0.1815 (l) (+23.2%)	0.1758 (p) (+19.3%) 0.1781 (l) (+20.9%)
30 docs	0.1154	0.1200 (p) (+4.0%) 0.1215 (l) (+5.3%)	0.1274 (p) (+10.4%) 0.1374 (l) (+19.1%)	0.1297 (p) (+12.4%) 0.1408 (l) (+22.0%)

Table 2. Average of MAPs

	Result of **zettair**	Web Co-citation	Content-Based	Hybrid Approach
Average of MAPs	0.2631	0.2966 (p) (+12.7%) 0.3017 (l) (+14.7%)	0.2939 (p) (+11.7%) 0.3207 (l) (+21.9%)	0.3190 (p) (+21.2%) 0.3391 (l) (+29.9%)

zettair and the citation-based document-profile similarity (computed by the Web co-citation method). The fourth column corresponds to the re-ranking method using the original score of **zettair** and the content-based document-profile similarity (computed by the vector-space model using **zettair**). The fifth column corresponds to the hybrid re-ranking method using three scores: the original score of **zettair**, the citation-based document-profile similarity, and the content-based document-profile similarity. With each method, **p** means product combination (cf. formula 5) and **l** means linear combination (cf. formula 4). In the first method, the coefficients (used in linear combination) for the original score of zettair and citation-based document-profile similarity are respectively 0.5 and 0.5; in the second method, the coefficients for the original score of zettair and content-based document-profile similarity are respectively 0.25 and 0.75; in the hybrid method, the coefficients for the original score of zettair, the citation-based document-profile similarity and the content-based document-profile similarity are respectively 0.25, 0.20 and 0.55.

From these results, we can see that all three methods can bring good amelioration. The content-based method is better than citation-based method. However, the hybrid approach is the best among the three methods, it brings +35.1% improvement with precision at 10 documents and 29.9% improvement with the mean average precision measure. In these experiments, the linear combination is better than the product combination. Furthermore, the amelioration seems to be more clear with precisions at 5, 10 and 15 documents.

5 Conclusions and Future Work

In this paper, we presented some methods for personalized search in digital libraries. In our approaches, the user profile which represent the user's interests is a set of papers. The user's search results are re-ranked using similarity between them and the user profile. We did experiments on a collection of IEEE papers used in the INEX 20005 campaign to compare the performances of the citation-based method, the content-based method and the hybrid method. Experimental results showed that these methods are efficient and the hybrid method is the best method. Our work is close to the work of Bollacker et al with the CiteSeer system [10]; however we focus on information retrieval while they focus on information filtering.

One of the future directions is to combine the bibliographic coupling method [21] (another citation-based method) with these methods, which could lead to better performance. In the future, knowing that there are similar points between citations and hyperlinks, we intend to do similar experiments on a collection of Web pages to compare the performance of these methods in the hyperlinked environment.

Acknowledgement

This work is done in the context of the European Project CODESNET and supported by the Web Intelligence Project of the "Informatique, Signal, Logiciel Embarqué" cluster of the Rhône-Alpes region.

References

1. Kelly, D., Teevan, J.: Implicit feedback for inferring user preference: a bibliography. SIGIR Forum 37, 18–28 (2003)
2. Shen, X., Tan, B., Zhai, C.: Implicit user modeling for personalized search. In: CIKM 2005: Proceedings of the 14th ACM international conference on Information and knowledge management, pp. 824–831. ACM Press, New York (2005)
3. Chen, L., Sycara, K.: Webmate: a personal agent for browsing and searching. In: AGENTS 1998: Proceedings of the second international conference on Autonomous agents, pp. 132–139. ACM Press, New York (1998)
4. Speretta, M., Gauch, S.: Personalizing search based on user search histories. In: Thirteenth International Conference on Information and Knowledge Management (CIKM) (2004)
5. Seo, Y.-W., Zhang, B.-T.: A reinforcement learning agent for personalized information filtering. In: IUI 2000: Proceedings of the 5th international conference on Intelligent user interfaces, pp. 248–251. ACM Press, New York (2000)
6. Singh, A., Hierarchical, K.N.: classification of web search results using personalized ontologies. In: Proceedings of HCI International 2005, Las Vegas (2005)
7. Amato, G., Straccia, U.: User profile modeling and applications to digital libraries. In: Abiteboul, S., Vercoustre, A.-M. (eds.) ECDL 1999. LNCS, vol. 1696, pp. 184–197. Springer, Heidelberg (1999)

8. Rohini, U., Ambati, V.: A collaborative filtering based re-ranking strategy for search in digital libraries. In: ICADL, pp. 194–203 (2005)
9. Torres, R., et al.: Enhancing digital libraries with techlens+. In: JCDL 2004: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries, pp. 228–236. ACM Press, New York (2004)
10. Bollacker, K., Lawrence, S., Giles, C.L.: A system for automatic personalized tracking of scientific literature on the web. In: Digital Libraries 99 - The Fourth ACM Conference on Digital Libraries, pp. 105–113. ACM Press, New York (1999)
11. Small, H.G.: Co-citation in the scientific literature: A new measure of the relationship between two documents. Journal of American Society for Information Science 24, 265–269 (1973)
12. Huang, S., et al.: Tssp: A reinforcement algorithm to find related papers. In: WI 2004: Proceedings of the Web Intelligence, IEEE/WIC/ACM International Conference on (WI 2004), Washington, pp. 117–123. IEEE Computer Society, Los Alamitos (2004)
13. Couto, T., et al.: A comparative study of citations and links in document classification. In: JCDL 2006 (2006)
14. Vaughan, L., Shaw, D.: Bibliographic and web citations: what is the difference? J. Am. Soc. Inf. Sci. Technol. 54, 1313–1322 (2003)
15. Kousha1, K., Thelwall, M.: Motivations for url citations to open access library and information science articles. Scientometrics 68, 501–517 (2006)
16. Prime-Claverie, C., Beigbeder, M., Lafouge, T.: Transposition of the cocitation method with a view to classifying web pages. J. Am. Soc. Inf. Sci. Technol. 55, 1282–1289 (2004)
17. Pehcevski, J., Thom, J.A., Tahaghoghi, S.M.M.: RMIT university at INEX 2005: Ad hoc track. In: Fuhr, N., et al. (eds.) INEX 2005. LNCS, vol. 3977, Springer, Heidelberg (2006)
18. Malik, S., et al.: Overview of INEX 2005. In: Fuhr, N., et al. (eds.) INEX 2005. LNCS, vol. 3977, pp. 1–15. Springer, Heidelberg (2006)
19. Hull, D.A.: The trec-7 filtering track: description and analysis. In: Voorhees, E.M., Harman, D.K. (eds.) Proceedings of TREC-7, 7th Text Retrieval Conference, National Institute of Standards and Technology, Gaithersburg, US, pp. 33–56 (1998)
20. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: IJCAI, pp. 1137–1145 (1995)
21. Kessler, M.M.: Bibliographic coupling between scientific papers. American Documentation 14, 10–25 (1963)

Alignment-Based Expansion of Textual Database Fields

Piroska Lendvai

ILK / Communication and Information Sciences
Tilburg University
P.O. Box 90153, 5000 LE, Tilburg
The Netherlands

Abstract. Our study describes the induction of a secondary metadata layer from textual databases in the cultural heritage domain. Metadata concept candidates are detected and extracted from complex fields of a database so that content can be linked to new, finer-grained labels. Candidate labels are mined drawing on the output of Alignment-Based Learning, an unsupervised grammatical inference algorithm, by identifying head - modifier dependency relations in the constituent hypothesis space. The extracted metadata explicitly represent hidden semantic properties, derived from syntactic properties. Candidates validated by a domain expert constitute a seed list for acquiring a partial ontology.

1 Introduction

The data model underlying the structure of a database is often manually constructed, with the risk of becoming out-of-date over time: records that are arranged according to this structure outgrow it as the number of data attributes increases when resources of various formats get merged, updated, and new concepts emerge. These tendencies sometimes result in a mix of attributes joined in an ad-hoc way in loosely defined (free text) columns of the database, typically labelled as *Special remarks*. Such columns are of lower semantic coherence, and are suboptimal for effective database querying as they may contain several identical data types, for example numbers, that implicitly describe different, perhaps idiosyncratic properties, of a record.

Consider the following example from the SPECIALREMARKS column of a museum collection database:

```
Slides MSH 1975-xviii-27/29, 1975-xix-20/25; tape recording 1975 II
B 297-304. Acquired as gift from the British Museum (Nat. Hist.),
BMNH 1975. 1348
```

If a researcher is searching this column for tape recordings of a certain year, he needs to browse through all slide identification numbers and other ID numbers as well, because retrieving numbers cannot be securely narrowed down any further than to accessing the entire field. A query would be more efficient if the various ID numbers of slides, tape recordings, registration numbers, etc. would be separately accessible, for example by assigning corresponding labels to these. We call the assignment of metadata labels to content (values) in complex fields *database field expansion*.

Search is only one procedure that benefits from database field expansion. It also facilitates the application of data mining tools to the database, since many hidden properties of the data are made explicit. The following text is taken from another field of the SPECIALREMARKS column of the same database:

```
dorsolateral folds to sacrum, no fold on flanks, dorsum with many
transverse bars. M.S.H. 15-05-1986
```

The contents of the field include descriptive features of the appearance of an animal, each feature taking its value. In fact, the database would become better organised and accessible if the separate properties in the description would become structured, for example in the form of paired *information types* and their *values*, as seen below.

```
body - dorsolateral folds to sacrum
flanks - no fold
dorsum - with many transverse bars
```

Such an explicit descriptive form is often used in relational databases. We can regard the unique information types (on the left) as secondary metadata, because these constitute an intermediary layer between primary metadata, i.e., the database column label, and the instantiation of domain concepts, i.e., the content of the field.

The above examples illustrate metadata structures in museum collections in the zoology domain, but would also occur in other (cultural heritage) domains. In the case of large and complex databases, it is a tedious, suboptimal, and perhaps infeasible job to manually define and construct the secondary metadata layer labels that qualify for field expansion. To this end, we propose using an unsupervised grammar inference framework, Alignment-Based Learning [2] to induce a grammar of multi-word textual database columns, and find feature – value pairs based on the grammar.

Grammar inference systems are language-independent and require no linguistic markup to induce structure from texts, which makes them especially suited for processing cultural heritage texts that often amalgamate several languages, not rarely in diachronically different variants. Most importantly, our observation is that despite the use of objective and concise text in such databases, these can be syntactically, and certainly semantically, rich. We argue that a grammatical inference approach can exploit the semantic-syntactic variation in free-text database columns for field expansion purposes. A grammar inference approach has not yet been applied to semantic end tasks, except for [1] who target named entity recognition in the biomedical domain, similarly motivated by the vast

Table 1. Datasets from four animal collection database fields

	specRA	bioRA	specCRU	bioCRU
# sents	2,641	694	665	781
# words / sent	11.8	6.5	8.5	4.7
# vocabulary	2,570	1,047	1,607	588

amount of unannotated resources that invite unsupervised methods for preprocessing data and presenting it to a human expert.

In the following section we describe our resources. Section 3 explains the way we utilised grammar induction. In Section 4 the experimental results are presented. We conclude the study by arguing that the proposed method is in line with the goals of [3] of extracting domain concepts and attributes, but works in an unsupervised way. This implies that the discovered metadata are constituents of a partial domain ontology.

2 Resources

Our data comes from different databases that describe collections of the Dutch National Museum of Natural History, Naturalis[1]. These databases vary in size, data model, and interface for entering and searching records; recently, an initiative has been set up to unify all resources in one database, for which automatic discovery of metadata structures would be beneficial. Free-text fields of the databases are typically labelled specialRemarks, biotope, locality, while fields specifying a single concept, entity, or measurement are genus, species, country, altitude, recorder, and the like.

We prepared four distinct datasets from two databases. The specRA dataset is drawn from the specialRemarks column of the Reptile and Amphibian database, the bioRA dataset from the biotope column of the same database. The specCRU and bioCRU datasets are gained from the corresponding columns of the database holding descriptions of Crustaceans. The full content of each field belonging to these columns is regarded as a sentence. The words in the sentences are tokenised. The datasets are filtered from duplicate sentences that occur as the animal specimens are often collected under similar circumstances, and can share many bodily features, and so on. In the specRA and specCRU sets all ocurrences of numbers are collapsed to the symbol NUM. Table 1 describes the datasets.

The sets differ in size and complexity: by far the most complex dataset is specRA, both in corpus size and sentence length. Special remarks can be made on any properties a specimen, thus the vocabulary and the length of sentences in this column is larger than even those from other columns. All sets contain a mix of languages: mainly Dutch and English, together with domain terminology in Latin, as well as some named entities, such as an animal's local name, in other languages.

[1] www.naturalis.nl

3 The Alignment-Based Approach

The Alignment-Based Learning algorithm (ABL) is an unsupervised, symbolic, structure bootstrapping system, described in [4]. ABL finds a grammar that underlies a corpus of plain text sentences, without using any external sources of information, thus its application is attractive when no linguistic annotation, vocabularies, and other resources, are available. Because its output is entirely data-driven, input to ABL predefines the properties of the generated grammar. This feature makes ABL especially suitable for our task at hand.

Since the datasets are extracted from single database columns, most of them contain grammatically elliptical sentences. This implies that ABL will not be able to output a full grammar (which is not necessary anyway for the field expansion task), but it will induce a grammar of elliptical, descriptive texts. This would be hard, if not impossible, to achieve by syntactic parsers trained on full sentences.

The first cycle of ABL aligns the sentences in the input corpus and creates of a pool of hypotheses, where unequal parts of the sentences are hypothetical constituents, clustered and judged interchangeable in their given context. The procedure can be likened to bracketing or chunking the words in each sentence, by comparing it to every other sentence in the corpus, using parametric string edit distance metrics. Based on the identified equal and unequal parts of sentences, a grammar is induced that is best visualised in terms of production rules from non-terminals to terminals.

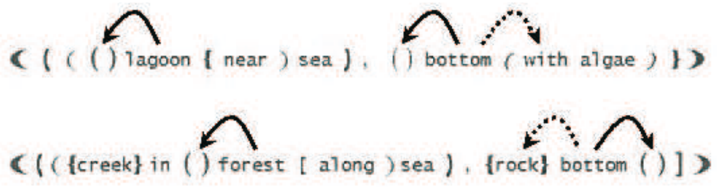
The subsequent step in ABL optimises the grammar by heuristically removing overlapping constituents, and selecting the hypotheses that are judged most probable by expectation maximisation. In the current study only the first cycle, alignment learning, is used (with default settings), in order to maximise the amount of field expansion candidates. This means that overlapping constituents are not removed from the hypothesis space, i.e., the complete fuzzy tree output of the alignment learning phase is being processed.

The extraction of field expansion candidates draws on the observation that INFOTYPEs are typically (phrasal) heads modified by their INFOVALUE. For example, the Birds database at Naturalis defines INFOTYPE labels such as SIZE, WEIGHT, FEET, ALTITUDE, EXPEDITION, PLUMAGE, with corresponding textual and numeric values.

In the constituent hypotheses of ABL we identify terminals that are able to take dependents in a modifier relation, but have also been seen in the corpus without a dependent, indicated by an *empty constituent* in the hypothesis space. Such terminals are regarded by our extraction process as field expansion candidates, whereas the constituent that is dependent on it, as the actual value. All INFOTYPE candidates are presented to a domain expert for evaluation, listed with the dependent value, and, if accepted, stored as secondary metadata.

3.1 Identifying Modifier Dependency Relations

Figure 1 illustrates the hypothesis space of ABL for two sentences, induced from a toy corpus of six sentences from the BIOCRU set, as listed in Table 2. In this



《 ((( ) lagoon (near) sea) , ( ) bottom (with algae)) 》
《 (({creek} in ( ) forest [along) sea) , {rock} bottom ( )] 》

Fig. 1. Bracketed output of consituent hypotheses from ABL. After empty constituents are identified in the context of metadata candidate 'bottom' (full arrow), the corresponding values 'rock' and 'with algae' are extracted (dashed arrows) from modifier dependents based on this evidence. Empty constituents are also identified in the context of potential candidates 'lagoon' and 'forest'.

figure, brackets indicate the hypothetical constituents. Note the empty brackets that signal a zero-length constituent. Empty constituents, i.e., indication of contextually possible but unrealised constituents, are found by ABL in the context of 'lagoon' and 'forest' (with left context values), as well as of 'bottom' (both with left and right context values). 'Bottom' has an empty constituent both on its left and on its right, and is counted as a head. In the extraction procedure, for all sentences with the word 'bottom', immediate left and right constituents are regarded as modifier dependants of this head, and are extracted as values of this field expansion candidate. By definition, the value may constitute both the immediate left and right context of the candidate, so that both endocentric and exocentric modifier relations (cf. [6]) are extracted.

In other words, the approach can be seen as a heuristic model for retrieving head-modifier relations and their directions on the phrasal level from ABL's hypothesis space. Note that 'head', 'dependency', and 'modifier' are our own terminology applied to the observed grammar strucure obtained from ABL. ABL itself does not provide any information on possible relations and does not name (non-)terminal types, but marks these by arbitrary symbolic labels.

Table 2. Toy corpus from the bioCRU set

```
lagoon near sea , bottom with algae
creek in forest along sea , rock bottom
pitfalls in swamp forest ; dead humid leaves
water pool
clear water , light muddy bottom
fresh water lagoon
```

In Table 3 we list the extracted head - modifier structures, i.e., infoType and infoValue pairs, from the toy corpus in Table 2.

In this case, 'lagoon', 'bottom', 'forest', 'water', and 'in' will be proposed to the domain expert as field expansion candidate. We see that the extraction process may miss some good candidates, those that are not considered constituents by ABL: e.g., `forest - along sea` is not extracted, since 'along sea' is not a

Table 3. infoType and infoValue pairs extracted from the toy corpus

`lagoon`	`fresh water`
`lagoon`	`near sea , bottom with algae`
`bottom`	`with algae`
`bottom`	`rock`
`bottom`	`light muddy`
`forest`	`swamp`
`water`	`clear`
`water`	`fresh`
`in`	`swamp`

constituent (cf. Figure 1). At the same time, semantically false positives (e.g., ‘in’) may also be retrieved; this may be alleviated by filtering on function words.

The extraction procedure cannot be likened to concordancing, because only those words are regarded candidates for which there is evidence of taking a modifier, and, most importantly, both the candidates and their potential values are retrieved at the phrasal (not the word) level from the corpus. The method can in fact be regarded as an unsupervised domain seed list selection procedure (e.g. for a bootstrapping approach to learn more candidates), where the seeds (i.e., heads of zero-length constituents) are indentified in a content-driven way, and not the usual frequency-based way.

Clearly, there are various levels of difficulty in extracting candidates from fields. Consider the examples given in the Introduction about expanding the specialRemarks field that includes descriptions of the appearance of a specimen. It is possible to retrieve the head in the phrase `no fold on flanks` (flanks, left modification) and for `dorsum with many transverse bars` (dorsum, right modification), but the infoType in the first part of the sentence (i.e., where exactly the dorsolateral fold occurs) needs to be induced by a different produre, since this head is not lexicalised in the text. Our study addresses these cases with explicit modifier relations only.

3.2 Preprocessing

Because the extraction procedure is data-driven, special attention needs to be paid to preprocessing a corpus. This includes removing noise from the data, such as typos and wrong-column errors. Depending on the language, tokenisation or stemming is desirable for optimal grammar induction results. Collapsing certain groups of words to a symbolic token (e.g. conversion of numbers to a common symbol, named entity recognition, etc.), the masking of hapax legomena, the removal of stop words, and other distribution-related issues need to be addressed to facilitate the extraction of higher-level metadata patterns. Automatic approaches to these preprocessing tasks are available in a number of NLP toolkits or can be achieved by rule-based methods or lookup lists.

In the following section we describe the magnitude of the hypothesis space, the automatically extracted field expansion candidates, and the expert-selected candidates on the four separate datasets from zoological databases.

Table 4. Metadata extraction results with ABL and heuristics on the four datasets

	specRA	bioRA	specCRU	bioCRU
# production rules	5,305	1,402	1,248	855
# non-terminals	62,574	10,533	9,971	7,539
# terminals	1,703	886	646	445
# candidates	650	198	149	128
# accepted candidates	37	25	15	24

4 Experimental Results

4.1 Quantitative Evaluation

Table 4 describes the results of processing the four datasets: the upper section in terms of the induced grammar by ABL, and the lower section the candidate extraction process. Comparing the amount of tokens in each datasets (cf. Table 1) and the number of proposed candidates, we characterise the magnitude of reduction in time needed for a human expert to validate the candidates. In particular, we argue that it is much more feasible to evaluate a few hundred type-value patterns than having to browse thousands of fields manually.

The mean ratio of accepted and proposed candidates (i.e., precision) is 11.1%. The average number of accepted new labels is 25, a magnitude which is in line with manually expanded and linked database fields of the same cultural heritage institution. For example, the Birds database contains around 40 infoType labels in the additionalInfo column. Moreover, these are semantically on the same granularity level as those extracted from our experimental datasets, i.e., are typical secondary layer metadata labels, such as iris, behaviour, ring, etc. (see also Section 2), but also describe numerical information such as label, regionID, recordDate, and the like.

4.2 Qualitative Evaluation

To illustrate the semantic range of the output of our approach, Table 5 displays a shortened list of the validated metadata candidates in each dataset. Some labels we translated from Dutch, some are the English original, such as 'loan' or 'formerly' in specRA, and all labels in bioCRU. It is interesting to observe that in the bioRA dataset several candidates are extracted both in their English form and in Dutch. Often, synonyms (e.g., 'formerly' vs. 'originally'), spelling variants, as well as semantically related word forms are extracted, such as both the nominalised and the inflected verb form (e.g., 'loan' vs. 'loaned'). In the final application, i.e., in the ontology, these terms (English or Dutch, respectively nominalised or inflected verb) are collapsed into a single concept, but for the markup procedure it is important that several syntactic or language variations of one and the same term are detected. The reported precision score is based on this collapsed concept group.

Table 5. Secondary metadata extracted from four animal collection database fields

specRA	bioRA	specCRU	bioCRU
born	bush	colour	algae
died	forest	drawing	beach
formerly	ground	female	bottom
length	m / cm	male	cave
loan	pool	NUM	clay
museum	river	photo	coral
NUM	road	pot	creek
photo	swamp	see	forest
slide	vegetation	size	water
tank	water	tube	zone

Evaluating the quality of the results can thus be rather challenging. A satisfaction score is difficult to create from the accepted term ratio, because the acceptance of terms might be biased by individual preferences. For example, a collection manager may disagree that certain candidates are helpful when directly searching the database, or may have preconceptions about how a database field structure should look like. Field expansion does not necessarily mean the physical modification of a database structure by adding more fields, rather, it is a method to induce additional layers of metadata, and linking the primary layer with the content of the fields through domain concepts of various granularity. It allows for the induction of domain concepts that, when integrated in an ontology, enable advanced searching and mining of the museum collection.

When comparing frequency lists from the corresponding columns, around 65% overlap can be found between the candidate list and the 200 most frequent tokens; however, heads and modifiers could not be separately retrieved based on just frequency lists. Separating head terms from modifiers is important in establishing relations between concepts in the ontology. For languages such as English, where noun-noun modification is a highly productive structure, our unsupervised procedure can serve a disambiguating role, as a word can often be both a head ('swamp forest') and a modifier ('forest floor') in a closed domain corpus.

Closely connected to this, we would like to discuss two advantageous features of the proposed alignment-based field expansion method. Next to being able to identify *directed* dependency relations (endocentric and exocentric), it also structures recursive modification. In particular, the acquired values of the infoTypes can themselves be embedded modifier dependencies, such as in

```
[ ((fresh) water) <= lagoon ]
[ ((light) muddy) <= water ]
```

Both the directedness of modification and the granularity of recursion are important information for the creation of a partial ontology from the validated terms and their values.

Besides, the approach allows for hierarchically linking metadata. For example, in the Birds database we observe that all INFOTYPE labels are marked as NUMBERS. By masking certain tokens in the data (e.g. not only numbers, but person names, geographical entities, etc.) it is straightforward in the current approach to mark the corresponding candidates with an additional, high-level metadata layer as well.

5 Conclusions and Future Work

We proposed inducing a new metadata layer from textual database fields. The goal of this study was to devise automatic means to preprocess cultural heritage resources, and present the results to a domain expert for validation. We extracted a secondary metadata layer with a heuristic method that draws on the output of alignment-based learning. No linguistic analysis is required for this process, making it attractive for cultural heritage materials that often feature several languages and distinct domain lexicons. Moreover, database fields typically contain grammatically elliptical text, which is difficult to process with state-of-the-art NLP tools.

The extracted field expansion candidate terms are to be evaluted by a domain expert. We show that the method effectively finds head-modifier dependencies in a corpus of texts from a database column, and provides tangible output in the form of elements of a secondary metadata layer. We also point out that by masking domain entities in the data is possible to link the expanded fields to another, high-level metadata layer. [3] describes an approach to extract – among others – concepts and attributes from linguistically annotated texts, whereas our approach achieves the same goal without the need for linguistic analysis and markup, only the contents of a database field are required.

An important aspect of our research is that it allows acquisition of readily utilisable semantic information with an unsupervised approach. The acquired set of terms can be used as a seed list in bootstrapped extraction of more candidates, both for field expansion and ontology population. Since a database column is semantically restricted, we conjecture that validated terms can directly be integrated in the corresponding part of a domain ontology.

An intriguing feature of the approach is that candidates extracted from BIORA and BIOCRU somewhat overlap (cf. Table 5), characterising the extent of similarity of biotopes between reptiles and amphibians and crustaceans. It will be interesting and straightforward to observe overlap of other aspects across (taxonomic) groups of specimens in terms of overlap in metadata.

A shortcoming of the currently implemented procedure is that it does not yet extract candidates that are always seen with a modifier in the data (i.e., always take a value), for example `green iris`. Nevertheless, it should be possible to extract such candidates by a follow-up bootstrapping loop (the focus of ongoing work), for example by drawing on modifier patterns generated by the seed list we identified with the current approach. Further straightforward extensions to the model include experimenting with more parametric variations of ABL, and transporting the approach to other languages, especially to ones of free word order, where modification might feature morpho-syntactically richer patterns in the context of empty constituents.

Acknowledgements

We thank Marian van der Meij and Ronald de Ruiter for being our domain experts at Naturalis, as well as Antal van den Bosch for commenting on the final version of the paper. This research was funded by the CATCH program of the Netherlands Organisation for Scientific Research (NWO).

References

1. Katrenko, S., Adriaans, P.: Grammatical Inference in Practice: A Case Study in the Biomedical Domain. In: Sakakibara, Y., et al. (eds.) ICGI 2006. LNCS (LNAI), vol. 4201, pp. 188–200. Springer, Heidelberg (2006)
2. van Zaanen, M., Geertzen, J.: ABL: Alignment-Based Learner, version 1.1, Reference Guide (2006), `http://www.ics.mq.edu.au/~menno/research/software/abl/`
3. Sintek, M., Olejnik, D., Buitelaar, P.: A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis. In: Bussler, C.J., et al. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 31–44. Springer, Heidelberg (2004)
4. van Zaanen, M.: Bootstrapping Structure into Language: Alignment-Based Learning. PhD thesis, School of Computing, University of Leeds, UK (2001)
5. van Zaanen, M.: Alignment-Based Learning versus Data-Oriented Parsing. In: Bod, R., Scha, R., Sima'an, K. (eds.) Data-Oriented Parsing, pp. 385–403. University Chicago Press (2003)
6. Nivre, J.: Dependency Grammar and Dependency Parsing. MSI Report 05133. Vaxjo University, School of Mathematics and System Engineering (2005)

Detecting Expected Answer Relations through Textual Entailment

Matteo Negri, Milen Kouylekov, and Bernardo Magnini

Fondazione Bruno Kessler
Via Sommarive, 18 - Povo, Trento, Italy
{negri,kouylekov,magnini}@fbk.eu

Abstract. This paper presents a novel approach to Question Answering over structured data, which is based on Textual Entailment recognition. The main idea is that the QA problem can be recast as an entailment problem, where the text (T) is the question and the hypothesis (H) is a relational pattern, which is associated to "instructions" for retrieving the answer to the question. In this framework, given a question Q and a set of answer patterns P, the basic operation is to select those patterns in P that are entailed by Q. We report on a number of experiments which show the great potentialities of the proposed approach.

1 Introduction

Question Answering (QA) over structured data has been traditionally addressed through a deep analysis of the question in order to reconstruct its logical form, which is then translated in the query language of the target data ([1], [2]). This approach implies a complex mapping between linguistic objects (*e.g.* lexical items, syntactic structures) and data objects (*e.g.* concepts and relations in a knowledge base). Several experiences, however, have shown that such a mapping requires intensive manual work, which represents a bottleneck in the realization of large scale and portable natural language interfaces to structured data.

More recently, Textual Entailment (TE) has been proposed as a unifying framework for applied semantics ([3]), where the need for an explicit representation of a mapping between linguistic objects and data objects can be, at least partially, bypassed through the definition of semantic inferences at the textual level. In this framework, a text (T) is said to entail a hypothesis (H) if the meaning of H can be derived from the meaning of T.

According to the TE framework, in this paper we propose that QA can be approached as an entailment problem, where the text (T) is the question, and the hypothesis (H) is a relational pattern, which is associated to instructions for retrieving the answer to the question. In this framework, given a question Q and a set of relational patterns P=$\{p_1, ..., p_n\}$, the basic operation is to select those patterns in P that are entailed by Q. Instructions associated to patters may be viewed as high precision procedures for answer extraction, which are dependent on the specific data source accessed for answer extraction. In case of QA over structured data, instructions would be SPARQL queries to a database; in case of QA on the Web an instruction would be the URL of a Web page containing the answer to a question. As an example, consider the question *"What movie is scheduled at cinema Astra tonight?"*, and a set of answer patterns including:

p1: <Movie:X> is on at <Cinema:Y>
p2: The ticket price at <Cinema:X> is <Price:Y>

An entailed-based QA system will first try to establish an entailment relation between the question (T) and each of the available patterns (H). Then, since in our example the question entails pattern *p1* and it does not entail *p2*, the system will output the database query associated to pattern *p1*, such as the SPARQL query:

```
SELECT ?MovieTitle
    WHERE { ?movie rdf:type myOntology:Movie .
            ?movie myOntology:name ?MovieTitle .
            ?cinema myOntology:name "Astra" .
            ?movie myOntology:isInSite ?cinema }
```

The benefit of the entailment-based approach is that semantic inferences do not require explicit mappings between linguistic expressions and data objects, such as:

schedule(subj:x, obj:y)
$\longrightarrow$ HasMovieSite(Movie:y, Cinema:x)
is_on(subj:x, obj:y)
$\longrightarrow$ HasMovieSite(Movie:y, Cinema:x)

Rather, inferences are performed at the textual level. For instance, the variation *"scheduled at"*, *"is on at"*, in the context of *cinema* and *movie*, could be captured by applying corpus statistics techniques (*e.g.* checking for the co-occurrence of the two expressions within similar contexts, to determine if they are replaceable). The main advantage of the proposed approach is that most of the machinery of compositional semantics used in traditional approaches to QA over structured data becomes unnecessary, resulting in an overall simplification of the task.

There are three relevant issues for the approach we propose, namely: *i)* how are relational patterns defined?, *ii)* how are they automatically acquired, and *iii)* how many relational patterns are necessary, and how this amount impacts on the performance of the entailment system? In this paper we address issues *i)* and *iii)*, leaving issue *ii)* as a next step in our research plan.

The paper is organized as follows. Section 2 introduces Minimal Relational Patterns, the basic textual structure we use. Section 3 provides a definition of the QA task in terms of a classification problem. Section 4 describes a number of experiments on entailment-based QA, whose results are finally reported and discussed in Section 5. Section 6 compares our approach with related works.

2 Relational Patterns

We use relational patterns as an approximation of answer patterns, as it seems that, at least in current factual-based QA, the core of the process can be seen as a search of relations among entities. We focus on binary relations, although extensions to n-ary relations are expected. We define a relational pattern as a portion of text that expresses a relation between two entities. For instance, all the examples (1)-(4) express, among others, a relation between a CINEMA and a MOVIE shown in that CINEMA.

(1) *"Cars" is on at cinema Astra.*
(2) *We went yesterday to Astra and watched "Cars".*
(3) *Do you know what movie is now on at Astra in Trento?*
(4) *Is there any nice movie that I can see at the cinema tonight?*

We say that a relational pattern P expresses a relation $R(arg1, arg2)$ in a certain language L if speakers of L agree that the meaning of P expresses the relation R between $arg1$ and $arg2$, given their knowledge about the entities (*e.g.* that "Cars" is a movie and "Astra" is the name of a cinema). In a relational pattern the arguments of the relation are explicitly marked:

(5) <ARG2:*"Cars"*> *is on at cinema* <ARG1:*Astra*>.
(6) *We went yesterday to* <ARG1:*Astra*> *and watched* <ARG2:*"Cars"*>.
(7) *Do you know what* <ARG2:*movie*> *is now on at* <ARG1:*Astra*>*in Trento?*
(8) *Is there any nice* <ARG2:*movie*> *that I can see at the* <ARG1:*cinema*> *tonight*?

In most of the cases arguments are lexicalized by a noun phrase, whose head represents the actual extension of the argument. Usually, in fact, head modifiers are likely to bring a different semantic relation (*e.g.* in pattern (8) *"nice movie"*). Arguments of the relation can either be both instantiated as individual entities (*e.g.* patterns (5)-(6)), both instantiated as generic entities (*e.g.* pattern (8)), or instantiated as individual and generic entities (*e.g.* pattern (7)). In addition, relational textual patterns can be either in affirmative form, as (5) and (6), or in interrogative form, as (7) and (8).

2.1 Minimal Relational Patterns

Given our entailment framework we are interested in those relational patterns that express the relation R using a minimum amount of textual material. For instance, supposing R is a relation between a CINEMA ($arg1$) and a MOVIE ($arg2$) that is shown in that cinema, the relational pattern (1) above is a minimal pattern, while (3) is not, since it provides additional constraints (*i.e.* the

Astra is *"in Trento"*, the movie is on *"now"*) which are not essential to express the relation R (a speaker would still judge (8) as a pattern for R). The patterns below (9)-(12) are the Minimal Relational Patterns (MRPs) of examples (1)-(4).

(9) <ARG2:*"Cars"*> *is on at cinema* <ARG1:*Astra*>.
(10) *We went to* <ARG1:*Astra*> *and watched* <ARG2: *"Cars"*>.
(11) *what* <ARG2:*movie*> *is on at* <ARG1:*Astra*>?
(12) *Is there any* <ARG2:*movie*> *that I can see at the* <ARG1:*cinema*>?

A more formal definition of MRP is based on TE. Given a set $P=\{p_1, p_n\}$ of relational patterns for a relation R, a pattern p_k belonging to P is a MRP for the relation R if condition (1) holds.

$$\forall p_i \in P, p_k \mapsto p_i = \emptyset \tag{1}$$

In other words, a pattern p_k is minimal if none of the other relational patterns contained in P can be derived from p_k (*i.e.* is logically entailed by p_k).

2.2 Typed Minimal Relational Patterns

For a number of practical purposes MRPs can be profitable generalized in order to cover classes of textual MRPs. A typed minimal relational pattern (TMRP) is a MRP where at least one argument has been typed, and substituted by a variable. The patterns below (13)-(16) are the TMRPs for examples (1)-(6).

(13) <ARG2:MOVIE:X> *is on at cinema* <ARG1:CINEMA:Y>.
(14) *We went to* <ARG1:CINEMA:Y> *and watched* <ARG2:MOVIE:X>.
(15) *what* <ARG2:*movie*> *is on at* <ARG1:CINEMA:Y>*?*
(16) *Is there any* <ARG2:*movie*> *that I can see at the* <ARG1:*cinema*>*?*

3 Task Definition

We have defined the entailment-based QA task as a classification problem where a question Q has to be assigned to all the relations R_1, ..., R_n it expresses. For instance, given the question *"What can I see today at cinema Astra"*, the following relations represent the expected system's output:

R1: HasMovieSite(Movie:x, Cinema:y)
R2: HasDate(Movie:x, Date:y)

Each relation is represented by at least one MRP (in some of the experiments described in the following Section 4.3 TMRPs will be used instead of MRPs), which is stored in a *Pattern Repository*. Given a question Q_i, the system attempts to verify whether an entailment relation holds between Q_i and each MRP in the repository. All relations for which an entailment relation is found are output by the system. In case no relation is found, this is interpreted as evidence that the question is out of domain.

4 Experiments on Entailment-Based QA

This Section describes a number of experiments we carried out, to evaluate the proposed entailment-based approach to QA under different configurations of the system.

4.1 Experimental Setting

The experimental setting includes:

Question Set. 1212 questions (in Italian), taken from the Italian part of the QALL-ME benchmark ([4]), which includes around 4,000 questions related to the domain of interest for the European Project QALL-ME (*i.e.* cultural events in a town)[1]. The available dataset has been divided into a *training set* (823 questions), and a *test set* (389 questions). The training set has been used for acquiring the minimal relational patterns; the test set has been used to evaluate system's performance. Each question has been manually annotated with the relations that are expressed in it. As an example, the question *"What can I see today at cinema Vittoria in Trento?"* has been annotated with the following three relations:

HasDate(Movie,Date)
HasMovieSite(Movie,Cinema)
IsInCity(Cinema,City)

Relation Set. 118 binary relations, defined in the QALL-ME ontology covering the cultural events domain. As an example, the relation HasDate(Movie,Date) represents a relation which has Movie as domain and Date as range. Possible lexicalizations of the relation are:

1. *"When can I see Pulp Fiction at cinema Vittoria?"*
2. *"What can I see today at cinema Astra?"*
3. *"How much is a ticket for Pulp Fiction next Saturday at Astra?"*

Minimal Relational Patterns (MRPs). For each relation we have manually created at least two MRPs, according to the definition given in Section 2.1. Patterns have been defined considering the questions training set, and they can be either in interrogative or affirmative form. Given a set of questions Q describing a relation R_i, the pattern creation guidelines we adopted are:

1. A valid pattern should describe only one relation.
2. A valid pattern must be entailed by each question in Q.

[1] The QALL-ME benchmark is being made incrementally available at the project Web site (http://qallme.itc.it). Note that, for the sake of clarity, this paper reports only English examples translated from Italian, the language we actually deal with.

As an example, given the relation HasDirector(MovieDirector), and the the following training examples:

Q1: *What's the title of the last action movie directed by Martin Campbell?*
Q2: *Is it scheduled for tomorrow Muccino's movie "La ricerca della felicitá?"*
Q3: *What's the name of the director of "Il grande capo" shown in these days?*

the extracted valid MRPs are:
p1: director of [Movie]
p2: movie directed by [Person]

Adopting the aforementioned criteria, we populated the pattern repository with a total of 249 patterns, with most of the relations associated to 2 patterns (94 out of 118), and four relations associated to at least 9 patterns.

4.2 Baseline

Our experimental baseline has been calculated considering: i) a Pattern Repository instantiated with *1 MRP per relation*, and ii) a simple entailment engine. Such basic entailment engine has been implemented within a distance-based framework, and is based on *Levenshtein Distance* (LD) or *Linear Distance* ([5]). The intuition is that, given a question Q and a pattern p, the probability of an entailment relation between Q and p is related to the possibility of mapping the whole content of Q into the content of p. The more straightforward the mapping can be established, the more probable is the entailment relation (see [6] for a thorough description of the overall approach).

The mapping between Q and p can be described as the sequence of edit operations needed to transform Q into p, where each edit operation has a cost associated with it. Edit operations are defined as follows:

Insertion $\Lambda \longmapsto A$: insert a word A from p into Q.
Deletion $A \longmapsto \Lambda$: delete a word A from Q.
Substitution $A \longmapsto B$: substitute a word A from Q with a word B from p.

We assign an entailment relation to a Q/p pair if the overall cost of the transformation is below a certain threshold, empirically estimated on the training data. The entailment score function is defined in the following way:

$$score_{entailment}(Q,p) = \frac{\gamma(Q,p)}{\gamma_{nomap}(Q,p)} \tag{2}$$

where $\gamma(Q,p)$ is the function that calculates the edit distance between Q and p and $\gamma_{nomap}(Q,p)$ is the *no mapping* distance equivalent to the cost of inserting the entire text of p, and deleting the entire text of Q. The entailment score function has a range from 0 (when Q is identical to p), to 1 (when Q is completely different from p).

The costs of the edit operations are defined in the following way:

$$\gamma(\Lambda \longmapsto A) = length(Q) \tag{3}$$

$$\gamma(A \longmapsto \Lambda) = length(p) \tag{4}$$

$$\gamma_{i+d}(A \longmapsto B) = \gamma(\Lambda \longmapsto A) + \gamma(A \longmapsto \Lambda) \tag{5}$$

$$\gamma(A \longmapsto B) = \begin{cases} 0 & A = B \\ \gamma_{i+d}(A \longmapsto B) & otherwise \end{cases} \tag{6}$$

where the costs of inserting or deleting a word in Q are respectively proportional to the length (*i.e.* the number of words) of Q and p, and $\gamma_{i+d}(A \longmapsto B)$ is the sum of the costs of inserting A and deleting B. Setting the insertion and deletion costs respectively to the length of Q and p is motivated by the fact that shorter Qs should not be preferred over longer ones while computing the overall mapping costs. The problem is evident in the following example:

Q1: *Could you please give me the telephone of Astra?*
Q2: *Where is Cinema Astra?*
p: *telephone number of Cinema Astra*

If we set the insertion and deletion costs to constant values (*e.g.* 1), mapping $Q1$ to p will result in a higher cost than mapping $Q2$ to p, even though $Q1$ is clearly closer to p than $Q2$. In fact:
$\gamma_{Fixed}(Q1, p) = 0.571$ (Deletions=6, Insertions=2, Nomap=14);
$\gamma_{Fixed}(Q2, p) = 0.555$ (Deletions=2, Insertions=3, Nomap=9).

If we set the insertion and deletion costs respectively to 9 ($length(Q)$) and 5 ($length(p)$), mapping $Q1$ to p will correctly result in a lower cost than mapping $Q2$ to p. In fact:
$\gamma_{Proportional}(Q1, p) = 0.533$ (Deletions=30, Insertions=18, Nomap=90);
$\gamma_{Proportional}(Q1, p) = 0.8$ (Deletions=10, Insertions=12, Nomap=40).

4.3 Experiments

The aim of the experiments was to estimate the impact on entailment-based QA of two different factors, namely: *i)* the number of MRPs stored in the Pattern Repository, and *ii)* different versions of the entailment engine. Such impact has been estimated in terms of Precision, Recall, and F-Measure improvements over the baseline described in the previous Section 4.2.

Experiment 1: Adding Patterns. The rational is that the more the patterns, the less the role of the entailment engine. We expect a performance increase with larger amounts of patterns. To check this hypothesis we experimented with a larger Pattern Repository. More specifically, considering all the 118 relations of interest, we evaluated the overall system's performance variations while considering *2 MRPs per relation* instead of only *1 MRP per relation* as we did for baseline calculation.

Experiment 2: Improving the Entailment Engine. The idea is that the deeper (*i.e.* semantically oriented) is the entailment engine, the higher should be the accuracy of the results. A more semantically oriented version of the algorithm has been implemented considering, as a first abstraction level, the information about Named Entities (NEs) that might appear both in Q and in p. This means, in practice, moving from comparisons between questions and MRPs (as done for baseline calculation and in Experiment 1), to comparisons between questions and TMRPs (see Section 2.2).

Considering NEs has the twofold objective of: i) reducing the distance between Q/p pairs containing concepts and/or expressions belonging to the same semantic category, and ii) enlarging the distance between Q/p pairs dealing with different types of concepts. For this purpose, while insertion and deletion costs are still set to $length(Q)$ and $length(p)$, the cost of the *substitution* operation defined in 6 has been adapted as follows:

$$\gamma(A \longmapsto B) = \begin{cases} 0 & A = B \vee A.NEcat = B.NEcat \\ \gamma_{i+d}(A \longmapsto B) & otherwise \end{cases} \tag{7}$$

The difference with respect to the basic Linear Distance algorithm is that the calculation of the substitution cost now considers the NE category of the two terms, instead of their simple string equality (to be replaceable, *A.NEcat* and *B.NEcat* should be the same).

In this experiment, the impact of improving the entailment engine has been estimated using a Pattern Repository of the same size as the one used for baseline calculation (*i.e.* containing *1 pattern per relation*), with MRPs transformed into TMRPs. NE recognition has been carried out with a ML-based tool for Italian ([7]), trained over Italian newspaper articles, capable of recognizing broad NE categories such as "Person", "Location", and "Organization".

Experiment 3: Adding Patterns and Improving the Engine. The two variations of the baseline system have been combined. In this experiment we checked the impact of dealing with the deeper entailment engine previously described, and a Pattern Repository instantiated with *2 TMRPs per relation.*

Experiment 4: Adding Even More Patterns. For a more precise plot of performance improvements while the number of patterns increases, a final experiment has been carried out over the 4 relations for which at least 9 patterns are available. In this case we experimented with 9 different Pattern Repositories, starting with one pattern per relation and adding one pattern at a time. In this case, both the entailment algorithms have been evaluated.

5 Results and Discussion

In each experiment we calculated Precision, Recall and F-Measure with respect to the relations selected by the system for each question. Given a question, a relation is correctly selected if the relation associated to the pattern belongs to the set of relations associated to the question in the gold standard. Precision is the proportion of relations correctly selected by the system, with respect to all the selected relations. Recall is the proportion of relations correctly selected by the system, with respect to all correct relations present in the gold standard. F-Measure is the harmonic mean of Precision and Recall.

Table 1 refers to Experiments 1, 2, and 3 (for the sake of comparison, results are summarized in one table). As can be seen, our working hypotheses are substantially validated.

Table 1. Results of Experiments 1, 3, and 4

Experiment	Algorithm	**♯Patterns**	**Precision**	**Recall**	**F-Measure**
Baseline	LD	1	0.805	0.142	0.241
Experiment 1	LD	2	0.784	0.187	0.302
Experiment 2	LD+NE	1	0.804	0.161	0.269
Experiment 3	LD+NE	2	0.770	0.197	0.314

As far as the **number of patterns** is concerned, we observe considerable F-Measure improvements over the baseline, with both versions of the entailment algorithm, when the number of patterns moves from 1 to 2 (+25.3% with LD; +16.7% with LD+NE). It's worth noting that: *i)* as expected such improvements depend on enhanced Recall performance (+31% with LD, +22% with LD+NE), and *ii)* the impact of such improvement on Precision is limited (-2.6% with LD, -4.2% with LD+NE). As far as the **entailment engine** is concerned, also in this case we observe F-Measure improvements, with both versions of the Pattern Repository (*i.e.* 1 pattern per relation, 2 patterns per relation), when moving from the simple LD algorithm to the deeper LD+NE algorithm (+ 11.6% with 1 TMRP per relation; +4% with 2 TMRPs per relation). If we consider that a general purpose Italian NE recognizer has been used both for creating TMRPs and processing the input questions, it is reasonable to expect significant performance boosts with a more accurate tool, adequately trained over the specific NE categories we are dealing with (*e.g.* "CINEMA", "TELEPHONENUMBER", "RESTAURANT"). As far as the **combined effect** of adding patterns and improving the engine is concerned (*i.e.* Experiment 3), we observe an overall +30.29% F-Measure improvement over the baseline, with a considerable Recall increase (+38.73%), and a limited Precision decrease (-4.34%).

Figures 1-2 refer to Experiment 4, where we tried to plot performance improvements while the number of patterns increases from 1 to 9 (this was possible only for 4 relations, for which at least 9 patterns have been created). Also these results confirm our hypotheses, showing general performance improvements both by increasing the number of patterns stored in the Repository, and by adopting an entailment engine based on deeper analysis techniques. As far as the number of pattern is concerned, however, we observe an unexpected performance drop when we add more

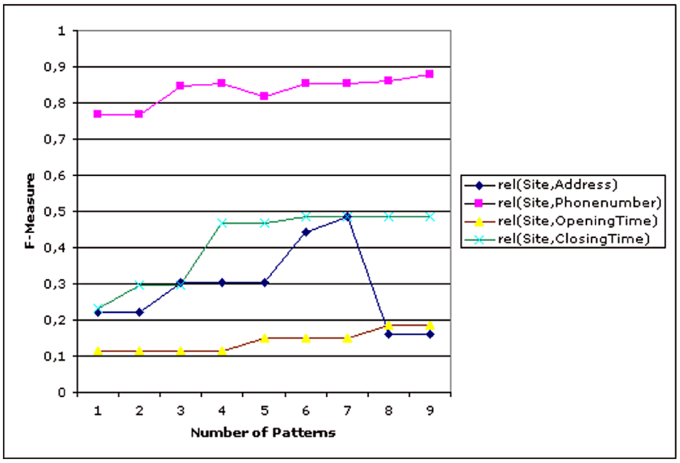


Fig. 1. F-Measure variations, using from 1 to 9 MRPs, with basic LD

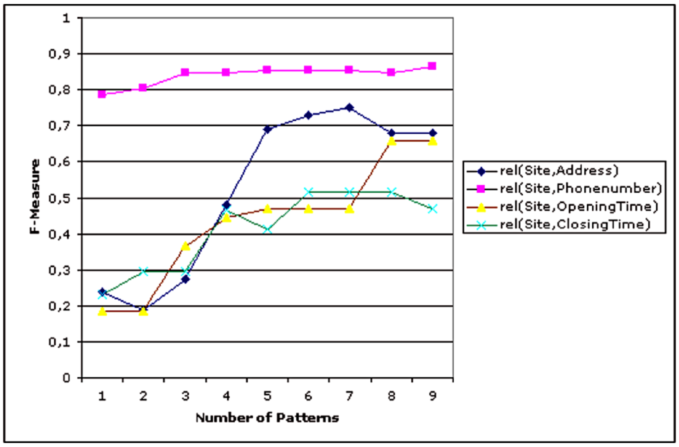


Fig. 2. F-Measure variations, using from 1 to 9 TMRPs, with the LD+NE algorithm

than 7 patterns for one of the relations (*i.e.* HasAddress(Site,Address)). Further investigations will explore the reasons of such F-Measure decrease.

6 Related Work

The growing interest towards applying entailment-based approaches to the QA task is demonstrated by several evaluation initiatives and recent works. Among these, the RTE Challenge ([8]), aims at evaluating TE recognition systems against a test set containing T/H pairs (small text snippets in English) derived from multiple NLP applications, including: QA, Information Retrieval (IR), Information Extraction (IE), and document summarization. Systems are required to decide, for each T/H pair, whether T entails H or not. In case of T/H pairs derived from the QA scenario, T is a text snippet containing a possible answer to a question Q, while H is the question Q turned into an affirmative sentence. This simulates the need, for a QA system, to verify that the retrieved passage text indeed entails the provided answer. A similar evaluation setting is proposed by the AVE multilingual task organized within the CLEF evaluation campaign ([9]), which is explicitly defined as a QA task (answer validation) based on TE recognition. Participating systems, in fact, have to consider triplets (Question, Answer, Supporting Text) and decide whether the Answer to the Question is correct and supported (*i.e. entailed*), according to the given Supporting Text. The accuracy achieved by the performing systems, both in RTE and AVE (see for example [10] and [11]), demonstrates the great potentialities of applying TE recognition to the answer selection/validation process.

The same issues are addressed in [12], which explores different methods of using TE information either filter or rank answers returned by a QA system. Also in this case, experimental results show that TE can be used as a mechanism for approximating the types of inference needed to answer questions, boosting the accuracy of a baseline system up to 26%.

Even though they share a common entailment-oriented perspective on the QA problem, the reported works and evaluation settings substantially differ from the approach proposed in this paper. Such difference is in the way the problem is posed. On the one side, the mentioned TE applications are focused on the final stages of a traditional open domain QA process, and check for the existence of entailment relations between answer passages (T) and possible answer candidates (H). On the other side, our approach addresses the problems posed, at earlier stages of the process, by QA over structured data. Here, instead of *validating* or *ranking* answers, the actual issue relates to *query formulation*, and consists in capturing all the relations of interest for a certain domain, that might appear in a question.

7 Conclusions

In this paper we presented a novel approach to Question Answering over structured data, which is based on Textual Entailment recognition. The approach relies on checking for the existence of entailment relations between an input question Q and a set of Minimal Relational Patterns that describe possible relations of interest involved in the question, and are associated to "instructions" (*i.e* database queries) for answer extraction. The impact of *i)* increasing the number of patterns, and *ii)* using deeper (*i.e.* more semantically oriented) versions of the entailment recognition algorithm has been evaluated, resulting in significant performance improvements along both the dimensions.

References

1. Androutsopoulos, I., Ritchie, G., Thanisch, P.: Natural Language Interfaces to Databases – an Introduction. Journal of Natural Language Engineering 1 (1995)
2. Popescu, A., Etzioni, O., Kautz, H.: Towards a Theory of Natural Language Interfaces to Databases. In: Proceedings of the Conference on Intelligent User Interfaces, Miami, Florida (2003)
3. Dagan, I., Glickman, O.: Probabilistic Textual Entailment: Generic Applied Modeling of Language Variability. In: Proceedings of the PASCAL Workshop on Learning Methods for Text Understanding and Mining, Grenoble, France (2004)
4. Cabrio, E., et al.: Question Answering Based Annotation for a Corpus of Spoken Requests. In: Proceedings of the Workshop on Semantic Representation of Spoken Language (SRSL07), Salamanca, Spain (2007)
5. Levenshtein, V.: Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. Doklady Akademii Nauk SSSR 163 (1965)
6. Kouylekov, M.: Recognizing Textual Entailment with Tree Edit Distance: Application to Question Answering and Information Extraction. PhD thesis, International Graduate School in Information and Communication Technologies, Faculty of Science, University of Trento (2006)
7. Pianta, E., Zanoli, R.: Exploiting SVM for Italian Named Entity Recognition. In: Proceedings of EVALITA 2007, Rome, Italy (2007)
8. Bar-Haim, R., et al.: The Second PASCAL Recognising Textual Entailment Challenge. In: Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy (2006)
9. Peñas, A., et al.: Overview of the Answer Validation Exercise 2006. In: Peters, C., et al. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 257–264. Springer, Heidelberg (2007)
10. Hickl, A., et al.: Recognizing Textual Entailment with LCCs GROUNDHOG System. In: Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy (2006)
11. Tatu, M., Iles, B., Moldovan, D.: Automatic Answer Validation Using COGEX. In: Peters, C., et al. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 494–501. Springer, Heidelberg (2007)
12. Harabagiu, S., Hickl, A.: Methods for Using Textual Entailment in Open-Domain Question Answering. In: Proceedings of COLING/ACL 2006, Sydney, Australia, pp. 905–912 (2006)

Improving Question Answering by Combining Multiple Systems Via Answer Validation

Alberto Téllez-Valero[1], Manuel Montes-y-Gómez[1], Luis Villaseñor-Pineda[1], and Anselmo Peñas[2]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica
Grupo de Tecnologías del Lenguaje
Luis Enrrique Erro no. 1, Sta. María Tonantzintla, Pue.; 72840; Mexico
{albertotellezv,mmontesg,villasen}@inaoep.mx
[2] Universidad Nacional de Educación a Distancia
Depto. Lenguajes y Sistemas Informáticos
Juan del Rosal, 16; 28040 Madrid; Spain
anselmo@lsi.uned.es

Abstract. Nowadays there exist several kinds of question answering systems. According to recent evaluation results, most of these systems are complementary (i.e., each one is better than the others in answering some specific type of questions). This fact indicates that a pertinent combination of various systems may allow improving the best individual result. This paper focuses on this problem. It proposes using an answer validation method to handle this combination. The main advantage of this approach is that it does not rely on internal system's features nor depend on external answer's redundancies. Experimental results confirm the appropriateness of our proposal. They mainly show that it outperforms individual system's results as well as the precision obtained by a redundancy-based combination strategy.

1 Introduction

Question Answering (QA) systems are a kind of search engines that allow responding to questions written in unrestricted natural language. Different to traditional IR systems that focus on finding relevant documents for general user queries, this kind of systems are especially suited to resolve very specific information needs.

Currently, given the great number of its potential applications, QA has become a promising research field. As a result, several QA methods have been developed and different evaluation forums have emerged (such as those at TREC[1] and CLEF[2]). Latest results from these forums evidenced two important facts about the state of the art in QA. On the one hand, they indicated that it already does

[1] Text REtrieval Conference. http://trec.nist.gov/
[2] Cross Language Evaluation Forum. http://www.clef-campaign.org/

not exist any method capable of answering all types of questions with similar precision rates. On the other hand, they also revealed that most current QA systems are complementary. That is, each system tends to be better than the others in answering some specific type of questions. Just as an example, in the Spanish QA evaluation at CLEF 2005, the best individual QA system could only answer 42.5% of the questions, whereas the ideal combination of correct answers from all participating systems could achieved a precision of 73.5% [1]. Based on these two facts, a new problem has emerged, namely, how to automatically get the appropriate combination of answers from several QA systems.

This paper focuses on this new problem. It proposes using an answer validation method to handle a superficial combination of several QA systems. It is important to mention that answer validation was mainly conceived as a means to help individual QA systems to automatically detect its own errors [2]. In accordance with this idea, several QA systems have included an answer validation module that helps them in deciding whether a candidate answer should be accepted or rejected [3]. Our proposal goes a step forward demonstrating the usefulness of answer validation for combining several complementary QA systems. In other words, this paper shows the effectiveness of answer validation for leading an ensemble of QA systems.

The rest of the paper is organized as follows. Section 2 describes some related work on QA ensemble approaches. Section 3 presents our general proposal about using answer validation as integration mechanism for combining several QA systems. Section 4 gives some details on the answer validation method. Section 5 shows some evaluation results in Spanish QA. Finally, section 6 offers some conclusions and ideas for the future work.

2 Related Work

Ensemble methods are very popular in machine learning tasks. They are based on the idea of using multiple classifiers to solve a common problem [4]. The success of these methods has motivated the implementation of "ensemble" approaches for other tasks. In particular, in question answering, the objective of an ensemble method is to combine the capacities of several QA systems in order to increase the number of correct answers.

Ensemble methods for QA are of two main types: internal and external. In the internal ensembles the combination of systems occurs at component level. Traditionally a QA system has three main components: one for question analysis, one for passage retrieval, and another one for answer extraction. Therefore, this kind of ensembles distinguishes for applying more than one technique in some particular component. For instance, [5] describes a QA system that uses several passage retrieval methods, and [6] presents a system that applies two distinct strategies at each component.

On the other hand, external or superficial ensembles combine different QA systems at answer level, i.e., they directly combine the answers extracted by several systems and select one of them as final answer. In this case, it is possible to distinguish two different combination strategies. The first one is solely based on answer's redundancies, i.e., the ensemble selects as final answer the most frequent one [7]. The second one, in contrast, not only takes into account the answer's redundancies but also a confidence value associated with the capability of each system to answer each specific type of question [8]. It is also important to mention that there are some ensemble methods for multilingual QA [9]. These methods consider answers from different languages and select the final answer based on its monolingual ranking as well as on its multilingual redundancy.

3 Proposed Ensemble Architecture

Figure 1 shows the general scheme of our proposal for a QA ensemble. This ensemble uses an answer validation method to superficially combine several QA systems.

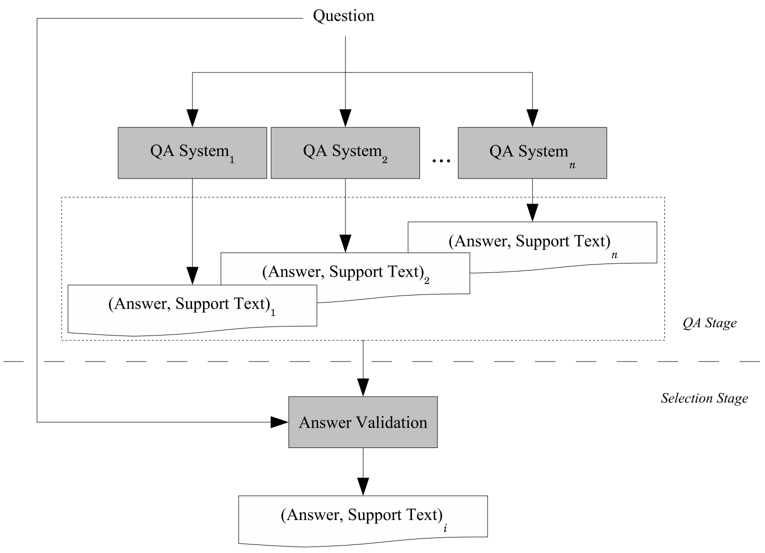

Question
QA System$_1$
QA System$_2$
...
QA System$_n$
(Answer, Support Text)$_n$
(Answer, Support Text)$_2$
(Answer, Support Text)$_1$
QA Stage
Selection Stage
Answer Validation
(Answer, Support Text)$_i$

Fig. 1. QA ensemble based on answer validation

Our QA ensemble consists of two main stages. In the first stage (called QA stage), several different QA systems extract – in parallel – a candidate answer (with its corresponding support text) for the given question. Then, in the second stage (called selection stage), an answer validation module evaluates – one by one – the candidate answers and selects as output the first accepted answer (for details on the validation process refer to section 4). In the case all candidate answers were rejected the output is set to NIL.

Given that the answer validation method is not perfect, the order of evaluation of the candidate answers is very relevant. Our current implementation considers a random order as well as a decreasing order based on the general confidence (accuracy) of the used QA systems.

The proposed ensemble distinguishes from previous approaches in three main concerns. First, it does not require to know (or adjust) internal details of the participating QA systems. Second, different to other previous external ensembles, it does not dependent on answer's redundancies. This is of crucial importance since there are many questions for which only one or very few QA systems could extract the correct answer. Third, in the case that there is not any correct answer, our approach could return a NIL answer, i.e., it is not obligated (as others) to always select one candidate answer. Finally, given the use of an answer-validation selection strategy, our ensemble not only returns correct but also supported answers.

4 Answer Validation Module

Given a question, a candidate answer and a support text, the answer validation module must decide whether to accept or reject the candidate answer. In other words, it must determine if the specified answer is correct and supported [2].

Our answer validation module is based on the idea of recognizing the textual entailment between the support text (T) and an affirmative sentence (called hypothesis, H) created from the combination of the question and the answer. The entailment between the pair (T, H) occurs when the meaning of H can be inferred from the meaning of T [10].

Figure 2 shows the general architecture of the answer validation module. As it can be seen, this module is based on a supervised learning approach and considers three main processes: hypothesis generation, feature extraction and entailment recognition.

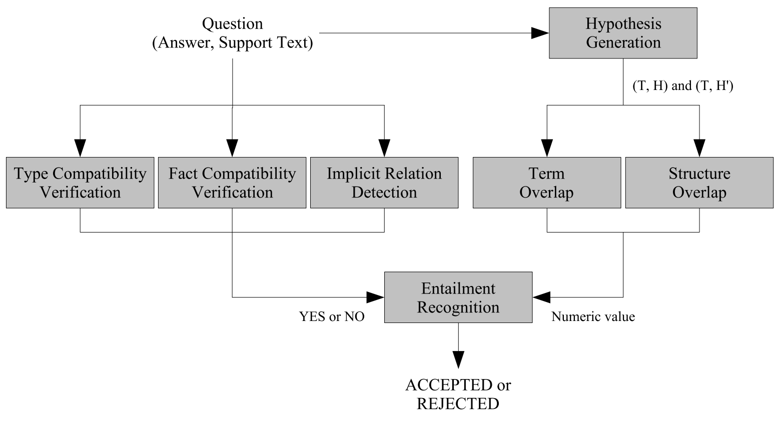


Fig. 2. General architecture of the answer validation module

4.1 Hypothesis Generation

The main task of this initial process is to construct two distinct hypotheses combining the given question and answer. In order to do that it firstly applies a superficial syntactic analysis over the question[3]. Then, using the obtained syntactic tree, it generates both hypotheses.

The first hypothesis (H) is constructed by replacing the nominal phrase that contains the interrogative particle by the given answer. For instance, given the question *"How many inhabitants are there in Longyearbyen?"* and the answer *"180 millions of inhabitants"*, this approach allows generating the hypothesis *H="180 millions of inhabitants are there in Longyearbyen"*.

The second hypothesis (*H'*) is obtained doing a simple transformation on *H*. The idea is to detect the main verb phrase of the H (that is the main verb phrase of the question) and then interchange its surrounding nominal phrases. This way the second hypothesis for our example is *H'="in Longyearbyen are there 180 millions of inhabitants"*.

4.2 Feature Extraction

We used two different kinds of features for the entailment recognition. On the one hand, some features that indicate the compatibility of question and answer. On the other hand, some classical textual entailment features that denote the level of similarity between the support text (T) and the generated hypotheses H or *H'*. The following subsections describe all these features.

Type Compatibility Verification. This process captures the situation where the semantic class of the evaluated answer does not correspond to the expected class of answer (in accordance with the given question). For instance, having the answer *"yesterday"* for the question *"How many inhabitants are there in Longyearbyen?"*.

In essence, this process calculates a Boolean value that indicates if the general-class restriction is satisfied. This restriction is TRUE if the semantic class of the candidate answer and the expected class of the answer are equal; in other case, it is set to FALSE.

In the current module's implementation, three general classes are considered: quantities, dates, and names. Moreover, the question classification (i.e., the definition of the expected class of the answer) is done using the KNN supervised algorithm with $K = 1$ and the answer classification is done by a name entity recognition method.

Fact Compatibility Verification. This process focuses on the situation where the question asks about a specific fact and the answer makes reference to another different fact. For instance, answering *"eight"* to the example question,

[3] The language analysis in the answer validation module is carried out with the open source tool called Freeling [11].

using as support text *"... when eight animals parade by the principal street in Longyearbyen, a town of a thousand of inhabitants"*.

With the aim of capturing this situation, this process determines a Boolean value that indicates if a specific-type restriction is satisfied. In order to determine the specific target fact concerning the question it is necessary to perform the following procedure: (*i*) construct the syntactic tree of the question, and (*ii*) extract the principal noun from the noun phrase that contains the interrogative particle. Applying this procedure over the example question, the word *"inhabitants"* was selected as the specific target fact.

Once extracted the specific target fact from the question, it is possible to evaluate the specific-type answer restriction. Its value is set to TRUE if the specific target fact happens in the support text, in the immediate answer context (one content word to the right or left). In any other case its value is set to FALSE. Therefore, the candidate answer *"eight"* has its value set to FALSE since its immediate context (*"eight animals"*) does not contains the noun *"inhabitants"*. On the contrary, the candidate answer *"thousand"* will be have its value set to TRUE, since the noun *"inhabitants"* occurs in its immediate context (*"town thousand inhabitants"*).

It is important to notice that not for all questions it is possible to establish a specific target fact (e.g., consider the question *"When was Amintore Fanfani born?"*). In these cases we considered – by default – that all candidate answers satisfied the specific-type restriction.

Implicit Relation Detection. Commonly, support texts present language phenomena such as apposition and adjectival phrases. This kind of phenomena makes implicit a relation between some elements (noun phrases) from the support text, and therefore, causes a detriment in the overlap between T and H. For instance, in the text *"the quinua, an American cereal of great nutritional value,"*, the verb *"is"* is implicit, it taking into account the hypotheses in the example shows in table 1.

In order to help the entailment recognition process to adequately treat these cases, we decide including a Boolean feature that simply indicates the existence of implicit information, i.e., the presence of some apposition or adjectival phrase.

The detection of this language phenomena is done by a set of some manually constructed lexical-syntactic text patterns such as "$\langle NOMINAL\_PHRASE \rangle$, $\langle NOMINAL\_PHRASE \rangle$,". In the case that some pattern (instantiated with the question and answer) matches the support text, then this Boolean feature is set to TRUE, in other case it is set to FALSE.

For instance, when the last text pattern is instantiated with the question *"What is the quinua?"* (only the question's target is used) and the candidate answer *"an American cereal of great nutritional value"*, the following text is obtained *"the quinua, an American cereal of great nutritional value,"*. This text matches the before mentioned support text. Because that, in the example presents in table 1, the feature that indicates the implicit relation detection is set to TRUE.

Table 1. Overlap analysis example (in the LCS the AJ, N and V are POS tags that indicates adjective, noun and verb, respectively)

Question	*"What is the quinua?"*
Answer	*"an American cereal of great nutritional value"*
Support text	*T="the quinua, an American cereal of great nutritional value,"*
Hypotheses	*H="an American cereal of great nutritional value is the quinua"*
	H'="the quinua is an American cereal of great nutritional value"
Term overlap	rate of nouns $= \frac{\|\{quinua, cereal, value\} \in H \cap T\|}{\|\{quinua, cereal, value\} \in H\|} = 1$
	rate of verbs $= \frac{\|\{\} \in H \cap T\|}{\|\{be\} \in H\|} = 0$ *(the verb is implicit in T)*
	rate of adjectives $= \frac{\|\{american, great, nutritional\} \in H \cap T\|}{\|\{american, great, nutritional\} \in H\|} = 1$
	rate of adverbs $= \frac{\|\{\} \in H \cap T\|}{\|\{\} \in H\|} = 1$
	rate of dates $= \frac{\|\{\} \in H \cap T\|}{\|\{\} \in H\|} = 1$
	rate of numbers $= \frac{\|\{\} \in H \cap T\|}{\|\{\} \in H\|} = 1$
LCS(*H*,*H*)	*american* AJ *cereal* N *great* AJ *nutritional* AJ *value* N *be* V *quinua* N
LCS(*T*,*H*)	*american* AJ *cereal* N *great* AJ *nutritional* AJ *value* N
LCS(*T*,*H'*)	*quinua* N *american* AJ *cereal* N *great* AJ *nutritional* AJ *value* N
Structure overlap	normalized size $= \frac{\|LCS(T,H')\|}{\|LCS(H,H)\|} = 0.86$

Term Overlap. This process calculates the term overlap between the support text and the hypothesis by a simple counting of the common words in the pair (*T*, *H*). In order to avoid a high matching caused by functional terms (such as prepositions and determiners), it only considers the occurrence of content terms (nouns, verbs, adjectives and adverbs). This analysis allows generating the following six features: (1) the rate of noun overlap, (2) the rate of verb overlap, (3) the rate of adjective overlap, (4) the rate of adverb overlap, (5) the rate of date overlap, and (6) the rate of number overlap. Table 1 shows an example.

Structure Overlap. This process measures the surface structure overlap between the support text and the hypotheses. Similar to the term overlap process, it also only considers content words, but in addition, it takes advantage of the POS tags.

In order to compute this overlap we extract the longest common subsequence (LCS) between the support text and the hypotheses. In this case, it is necessary to compute the LCS from (*T*, *H*) as well as from (*T*, *H'*). Nevertheless, only the longest subsequence is used. This way we generate the following feature from this analysis: the normalized size of the LCS between (*T*, *H*) or (*T*, *H'*). That is, size of the LCS divided by the size of the longest subsequence in *H*. Table 1 shows an example.

4.3 Entailment Recognition

This final process generates the answer validation decision by means of a supervised learning approach, in particular, by a Support Vector Machine Classifier.

This classifier decides whether to accept or reject the candidate answer based on the ten previously described features along with the following two additional ones: the question category (i.e., factoid or definition) and the question interrogative particle (i.e., who, where, when, etc.).

An evaluation of the proposed features during the development phase – using the information gain algorithm – shows us that the question category and the question interrogative particle are between the five most discriminative features, while the nouns overlap and the LCS size are the most discriminative.

5 Evaluation Results

In order to evaluate the proposed QA ensemble we used a set of 190 questions and the answers from 17 different QA systems[4]. In total, we considered 2286 candidate answers (with their corresponding support texts) for the evaluated questions. It is important to mention that this test set was employed at the first Spanish AVE (Answer Validation Exercise)[5], and that the system's responses were previously evaluated in the QA track at CLEF 2006 [2].

The main objective of our experiment was to demonstrate that our system ensemble could outperform each individual result. To evaluate the ensemble performance we used the accuracy measure. This measure is the most common evaluation metric for QA and indicates the percentage of correctly-answered questions[6] [13]. Table 2 shows the accuracy rates for each individual QA system. Internal columns show the system's accuracies for each type of question (F – factual, T – temporal restricted, D – definition).

Table 3 shows the accuracy results from different QA ensembles. The first three rows indicate some baseline results. In particular, the first row (ensemble 1) shows the results from an ideal ensemble, and the second and third lines (ensembles 2 and 3) shows the results achieved by two traditional ensembles. Finally, the last two rows (ensembles 4 and 5) indicate the results obtained by two variations of the proposed ensemble. The following paragraphs give a brief description and discussion on these ensembles.

Ensemble 1 is the ideal external ensemble. It indicates the maximum accuracy that can be reached by any external ensemble in the given test set. Its result is of great relevance since it confirms that current QA systems are complementary (it is possible to achieved 34% more accuracy than the best individual system).

Ensemble 2 is a confidence-based ensemble. Its output is the candidate answer extracted by the system having the greatest confidence value associated to the given type of question. Although this ensemble could outperform the best individual result by 3%, it has an important limitation: it does not take advantage of complementary systems for the same type of question, i.e., it does

[4] For the train phase we used the SPARTE corpus [12].

[5] We thanks to the AVE organizers for provide us the answer-run id relations.

[6] It is important to notice that in the accuracy evaluation an unanswered NIL question is considered as correctly answered.

Table 2. QA system's accuracies (to details about these systems refers to [14])

System	ID at CLEF 2006	% Right F	T	D	ALL
1	alia061enes	17.59	12.50	40.48	21.58
2	alia061eses	37.04	22.50	38.10	34.21
3	aliv061eses	29.63	22.50	35.71	29.47
4	aliv062eses	21.30	22.50	28.57	23.16
5	aske061enes	6.48	2.50	7.14	5.79
6	aske061eses	16.67	15.0	11.90	15.26
7	aske061fres	12.96	5.0	7.14	10.0
8	inao061eses	47.22	35.0	83.33	**52.63**
9	lcc_061enes	20.37	25.0	14.29	20.0
10	mira062eses	10.19	12.50	23.81	13.68
11	mira061eses	21.30	15.0	16.67	18.95
12	pribe061eses	52.78	27.50	69.05	51.05
13	pribe061ptes	24.07	25.0	16.67	22.63
14	sinaiBruja06eses	16.67	17.50	33.33	20.53
15	upv_061eses	37.04	25.0	47.62	36.84
16	upv_062eses	27.78	25.0	40.48	30.0
17	vein061eses	32.41	25.0	83.33	42.11

Table 3. Ensemble's accuracies

Ensemble	Description	% Right F	T	D	ALL
1	Ideal external ensemble	87.96	72.50	100.0	87.37
2	Based on systems confidence	52.78	35.0	83.33	55.79
3	Based on answers redundancy	51.85	27.50	52.38	46.84
4	Based on answer validation *(random)*	46.3	40.0	73.81	51.05
5	Based on answer validation *(ordered)*	51.85	42.50	85.71	**57.37**

not contemplate that two or more systems can be good enough for answering an specific type of question.

Ensemble 3 is a redundancy-based ensemble. Its output is the most frequent candidate answer (or NIL if there is not a most frequent answer). This kind of ensemble allows taking into account the responses of all QA systems, and thus, their whole complementarity. However, it produced a very poor result, obtaining 6% less accuracy than the best individual result. A detailed analysis of this result showed us that even though only 19 questions were responded by just one system, the redundancies of the correct answers were very low (mainly because the same answer can be written in different ways). We also noticed that, given the low precision of most QA systems, in many cases incorrect answers had high redundancies. An additional problem emerged at the time of assigning the support text (for a frequent answer may exist several different support texts, in this case the problem is to select the most pertinent one).

Ensemble 4 is an ensemble based on answer validation (refer to section 3). Its overall accuracy was below the best individual result. We attribute this behavior to the fact that the answer validation module has a high recall (73%) but a very low precision (52%)[7]. Therefore, the strategy of selecting as final response the first validated answer is not adequate, since this answer has great probability (48%) of being erroneous. However, it is important to point out that there is also a great probability of capturing the correct response in one of the subsequent accepted answers.

Ensemble 5 is an extension of Ensemble 4. It introduces a simple modification that allows avoiding the problems caused by the low precision of the answer validation module. Different from Ensemble 4 that evaluates the answers in a random order, this new ensemble takes answers in a decreasing order based on the general confidence (accuracy) of their source QA system. The result achieved by this ensemble was very significant. It outperformed the best individual result by almost 5% and was better that all previous ensemble results.

6 Conclusions

In this paper we proposed an external QA ensemble based on answer validation. Like other external ensembles, it does not rely on internal system's features. Nevertheless, it distinguishes from these ensembles in that: (*i*) it does not depend on the answer's redundancies, (*ii*) it is not obligated to always select one candidate answer, and (*iii*) it not only allows returning correct answers but also supported ones.

The evaluation results demonstrated the appropriateness of our proposal. Although the current validation module is still very imprecise, our QA ensemble (using an ordered set of candidate answers) could outperform the best individual result as well as the results from traditional ensemble approaches.

It is important to notice that an increment on the answer validation precision will directly impact on the ensemble accuracy. Based on this observation, our future work will be focused on improving this module. In particular, we plan to include other features for the entailment recognition such as the edit distance between the syntactic trees of T and H, and to calculate an accepted confidence value based on the most discriminative features used for the textual entailment recognition.

Acknowledgments

This work was done under partial support of CONACYT (project grant 43990 and scholarship 171610).

[7] That indicates that this module detects most correct answers but also validates several incorrect answers.

References

1. Vallin, A., et al.: Overview of the clef 2005 multilingual question answering track. In: Peters, C., et al. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 307–331. Springer, Heidelberg (2006)
2. Peñas, A., et al.: Overview of the answer validation exercise 2006. In: [14], pp. 257–264 (2006)
3. Magnini, B., et al.: Is it the right answer? exploiting web redundancy for answer validation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, July 6-12, 2002, pp. 425–432 (2002)
4. Dietterich, T.G.: Machine-learning research: Four current directions. The AI Magazine 18(4), 97–136 (1998)
5. Pizzato, L.A.S., Molla-Aliod, D.: Extracting exact answers using a meta question answering system. In: Proceedings of the Australasian Language Technology Workshop, Sydney, Australia, December 2005, pp. 105–112 (2005)
6. Chu-Carroll, J., et al.: In question answering, two heads are better than one. In: Proceedings of the HLTNAACL, pp. 24–31 (2003)
7. Rotaru, M., Litman, D.J.: Improving question answering for reading comprehension tests by combining multiple systems. In: Proceedings of the American Association for Artificial Intelligence (AAAI) 2005 Workshop on Question Answering in Restricted Domains, Pittsburgh, PA (2005)
8. Jijkoun, V., de Rijke, M.: Answer selection in a multi-stream open domain question answering system. In: McDonald, S., Tait, J.I. (eds.) ECIR 2004. LNCS, vol. 2997, pp. 99–111. Springer, Heidelberg (2004)
9. Aceves-Pérez, R.M., y Gómez, M.M., Pineda, L.V.: Graph-based answer fusion in multilingual question answering. In: Matoušek, V., Mautner, P. (eds.) TSD 2007. LNCS (LNAI), vol. 4629, pp. 621–629. Springer, Heidelberg (2007)
10. Dagan, I., Magnini, B., Glickman, O.: The pascal recognising textual entailment challenge. In: Proceedings of Pascal Challenge Workshop on Recognizing Textual Entailment, Southampton, UK, April 2005, pp. 1–8 (2005)
11. Carreras, X., et al.: Freeling: An open-source suite of language analyzers. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004), Lisbon, Portugal (2004)
12. Peñas, A., Rodrigo, Á., Verdejo, F.: SPARTE, a test suite for recognising textual entailment in spanish. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 275–286. Springer, Heidelberg (2006)
13. Magnini, B., et al.: Overview of the CLEF 2006 multilingual question answering track. In: Peters, C., et al. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 223–256. Springer, Heidelberg (2007)
14. Peters, C., et al. (eds.): CLEF 2006. LNCS, vol. 4730. Springer, Heidelberg (2007)

Evaluation of Internal Validity Measures in Short-Text Corpora[⋆]

Diego Ingaramo[1], David Pinto[2,3], Paolo Rosso[2], and Marcelo Errecalde[1]

[1] Development and Research Laboratory in Computacional Intelligence (LIDIC), UNSL, Argentina
[2] Natural Language Engineering Lab., Department of Information Systems and Computation, Polytechnic University of Valencia, Spain
[3] Faculty of Computer Science (FCC), BUAP, Mexico
{daingara,merreca}@unsl.edu.ar, {prosso,dpinto}@dsic.upv.es

Abstract. Short texts clustering is one of the most difficult tasks in natural language processing due to the low frequencies of the document terms. We are interested in analysing these kind of corpora in order to develop novel techniques that may be used to improve results obtained by classical clustering algorithms. In this paper we are presenting an evaluation of different internal clustering validity measures in order to determine the possible correlation between these measures and that of the F-Measure, a well-known external clustering measure used to calculate the performance of clustering algorithms. We have used several short-text corpora in the experiments carried out. The obtained correlation with a particular set of internal validity measures let us to conclude that some of them may be used to improve the performance of text clustering algorithms.

1 Introduction

Document clustering consists in the assignment of documents to unknown categories. This task is more difficult than supervised text categorization [13,8] because the information about categories and correctly categorized documents is not provided in advance. An important consequence of this lack of information is that clustering results cannot be evaluated with typical external measures like F-Measure and, therefore, the quality of the resulting groups is evaluated with respect to *structural properties* or *internal measures*. Classical internal measures used as cluster validity measures include the *Dunn* and *Davies-Bouldin* indexes, new graph-based measures like *Density Expected Measure* and Λ-Measure as well as some measures based on the corpus vocabulary overlapping.

⋆ This work has been partially supported by the MCyT TIN2006-15265-C06-04 project, as well as by the BUAP-701 PROMEP/103.5/05/1536 grant.

When clustering techniques are applied to collections containing *very short* documents, additional difficulties are introduced due to the low frequencies of the document terms. Research work on "short-text clustering" is relevant, particularly if we consider the current/future mode for people to use 'small-language', e.g. blogs, text-messaging, snippets, etc. Potential applications in different areas of natural language processing may include re-ranking of snippets in information retrieval, and automatic clustering of scientific texts available on the Web [10].

In order to obtain a better understanding of the complexity in clustering short-text corpora, a deeper analysis of the main factors that have a direct impact on the obtained results is required. Specifically, we are interested in studying whether the internal clustering validity measures are good estimators of the usability of the results from an user viewpoint. For this purpose, several short-text corpora are considered. Since the information about the correct categories of the documents is available, then the quality of clustering results evaluated according to the internal measures can be compared with external ones, in our case, with F-Measure.

Our study is closely related to the work presented in [15] where different internal cluster validity measures are used to predict the quality of clustering results in experiments with samples of the RCV1 Reuters collection [12]. The predicted quality in this case is compared with the real quality expressed by the F-measure values obtained from the classification of a human editor. In our case, we study *very short-text corpora*. The aim is to determine the correlation degree between internal and external clustering validity measures.

The rest of paper is organized as follows. In Section 2 we explain the metrics that are used in the experimental work to determine the quality of the obtained clusters. Section 3 describes the short-text corpora used in the experiments. The experimental results are shown in Section 4. Finally, we draw some conclusions and we discuss the future work.

2 Validity Measures

Cluster validity is a measure of goodness for the results obtained by clustering algorithms. There exist two types of cluster validy measures, namely, external and internal. The difference relies, respectively, on the use or not of a pre-specified structure of the data which is imposed usually by an expert. Following we describe a particular set of internal measures, some of them previously investigated in [15]. We introduce also two new measures based on the vocabulary overlapping whose basics were already presented in [10]. We start below with a short description of the well-known external validity F-Measure we used to calculate the correlation of the obtained results. Other internal validity measures (such as the Silhouette coefficient, correlation, cophenetic distance, purity, Neill's conditional entropy and Newman's Q-Measure) could have been explored. For instance, relative closeness and relative interconnectivity were introduced in [5] in the framework of dynamic modeling for hierarchical clustering. However, we consider that the analysis of all of them would be out of the scope of this paper.

2.1 *F*-Measure

In the context of clustering, F-Measure is an external validity measure that combines both, *precision* and *recall*. It may be formally defined as follows. Let D represent the set of documents, $\mathcal{C} = \{C_1, ..., C_k\}$ be a clustering of D and $\mathcal{C}^* = \{C_1^*, \ldots, C_l^*\}$ designate the human reference classification of D. The *recall* of a cluster j with respect to a class i, $rec(i,j)$ is defined as $|C_j \cap C_i^*|/|C_i^*|$. The *precision* of a cluster j with respect to a class i, $prec(i,j)$ is defined as $|C_j \cap C_i^*|/|C_j|$. Thus, the F-measure of the cluster j with respect to a class i is $F_{i,j} = \frac{2 \cdot prec(i,j) \cdot rec(i,j)}{prec(i,j)+rec(i,j)}$ and the overall F-measure is defined as:

$$F = \sum_{i=1}^{l} \frac{|C_i^*|}{|D|} \cdot \max_{j=1,..,k} \{F_{i,j}\} \tag{1}$$

2.2 The *Λ*-Measure

Let us consider a data collection as a weighted graph $G = \langle V, E, w \rangle$ with node set V (representing documents), edge set E (representing similarity between documents) and weight function $w : E \to [0, 1]$ (representing a similarity function between documents). Λ-Measure computes the *weighted partial connectivity* of $G = \langle V, E, w \rangle$. Formally [15], let $C = \{C_1, ...C_k\}$ be a clustering of the nodes V of a weighted graph $G = \langle V, E, w \rangle$, then the Λ internal measure of C is:

$$\Lambda(\mathcal{C}) = \sum_{i=1}^{k} \lambda_i \cdot |C_i| \tag{2}$$

where λ_i designates the weighted edge connectivity of $G(C_i)$. The weighted edge connectivity λ of a graph $G = \langle V, E, w \rangle$ is defined as $min \sum_{\{u,v\} \in E'} w(u, v)$ where $E^{'} \subset E$ and $G^{'} = \langle V, E \setminus E^{'} \rangle$ is not connected. It is expected that the higher is the value of Λ the better is the clustering obtained.

2.3 The Density Expected Measure

A graph G $= \langle V, E, w \rangle$ may be called sparse if $|E| = \mathcal{O}(|V|)$, whereas it is called dense if $|E| = \mathcal{O}(|V|^2)$. Then we can compute the density θ of a graph from the equation $|E| = |V|^\theta$ where $w(G) = |V| + \sum_{e \in E} w(e)$, in the following manner:

$$w(G) = |V|^\theta \Leftrightarrow \theta = \frac{\ln(w(G))}{\ln(|V|)} \tag{3}$$

θ can be used to compare the density of each induced subgraph $G^{'} = \langle V^{'}, E^{'}, w^{'} \rangle$ with respect to the density of the initial graph G. G' is sparse (dense) compared to G if $\frac{w(G^{'})}{|V'|^\theta}$ is smaller (bigger) than 1. Formally [15], let $\mathcal{C} = \{C_1, .., C_k\}$ be a clustering of a weighted graph $G = \langle V, E, w \rangle$ and $G_i = \langle V_i, E_i, w_i \rangle$ be the induced subgraph of G with respect to cluster C_i. Then the *Density Expected*

Measure (DEM) $\overline{\rho}$ of a clustering $\mathcal{C}$ is obtained as shown in Eq. (4). A high value of $\overline{\rho}$ should indicate a good clustering.

$$\overline{\rho}(\mathcal{C}) = \sum_{i=1}^{k} \frac{|V_i|}{|V|} \cdot \frac{w(G_i)}{|V_i|^{\theta}} \tag{4}$$

2.4 The Dunn Index Family

The Dunn Index Family identifies cluster sets that are compact and well separated. Let $C = C_1, ...C_k$ be a clustering of a set of objects D, $\delta : C \times C \rightarrow \mathbf{R}$ be a cluster to cluster distance and $\Delta : C \rightarrow \mathbf{R}$ be a cluster diameter measure. Then all measures of the following form are called Dunn indices.

$$I(C) = \frac{min_{i \neq j} \delta(C_i, C_j)}{max_{1 \leq l \leq k} \Delta(C_i)} \tag{5}$$

For our analysis we have used $\delta(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i, y \in C_j} d(x, y)$ and $\Delta(C_i) = 2 \left(\frac{\sum_{x \in C_i} d(x, c_i)}{|C_i|} \right)$ (see the Bezdek definition [3]), where $d : D \times D \rightarrow R$ is a function that measures distance between objects and c_i denotes the centroid of a cluster C_i. Large values of $I(C)$ correspond to a good cluster partition.

2.5 The Davies-Bouldin Index

This measure combines within-cluster scatter and between-cluster separation of a clustering C. It is obtained as follows.

$$DB(C) = \frac{1}{k} . \sum_{i=1}^{k} R_i(C), \text{ with} \tag{6}$$

$$R_i(C) = \max_{\substack{j=1....n \\ i \neq j}} R_{ij}(C) \text{ and } R_{ij}(C) = \frac{(s(C_i) + s(C_j))}{\delta(C_i, C_j)}$$

where $s : C \rightarrow \mathbf{R}$ measures the scatter within a cluster, and $\delta : C \times C \rightarrow \mathbf{R}$ is a cluster to cluster distance (intercluster). For our analysis we defined $s(C_i) = \frac{1}{|C_i|} \sum_{x \in C_i} ||x - c_i||$ and $\delta = ||c_i - c_j||$. Small values of DB would correspond to good clusters, since the clusters should be compact and their centers will be far away from each other.

2.6 The Relative Hardness Measure

This measure, introduced first in [10], calculates the vocabulary overlapping degree of a given set of clusters. Although this measure may be used both, as a external or a internal clustering validity measure, here we perform the evaluation of the data from an internal viewpoint. Formally, given a corpus C made up of n categories (CAT), the Relative Hardness (RH) of $C = \{CAT_1, CAT_2, ..., CAT_n\}$ is:

$$RH(C) = \frac{1}{n(n-1)/2} \times \sum_{i,j=1;i<j}^{n} Similarity(CAT_i, CAT_j), \tag{7}$$

where the similarity among categories is obtained by using both, the Jaccard coefficient and the cosine measure in order to determine their overlapping (see Equation (8) and (11), respectively).

$$Similarity(CAT_i, CAT_j) = \frac{|CAT_i \bigcap CAT_j|}{|CAT_i \bigcup CAT_j|} \tag{8}$$

In the above formula we have considered each category i as the "document" obtained by concatenating all the documents belonging to the category i. In Equation (9), w_{ij} is the weight of the term t_j in the category i (CAT_i). idf_j (Eq. (10)) is the inverse category frequency of the term t_j and, finally, the similarity (Eq. (11)) is the cosine of the angle between the categories vectorial representation of a given corpus. We have named the RH calculated with Eq. (8) as RH-J, whereas the one that uses Eq. (11) as RH-C.

$$w_{ij} = tf_{ij} \times idf_j \tag{9}$$

$$idf_j = log\left(\frac{n}{df_j}\right) \tag{10}$$

$$Similarity(CAT_i, CAT_j) = \frac{\sum_k w_{ik} \times w_{jk}}{\sqrt{\sum_k w_{ik}^2} \times \sqrt{\sum_k w_{jk}^2}} \tag{11}$$

3 Data Sets

The aim of this research work was to analyse the behaviour of different clustering internal measures over different short-text corpora of unrelated domains. The datasets used in the experiments carried out are described as follows.

3.1 The CICLing-2002 Corpus

This dataset is made up of 48 abstracts from the *Computational Linguistics* domain, which corresponds to articles presented at the *CICLing 2002* conference. Despite the small size, this collection has been used in differents experiments (see [7,11,2,9]). The distribution and the features of this corpus is shown in Table 1.

Table 1. Main characteristics of the *CICLing-2002* corpus

Category	# of abstracts
Linguistics	11
Ambiguity	15
Lexicon	11
Text Processing	11

Feature	Value
Size of the corpus (bytes)	23,971
Number of categories	4
Number of abstracts	48
Total number of terms	3,382
Vocabulary size (terms)	953
Term average per abstract	70.45

3.2 The R8 Dataset

Reuters-21578[1] has been extensively used in text categorization. In the experiments we have carried out, we have used the R8 subcollection of the Reuters-21578 since it is a single-categorized dataset. The characteristics of this corpus are given in Table 2.

Table 2. Main characteristics of the R8 corpus

Category	Test Docs	Train Docs
trade	102	319
grain	34	78
monex-fx	130	366
crude	140	314
interest	87	202
acq	707	1608
ship	43	121
earn	1076	2831

Feature	Test	Train
Size of the corpus (KBytes)	≈912	≈2,567
Number of categories	8	8
Number of documents	2,319	5,839
Total number of terms	150,430	416,431
Vocabulary size (terms)	9,315	15,648
Term average per document	64.87	71.32

Table 3. Sample of the 100 ambiguous words of the *WSI-SemEval* corpora with their corresponding number of instances

Word	Instances
share	3061
rate	1154
president	1056
people	869
state	689
point	619
part	552
system	520
bill	506
future	496
⋮	⋮

(a) nouns

Word	Instances
say	2702
ask	406
turn	402
feel	398
keep	340
go	305
work	273
do	268
believe	257
start	252
⋮	⋮

(b) verbs

[1] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

3.3 The WSI SemEval Corpora

This corpora was provided by the organizers of the *Evaluating Word Sense Induction (WSI) and Discrimination Systems* task of the SemEval 2007 workshop [1]. The dataset consists of 100 ambiguous words (65 verbs and 35 nouns) taken from the English lexical sample task of the same workshop. The corpora is then composed of 100 data collections, each one, corresponding to a specific ambiguous word. The name of a sample of the ambiguous words (10%) along with the number of their instances are presented in Table 3. A set of average values of the characteristics of this corpus is given in Table 4.

Table 4. Other features of the *WSI-SemEval* corpora

Feature	**Value**
Size of the corpus (bytes)	10,644,648
Number of ambiguous words	100
Number of sentences	27,132
Total number of terms	1,555,960
Vocabulary size (terms)	27,656
Average number of sentences (instances)	271.32
Average vocabulary size	47,65
Term average per sentence	57.34

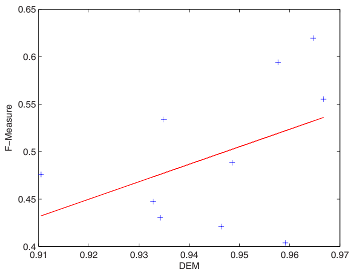
3.4 Subcorpora Generation

We have generated subsets for the *CICLing-2002* and the R8 corpora to analyse the behaviour of the Internal Clustering Validity Measures (ICVM) over all the differents variations of these corpora. We considered all the possible combinations of more than two categories from the corpus and for each of them we calculated its ICVM value. Therefore, for a corpus of n categories, a number of $2^n - (n+1)$ possible subcorpora is obtained.

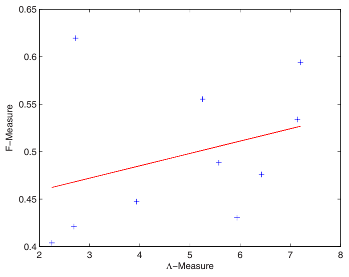
4 Experimental Results

The aim of this research work was to investigate the possible correlation between the external measure F and several internal clustering measures. We executed the K-Star agglomerative clustering method [14] over the corpora previously mentioned. The F-Measure and all the internal clustering validity measures were evaluated with the clusters obtained by this clustering method.

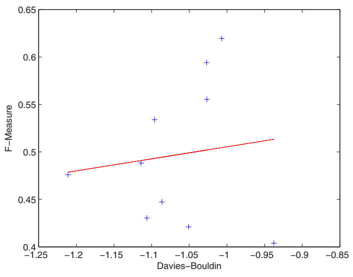
Figures 1, 2, 3 and 4 show the correlation results obtained for each corpus considered with the DEM, Λ-Measure, Davies-Bouldin and Dunn clustering validity measures. The x-axis corresponds to the different ICVM, whereas the y-axis corresponds to the F-Measure. In order to easily visualise the correlation between all the ICVM and F-Measure, we plotted the polynomial approximation of degree one. A desirable correlation would show points that start in the left corner



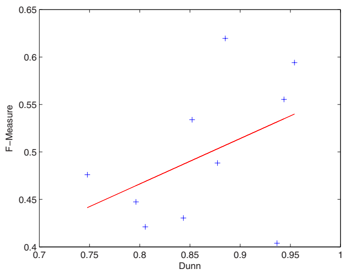
(a) DEM vs F-Measure



(b) *Λ*-Measure vs F-Measure

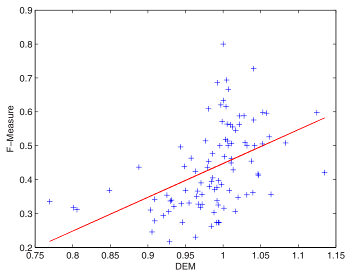


(c) Davies-Bouldin vs F-Measure

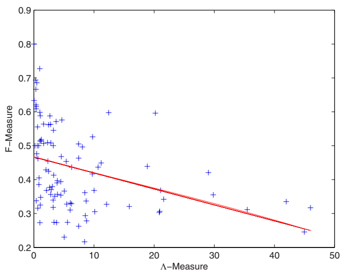


(d) Dunn Index vs F-Measure

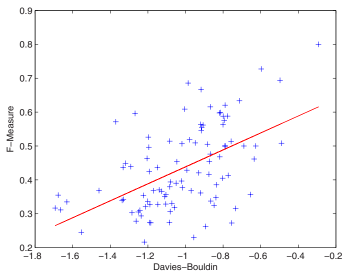
Fig. 1. Correlation of validity measures over the CICLing-2002 corpus



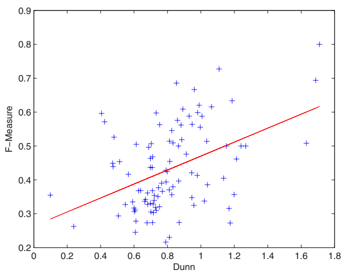
(a) DEM vs F-Measure



(b) *Λ*-Measure vs F-Measure

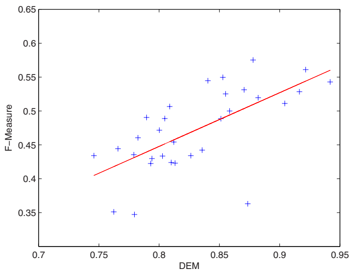


(c) Davies-Bouldin vs F-Measure

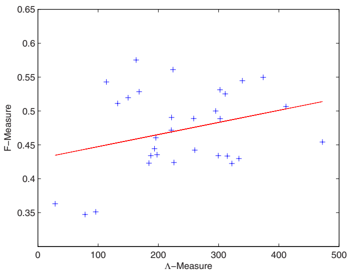


(d) Dunn Index vs F-Measure

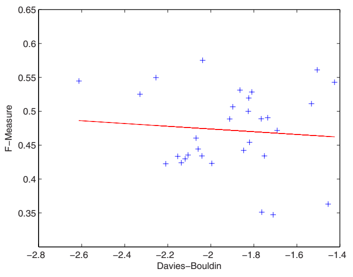
Fig. 2. Correlation of validity measures over the Semeval WSI corpus



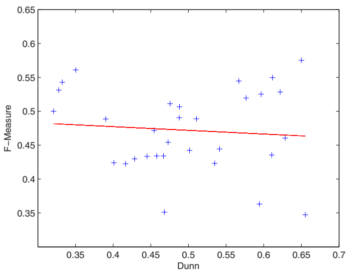
(a) DEM vs F-Measure



(b) Λ-Measure vs F-Measure

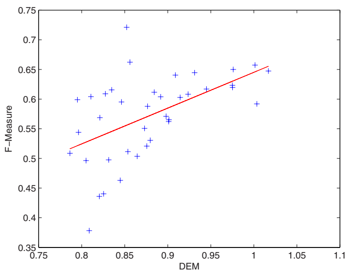


(c) Davies-Bouldin vs F-Measure

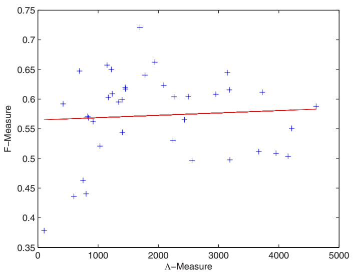


(d) Dunn Index vs F-Measure

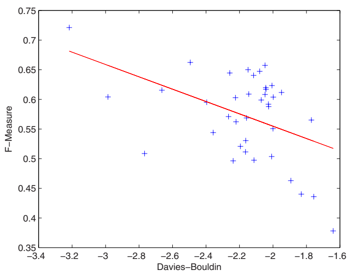
Fig. 3. Correlation of validity measures over the R8 test corpus



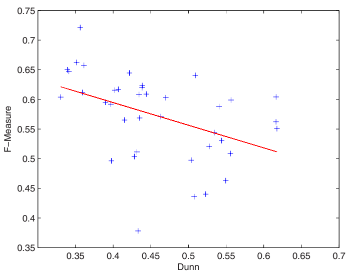
(a) DEM vs F-Measure



(b) Λ-Measure vs F-Measure

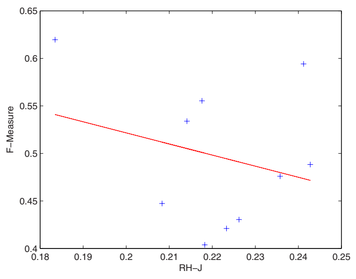


(c) Davies-Bouldin vs F-Measure

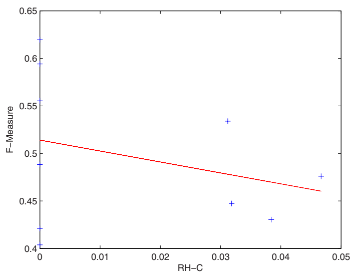


(d) Dunn Index vs F-Measure

Fig. 4. Correlation of validity measures over the R8 train corpus

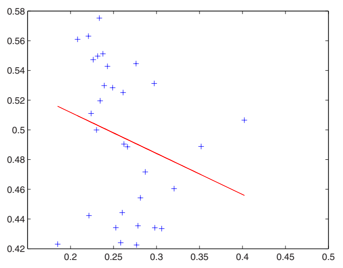


(a) RH-J

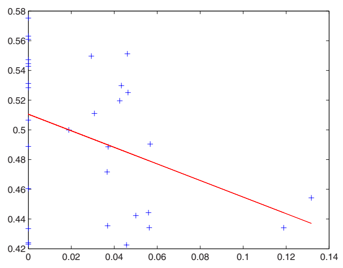


(b) RH-C

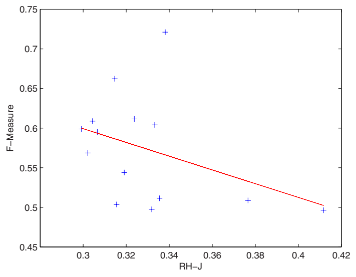
Fig. 5. Evaluation of the CICLing-2002 corpus with the RH formulae based on the Jaccard coefficient and the cosine measure



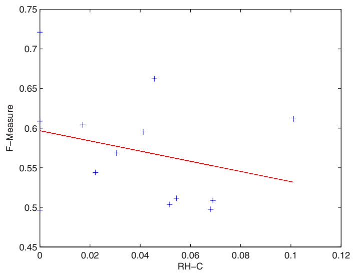
(a) RH-J with the R8 test corpus



(b) RH-C with the R8 test corpus



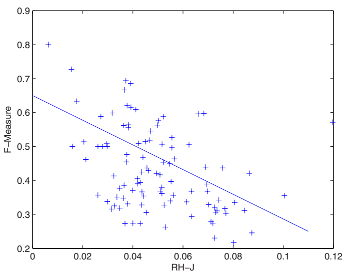
(a) RH-J with the R8 train corpus



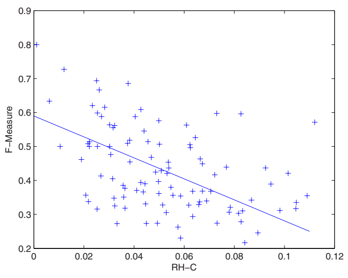
(b) RH-C with the R8 train corpus

Fig. 6. Evaluation of the R8 test and train corpora with the RH formulae based on the Jaccard coefficient and the cosine measure

(low values of F-Measure) and grows monotonically (high values of F-Measure). In this sense, for a better readability we changed the sign of the Davies-Bouldin index, which is the only measure to be minimised and in this way the results are directly comparable. This modification was not done in Figures 5, 6 and 7, where we present the obtained results of the two introduced internal clustering



(a) RH-J



(b) RH-C

Fig. 7. Evaluation of the Semeval WSI collection with the RH formulae based on the Jaccard coefficient and the cosine measure

validity measures (RH-J and RH-C), since we wanted to emphasize the specific behaviour of these new measures.

We observed that DEM is the only measure analysed that keeps the expected direct correlation in all the corpora. This behaviour suggests a certain robustness of this measure. Specifically, when it is evaluated with the WSI Semeval corpora, it appears to have a lineal correlation with the F-Measure.

The Λ-Measure obtains an "acceptable" correlation with the *CICLing-2002* and R8 corpora. However it is remarkable that the correlation obtained with the WSI SemEval corpus is inverse. We conclude that this ICVM is not adequate in general for short texts. In order to be more precise with the degree of "acceptability", as future work we aim to calculate some correlation index, such as the Spearman correlation coefficient which is a non-parametric (distribution-free) rank statistic which measures the strength of the associations between two variables [6]. One important finding is that if a clustering algorithm is designed in a way that attempts to optimise the Λ-Measure, then it will be negatively affected when using short-text corpora. The Davies-Bouldin index correlates very well with the F-Measure in the WSI SemEval collection, regular in the *CICLing-2002* corpus and quite bad in the R8 dataset. Finally, the Dunn measure behaves well with both, the *CICLing-2002* and WSI SemEval corpora, but it did not obtain a good correlation in the R8 dataset. We observed that the Davies-Bouldin and the Dunn indices have obtained similar results. With respect to the relative hardness, both, the one based on the Jaccard and the cosine similarity measures obtained good results in all the corpora.

From the corpora viewpoint, we may see that in the *CICLing-2002* corpus the ICVM measures obtain a good behaviour. In R8 all the results are consistent when evaluated in the test and train versions of this corpus; DEM, Λ-Measure and RH correlate very well with F-Measure, but Davies-Bouldin and Dunn obtain an inverse correlation. Future work should analyse the reason of these results. In WSI SemEval we obtained very good results for almost all ICVM (except Λ-Measure). The reader should pay attention that this collection consists of 100 corpora and, therefore, it makes sense to have obtained more stable results.

5 Conclusions and Future Work

The aim of this research work was to investigate whether the internal clustering validity measures may be used to improve the performance of clustering algorithms for short-text classification. In this paper we analysed different ICVMs with several short-text corpora.

Our findings indicate that the DEM and the RH measure are the ones that obtain the best results. However, it should be investigated whether the other ICVMs are related to specific kinds of corpora (for instance, narrow or wide domains). Thus, despite the corpora have very different characteristics it would be desirable to execute more experiments with other kind of domains, specifically to study, as we mentioned before, the narrow versus wide domain issue for short-text corpora. As further work we would also like to employ bio-inspired algorithms such as the DAntTree [4] to cluster short-text corpora. The main aim will be to investigate how to adapt the DAntTree algorithm to different internal clustering validity measures.

References

1. Agirre, E., Soroa, A.: Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In: Proc. of the SemEval Workshop, Prague, Czech Republic, The Association for Computational Linguistics, pp. 7–12 (2007)
2. Alexandrov, M., Gelbukh, A., Rosso, P.: An approach to clustering abstracts. In: Montoyo, A., Muńoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, pp. 8–13. Springer, Heidelberg (2005)
3. Bezdek, J.C., et al.: A geometric approach to cluster validity for normal mixtures. Soft Computing 1(4) (1997)
4. Ingaramo, D., Leguizamón, G., Errecalde, M.: Adaptive clustering with artificial ants. Journal of Computer Science & Technology 5(4), 264–271 (2005)
5. Karypis, G., Han, E.-H., Vipin, K.: Chameleon: Hierarchical clustering using dynamic modeling. Computer 32(8), 68–75 (1999)
6. Lehmann, E.L., D'Abrera, H.J.M.: Nonparametrics: Statistical Methods Based on Ranks. Prentice-Hall, Englewood Cliffs (1998)
7. Makagonov, P., Alexandrov, M., Gelbukh, A.: Clustering abstracts instead of full texts. In: Sojka, P., Kopeček, I., Pala, K. (eds.) TSD 2004. LNCS (LNAI), vol. 3206, pp. 129–135. Springer, Heidelberg (2004)
8. Montejo, A., Ureña, L.A.: Binary classifiers versus adaboost for labeling of digital documents. In: Procesamiento del Lenguaje Natural, pp. 319–326 (2006)
9. Pinto, D., Benedí, J.M., Rosso, P.: Clustering narrow-domain short texts by using the Kullback-Leibler distance. In: Gelbukh, A. (ed.) CICLing 2007. LNCS, vol. 4394, pp. 611–622. Springer, Heidelberg (2007)
10. Pinto, D., Rosso, P.: On the relative hardness of clustering corpora. In: Matoušek, V., Mautner, P. (eds.) TSD 2007. LNCS (LNAI), vol. 4629, pp. 155–161. Springer, Heidelberg (2007)
11. Pinto, D., Jiménez-Salazar, H., Rosso, P.: Clustering abstracts of scientific texts using the transition point technique. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 536–546. Springer, Heidelberg (2006)

12. Rose, T.G., Stevenson, M., Whitehead, M.: The Reuters Corpus volume 1 - from yesterday's news to tomorrow's language resources. In: Proc. of the 3rd International Conference on Language Resources and Evaluation - LREC 2002, pp. 827–832 (2002)
13. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
14. Shin, K., Han, S.Y.: Fast clustering algorithm for information organization. In: Gelbukh, A. (ed.) CICLing 2003. LNCS, vol. 2588, pp. 619–622. Springer, Heidelberg (2003)
15. Stein, B., Meyer, S., Wißbrock, F.: On cluster validity and the information need of users. In: Proceedings of the 3rd IASTED, pp. 216–221. ACTA Press (2003)

Arabic/English Multi-document Summarization with CLASSY—The Past and the Future

Judith D. Schlesinger[1], Dianne P. O'Leary[2], and John M. Conroy[1]

[1] IDA/Center for Computing Sciences, Bowie MD 20715, USA
{judith,conroy}@super.org
[2] University of Maryland, CS Dept. and Inst. for Advanced Computer Studies, College Park MD 20742, USA
oleary@cs.umd.edu

Abstract. Automatic document summarization has become increasingly important due to the quantity of written material generated worldwide. Generating good quality summaries enables users to cope with larger amounts of information.

English-document summarization is a difficult task. Yet it is not sufficient. Environmental, economic, and other global issues make it imperative for English speakers to understand how other countries and cultures perceive and react to important events.

CLASSY (Clustering, Linguistics, And Statistics for Summarization Yield) is an automatic, extract-generating, summarization system that uses linguistic trimming and statistical methods to generate generic or topic(/query)-driven summaries for single documents or clusters of documents. CLASSY has performed well in the Document Understanding Conference (DUC) evaluations and the Multi-lingual (Arabic/English) Summarization Evaluations (MSE).

We present a description of CLASSY. We follow this with experiments and results from the MSE evaluations and conclude with a discussion of on-going work to improve the quality of the summaries–both English-only and multi-lingual–that CLASSY generates.

1 Introduction

Automatic multi-document summarization poses interesting challenges to the Natural Language Processing (NLP) community. In addition to addressing single document summarization issues such as determining the relevant information, pronoun resolution, and coherency of the generated summary, multi-document summary-generating systems must be capable of drawing the "best" information from a set of documents.

Automatic single document text summarization [11] has long been a field of interest, beginning in the 1950s, with a recent renaissance of activity beginning in the 1990s. System generated single document summaries for English are generally of good quality. Therefore, NIST ended single document summarization evaluation after the 2002 Document Understanding Conference (DUC). See [17] for DUC research papers and results over the years.

In contrast to the single document task, summarization of multiple documents written in English remains an ongoing research effort. A wide range of strategies to analyze documents in a collection and then synthesize/condense information to produce a multi-document summary have been explored by various research groups. System performance has improved but still lags behind human performance.

Nevertheless, environmental, economic, and other global issues make it imperative for English speakers to understand how other countries and cultures perceive and react to important events. Thus it is vital that English speakers be able to access documents in a variety of languages.

The quantity of non-English documents makes it impossible to expect quality (or, even, *any*) human translation. Therefore, we have come to rely on machine translation (MT) systems for translation to English. While MT systems continue to improve, generated translations remain difficult to read and understand, with critical words often omitted, and inconsistent translations for the same word in a document [5,6]. Translation of Arabic documents is particularly challenging due to errors introduced by incorrect sentence-splitting, tokenization, and lemmatization.

Volumes of documents in one or more languages may be summarized by:

– creating summaries in the original language(s) which can then be translated by either humans or MT systems to determine "importance".
– creating summaries of the (MT-translated) documents which can be used to determine which documents are important and should be translated by humans.

CLASSY (Clustering, Linguistics, And Statistics for Summarization Yield) is an automatic summarization system, developed for summarizing English documents. CLASSY uses trimming rules to shorten sentences in the document, identifies sentences as being more or less likely to be included in a summary, generates a summary for each document, selects sentences for a multi-document summary for a cluster of related documents, and finally organizes the selected sentences for the final summary.

Our approach to multi-lingual summarization is based on the second approach listed above: we use CLASSY to generate single or multi-document (cluster) summaries of MT-translated documents. The experiments presented in Sect. 4. helped determine the best way to accomplish this.

We participated in the two Multilingual Summarization Evaluations (MSE)[1], which evaluated summaries of document sets containing a mix of both English and Arabic documents. Both the Arabic source and the MT output were

[1] MSE 2005: *Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization Workshop* at the Annual Meeting of the Association of Computational Linguistics (ACL 2005), Ann Arbor Michigan, 25-30 June 2005. MSE 2006: *Multilingual Summarization Evaluation* at the 21st International Conference on Computational Linguistics (ACL 2006)/44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, 17-21 July 2006.

available, and either or both could be used. This paper describes our use of CLASSY and its success in these evaluations.

The remainder of the paper is organized as follows. Section 2 contains a brief description of a sampling of other multi-lingual, multi-document summarization systems. Section 3 presents a description of CLASSY. Section 4 describes the experiments we ran and their success or failure. We then conclude with a discussion of current and future efforts to improve the generated summaries.

2 Related Work

There are many systems which summarize multi-lingual sets of documents, including languages such as Arabic, Chinese, Japanese, and Korean. We briefly describe four of these systems to indicate the breadth of work in this area.

Lakhas [5] is a summarization system that generates very short (headline) summaries. In contrast to many systems, Lakhas first summarizes the original Arabic documents and then applies MT to the summary only. While this eliminates the problems created by poor translations, it introduces its own myriad of difficulties related to Arabic sentence splitting, tokenization, and lemmatization. The scoring function is based on sentence position in the document, number of subject terms (i.e., words that appear in the headline) in the sentence, number of "indicative words" in the document (see the discussion of "signature terms" in Sect. 3.3), and the tf.idf value of each word in the sentence. This approach was very successful for very short (headline) summary generation (Task 3) in DUC 2004.

MEAD [15] is a platform for multi-document multi-lingual text summarization. It consists of multiple summarization algorithms including baselines (e.g., lead sentence) and both centroid-based and query-based methods. The MEAD architecture has four main components. First, each document is converted to an XML-based format. Then feature extraction is performed on each sentence of each document in a cluster, where the features are dependent on the selected summarization algorithm. Third, a composite score is calculated for each sentence. Finally, sentence scores may be refined based on considerations such as sentence repetition, sentence ordering, etc. MEAD currently supports Chinese and English summarization and can be extended to handle other languages.

The system described in [6] took an interesting approach. The DEMS summarizer [16] was first used to summarize a group of English and MT Arabic documents. DEMS produces summaries by extracting high-ranked sentences, where ranking is based on a set of features, some of which attempt to measure inherent importance of the thought. Text similarity measures [8] are then used to replace sentences chosen from the MT documents, which are generally ungrammatical and difficult to understand, with similar sentences from the English documents. This system performed quite well in DUC 2004, Task 3.

A multi-document, multi-lingual, theme-based summarization system based on modeling text cohesion (story flow) is presented in [7]. Some inherent text cohesion is specific to a particular story while some is specific to a particular language, and these differ across stories and across languages. To exploit the story flow, an unsupervised modified K-means method was used to iteratively cluster multiple documents into different topics (stories) and learn the parameters of parallel Hidden Markov Story Models (HMSM), one for each story. Story models were compared within and across stories and within and across languages (English and Chinese). Experimental results support "one story, one flow" and "one language, one flow" hypotheses.

Twenty-five teams participated in MSE 2005 while only eight did in MSE 2006. These teams were from both industry and academia, from various parts of the world. For example, the 8 teams from 2006 came from China, England, India, Japan, Tunisia, and the US. The 2005 teams were similarly distributed. A conflict with other conferences seems to be the major cause in the drop in participation. While this was unfortunate, the small number of participants did enable a comprehensive human evaluation. Reports about 4 of the 2006 systems (including CLASSY) are available on-line at http://research.microsoft.com/~lucyv/MSE2006_reports.htm. Reports from 2005 are no longer available.

3 Description of CLASSY

CLASSY architecture consists of five steps: document preparation, sentence trimming, sentence scoring, redundancy reduction, and sentence ordering, discussed in the following sections. These discussions are limited to English except where Arabic is explicitly mentioned.

3.1 Document Preparation

Every document is transformed to the CLASSY internal format by performing sentence splitting and sentence typing.

We currently use a Java-based sentence splitter, developed in-house and updated as needed. In addition, a post-processing phase that executes during tokenization (part of the sentence trimming task discussed in Sect. 3.2), corrects many of the sentence splitter's errors which result in either a single sentence erroneously being split into two or two sentences being run together. The main sources of sentence splitter errors are:

– foreign words, especially names that appear to be abbreviations of English words;
– less commonly used abbreviations not known to the sentence splitter;
– sentence termination punctuation embedded in parentheses or quotations;
– missing or bad punctuation; and
– ellipsis at sentence end.

Our sentence splitting is highly accurate, and the few errors that remain would require full parsing (which we do not perform) to detect.

After the initial sentence splitting step, all sentences are typed according to their potential usefulness in a summary. Sentences in headlines and other "title" roles are given a type of 0; this indicates that they may be useful for determining "signature terms" (see Sect. 3.3) but should not be selected for the summary. Sentences in the textual portion of a document are given a type of 1, indicating that they may be selected for a summary. All other text is given a type of -1: do not use. The sentence trimming algorithms (see below) may modify a sentence type from 1 to either type 0 or type -1, based on sentence length or content, i.e., boilerplate.

3.2 Trimming Sentences

Our trimming code has been written so that it does not require any part-of-speech (POS) tagging or parsing in order to perform its task. This decision was made based on the computational demands of both POS-taggers and parsers as well as the fact that, as good as they have become, both tasks still introduce errors. Instead, we make extensive use of word lists, along with the position of commas, periods, or the sentence start and end, to identify most of the phrases or clauses we remove.

Our sentence trimming approach has been documented in [3,1]. We continue to improve the algorithms to minimize the errors that are made, since these errors result in ungrammatical or, worse, erroneous sentences. We have also been able to identify a larger set of phrases and clauses to eliminate. The sentence trimming we perform is:

1. Removal of extraneous words that appear in a sentence, including date lines, editor's comments, etc.
2. Removal at the start of a sentence of many adverbs, all conjunctions, and about 2000 phrases such as "As a matter of fact," and "At this point."
3. Removal of a small selection of words that occur mid-sentence, such as "however" and "also".
4. Removal of age references such as ", 51," or ", aged 24,".
5. Removal of gerund phrases (phrases starting with the -ing form of a verb) from the start, middle, or end of a sentence.
6. Removal of relative clause attributives (clauses beginning with "who(m)", "which", "when", and "where") wherever possible.
7. Removal of attributions, such as "police said", at the start or end of sentences.

Additional trims, including removing many parenthesized or dashed (–) "asides", remain to be added. Figure 1 shows an example of each of the last three trims in the above list.

3.3 Scoring Sentences

We give a brief overview of an approximate Oracle score, which estimates the fraction of *human abstract terms* a sentence contains. Details of this approach and its motivation can be found in [4,2].

More than 800 lives were lost when the 21,794 tonne ferry**, sailing from the Estonian capital Tallinn to Stockholm,** sank within minutes early yesterday morning in the Baltic Sea 40 km south west of the Finnish island of Uto.

a. Example of a gerund phrase to be removed.

The Menendez family lived in the Princeton Area until 1986**, when they moved to California**.

b. Example of a restricted relative-clause appositive to be removed.

The federal Government's highway safety watchdog said Wednesday that the Ford Bronco II appears to be involved in more fatal roll-over accidents than other vehicles in its class and that it will seek to determine if the vehicle itself contributes to the accidents.

c. Example of an attribution to be removed.

Fig. 1. Examples of phrase/clause eliminations

Instead of using term frequencies of the corpus, as done by [12], to infer highly likely terms in human summaries, we directly model the *set* of terms (vocabulary) that is likely to occur in a sample of human summaries.

We model human variation in summary generation with a unigram language model. In particular, let $P(t|\tau)$ be the probability that a human will select term t in a summary given a topic τ. We define the *oracle score* for a sentence x to be

$$\omega(x) = \frac{1}{|x|} \sum_{t \in T} x(t) P(t|\tau)$$

where $|x|$ is the number of distinct terms that sentence x contains, T is the universal set of all terms used in the topic τ and $x(t) = 1$ if the sentence x contains the term t and 0 otherwise. This score depends on knowledge of human abstracts. Since this information is not available, we substitute a computable *approximate* oracle score [2].

In the absence of human abstracts, we view the *signature terms* as "samples" from idealized human summaries. A signature term is a term which occurs significantly more than expected in the document [9,2]). We use the Porter stemmer [14], which greatly improves the correlation of signature terms with human abstract terms. We define the signature term approximation to the oracle score for a sentence's expected number of human abstract terms as

$$\omega_s(x) = \frac{1}{|x|} \sum_{t \in T} x(t) P_s(t|\tau)$$

where $|x|$, T, and $x(t)$ are defined above, and $P_s(t|\tau) = 1$ if t is a signature term and 0 otherwise (a characteristic function).

The score is built upon an estimate of the probability that a term t will be included in a human summary given a topic τ. This probability is denoted $P(t|\tau)$.

It is approximated using the signature terms and the distribution of the terms in the relevant document cluster.

We estimate our target probability by a mixture of two distributions: the characteristic for the signature terms and the probability that a term occurs in the sentences to be considered for extraction:

$$P_{qs\rho}(t|\tau) = \frac{1}{2}s_t(\tau) + \frac{1}{2}\rho_t(\tau)$$

where $s_t(\tau)$=1 if t is a signature term for topic τ and 0 otherwise, and $\rho_t(\tau)$ is the maximum likelihood estimate of the probability that term t occurs in a sentence in the topic τ. Note that the mixture weights are balanced: both are set to 1/2. We found no statistical improvement in the performance of the approximate oracle score when other weights were used.

The correlation between the oracle score and the approximate oracle score is very strong. Figure 2 is a histogram of the Pearson correlation coefficients for 25 multi-lingual clusters from the MSE data sets.

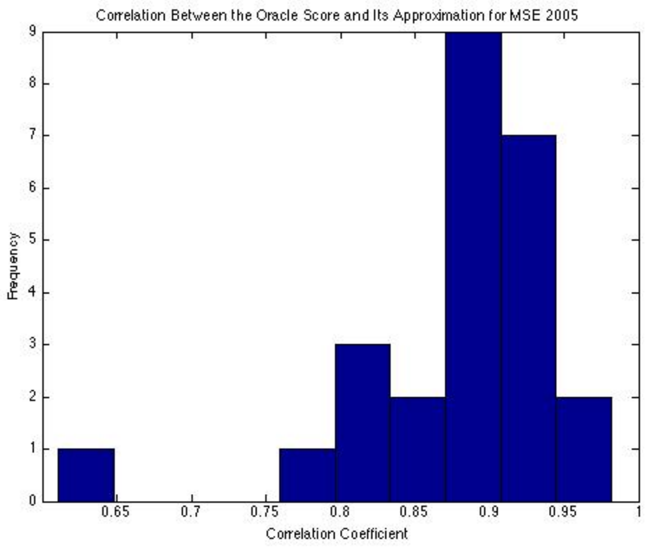


Fig. 2. Pearson correlation coefficients for 25 MSE multi-lingual clusters

3.4 Reducing Redundancy of the Selected Sentences

To reduce redundancy in the sentences chosen for inclusion in the summary, we have a three-step process.

1. Sentences are ordered by score, and enough sentences are chosen to produce a summary 9 times as long as desired. The length was chosen empirically, based on training on MSE 2005 data.

2. The approximate oracle score is simply the sum of the elements in the corresponding column of the (signature) term-sentence matrix A. To improve this score, we replace A by the rank-k matrix $\tilde{A}$ computed using the singular value decomposition. We choose $k = \max(1, \lfloor 0.65\, n \rfloor)$ where n is the number of sentences under consideration. This latent semantic indexing (LSI) improves the approximate oracle score, since it gives partial credit for closely related terms that are not literally in the sentence. This is an attempt to move from a term-based oracle to an idea-based one: to the extent that the sentences represent the *main ideas* of the document, LSI projects the sentences onto a subspace of these ideas. The column sums of $\tilde{A}$ can be then viewed as *refined* approximate oracle scores for the sentences.
3. Sentences are then chosen for inclusion using a pivoted-QR decomposition of the matrix $\tilde{A}$. The pivoted-QR decomposition proceeds as follows:
   (a) Begin with an empty summary.
   (b) As long as the summary length is shorter than desired, choose the largest remaining column and include its sentence in the summary.
   (c) Subtract a multiple of this column from each remaining column in order to account for duplicate coverage of terms.
   (d) Continue until the desired summary length is reached.

In the usual pivoted-QR decomposition, the size of a column is measured by its Euclidean norm; the norm of a vector q with entries q_i is computed as

$$\|q\| = \left(\sum_i |q_i|^2 \right)^{1/2} .$$

The multiples that are subtracted make the remaining columns orthogonal to the column chosen. In our latest system, we use a nonnegative-QR decomposition. We measure size using the 1-norm

$$\|q\| = \sum_i |q_i|,$$

and after subtracting off the multiple, we replace any negative entries in the matrix by zero to avoid having well-covered terms increase the length of the column and thus make the sentence appear to be more important than it is. In tests on the MSE 2005 data, we found that this works better than the standard pivoted-QR decomposition to identify sentences that provide distinct information.

4 CLASSY Experiments

Four experiments that we ran for the Multilingual Summarization Evaluations and afterward are discussed here. Data for MSE consisted of clusters of related documents where each cluster contained some number of English and Arabic documents. Machine translated versions of the Arabic documents were also available. Figure 3 shows the ROUGE-2 scores for both human and system-generated summaries.

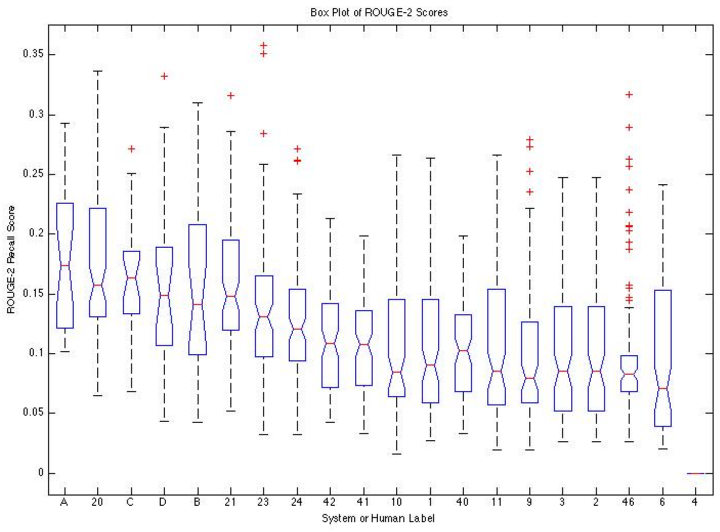


Fig. 3. Box Plot, sorted by mean, of ROUGE-2 Scores for All Humans and Systems. Letters represent human summary scores. Experiment 2 (Sect. 4.2) below is system 20; experiment 3 (Sect. 4.3) is system 21.

The experiments differed in which documents were used to select sentences for the summary and which documents were used to compute signature terms.

4.1 Experiment 1—English and Arabic Source Documents

The 2004 Document Understanding Conference (DUC '04, [13]) included two tasks to generate very short ($\leq$ 75 bytes, i.e., headlines) and short ($\leq$ 665 bytes) generic summaries using both MT-generated and "related" English background documents. The Lakhas system [5] used the original Arabic documents, rather than the translations, to generate headlines, which were then translated to English. Lakhas outperformed all the other systems that participated in this task.

Based on this result, we decided to experiment with using the original Arabic documents, rather than the MT translations, for one of our submissions to MSE. The Arabic documents were tokenized as 6-grams[2]. Signature tokens for each set of Arabic documents in a cluster were computed against an Arabic corpus. Independently, signature words were computed for the English documents in each cluster. Both the original Arabic and English sentences were scored using our summarization algorithms with the appropriate set of signature terms. When an Arabic sentence was selected for the summary, it was replaced with the corresponding sentence from the MT version of the document.

[2] 6-grams were chosen as a "reasonable" character length for tokens without creating too much of a computational load. For future efforts, we will most likely use white space splits for token identification.

We had expected this submission to perform well and were surprised when it scored lower than using the MT translations of the documents (described in Sect. 4.3). However, it still scored better than all submissions from other participants. We hypothesize that any gain we had from using the original Arabic was more than offset by the substitution of sentences from the noisy machine translations. This is consistent with results seen in Sects. 4.2, 4.3, and 4.4.

4.2 Experiment 2—English Documents Only

For this experiment, we used both the English and machine translations of the Arabic documents to compute signature terms for each cluster. Using the Arabic translations to compute the signature terms gave us larger clusters, which can improve the quality of the signature terms. However, in order to mitigate the noisy effects of machine translation, we chose sentences from the English documents only.

This English-only submission ranked first among all participating systems in MSE. Remarkably, the ROUGE [10] automatic evaluation system scores were better than 3 of the 4 human-generated summaries for ROUGE-2 and ROUGE-SU4, and 2 of the 4 human-generated summaries for ROUGE-1, and within the 95% confidence intervals for those humans who outscored the system. While CLASSY's performance is impressive, there are three points to remember. First, while the ROUGE performance measures have been shown to correlate well with human evaluation [10], they clearly are not a replacement. (We will address human evaluation in Sect. 4.5.) Second, the performance of the humans was limited by the poor quality of the translated documents. Third, we were able to exploit the fact that every Arabic document in a cluster had a closely related English document which, of course, is not always the case. Figure 4 shows a human-generated summary along with the CLASSY summary for the same document set.

4.3 Experiment 3—English and Translated Arabic Documents

Signature terms were computed identically as for Experiment 2 for this experiment. In this case, however, we used both the English documents and the machine translations of the Arabic documents to select sentences for the summary.

This English/MT-Arabic submission ranked second among all participating systems in MSE. While it was *always* outside the 95% confidence interval of the English-only submission on each of the ROUGE scores, it was always *within* the 95% confidence interval of at least 2 of the 4 human-generated summaries. We conjecture that the quality of the machine translation degraded both our summaries and the human summaries in a similar way.

4.4 Experiment 4—English Only

For this experiment, we used *only* the English documents for both signature term computation as well as summary selection. The purpose of this experiment was to measure the impact of using the translated Arabic documents for signature

Bombs exploded outside churches in Jakarta and five other Indonesian cities and towns on Christmas Eve, killing at least 14 people, injuring dozens and worsening the tension between Muslims and Christians. There were no immediate claims of responsibility, but religious violence and tensions have been rising throughout the predominantly Muslim country. Christians make up less than 10 percent, mostly ethnic Chinese, of Indonesia's 210 million people. President Abdurrahman Wahid asked Christians not to be provoked and blamed the attacks on forces intent on destabilizing the government". The Christmas celebrations coincide with the final days of Ramadan, Islam's month of fasting.

a. A Human-Generated Summary

Bombs exploded outside churches in Jakarta and five other Indonesian cities and towns on Christmas Eve, killing at least 14 people, injuring dozens and worsening the already difficult relations between Muslims and Christians throughout the fractured archipelago. Most of the bombs were planted in cars parked outside targeted churches _ including Jakarta's Roman Catholic cathedral, near the presidential palace and the capital's main mosque. Most of Indonesia's religious violence has been in the Moluccan islands, where about 5,000 Christians and Muslims have been killed over the past two years. Four of the dead Sunday were police officers who tried to

b. CLASSY-Generated Summary

Fig. 4. Example of Human- and CLASSY-generated Summaries

term computation (see Sect. 4.2). We computed a two-sided rank sum test, to test if the median ROUGE-2 Recall scores for both Experiment 2 and this experiment are equal for the MSE data. Forty-seven (47) of the scores from Experiment 2 were higher than their corresponding score from this experiment, 42 were less, and 7 were equal. The overall significance is a p-value of 0.2435, which means we cannot reject the null hypothesis that the medians are equal. Therefore, we conclude that while using the translated Arabic documents to compute signature terms did improve the ROUGE-2 scores, the improvement is not statistically significant.

4.5 Human Assessment

In addition to the automatic evaluation with ROUGE, a human evaluation was done. Human assessors read all the documents (both English and translated Arabic) for each cluster and then assigned each of the human- and machine-generated summaries to one of 5 equivalence classes–Unacceptable, Somewhat acceptable, Acceptable, Good, and Excellent (1 to 5, respectively)–describing *overall responsiveness* to the information presented in the documents in a cluster. Figure 5 is a scatter plot of the ROUGE-2 versus average overall responsiveness, the human evaluation score. The 8 machine systems and 4 human summaries

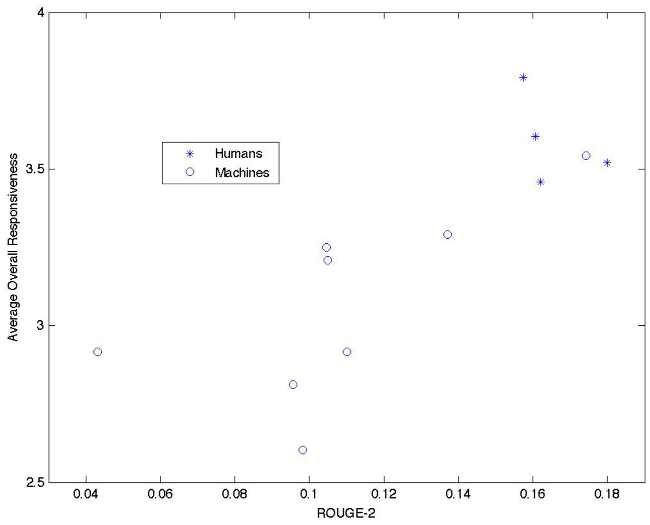


Fig. 5. Scatter Plot of ROUGE-2 Scores vs. Human Evaluation of Responsiveness for MSE systems. Our system is represented by the circle farthest to the right and to the top.

scores are displayed; CLASSY (system 20 in the scatter plot) is the only system to score at human levels of performance.

5 Conclusion and Continuing Efforts

Using the translated Arabic in conjunction with English to compute signature terms but then selecting sentences from *only* the English documents was a very successful approach. This perhaps indicates, as we have previously conjectured, that the Arabic documents in these collections did not provide any information beyond that contained in the English documents.

The summaries which used all documents for both computing signature terms *and* sentence selection, were statistically worse in a number of the ROUGE measures. We can only conclude that the inclusion of the MT sentences degraded the summary. With this said, this method scored second among all the submissions in all ROUGE measures.

These results indicate that when presented with a combination of documents in both English and Arabic (or, we suspect, any other language), that CLASSY, using signature terms computed from both English and the MT-versions of the Arabic documents, generates very good quality summaries.

A great deal more remains to be done. We realize that non-English documents will not always be as similar to "comparable" English documents as with the MSE data set. We would like to continue working with the original Arabic documents to better exploit them for the information and perspective that they contain, as compared to the English documents. We would also like to find ways to improve the machine translations of the documents in order to more effectively use the translated content of the documents.

For both of these, we would like to improve basic non-English language tasks such as sentence splitting and lemmatization. Arabic presents serious challenges for these tasks, as do other languages. Early experiments suggest, however, that improvements to these would yield significant improvements to both the MT and summarization tasks.

We would also like to evaluate each of the components of CLASSY on languages other than English. For example, we do not know if the method we use for redundancy removal will be effective on non-English languages. Our trimming methods are truly language dependent. We would like to identify a class of trims that are "universal" for all languages, even when they appear quite different in different languages. We also need to compile trims that are useful for a single language or class of languages.

Acknowledgments

The authors thank Michael Graham for the work he did in developing the sentence splitter that we are now using. We also wish to thank the organizers of both MSE 2005 and 2006 for all their hard work in creating the data sets and performing the evaluation.

The work of Dianne O'Leary was partially supported by NSF Grant CCF-0514213.

References

1. Conroy, J.M., et al.: Left-Brain Right-Brain Multi-Document Summarization. In: Procs. of 2004 Document Understanding Conference, Boston, MA (2004), `http://duc.nist.gov/`
2. Conroy, J.M., Schlesinger, J.D., O'Leary, D.P.: Topic-focused Multi-document Summarization Using an Approximate Oracle Score. In: Procs. of 2006 ACL/COLING Conference, Sydney, Australia (2006)
3. Conroy, J.M., Schlesinger, J.D., Stewart, J.G.: CLASSY Query-based Multi-document Summarization. In: Procs. of 2005 Document Understanding Conference, Vancouver, BC (2005), `http://duc.nist.gov/`
4. Conroy, J.M., et al.: Back to Basics: CLASSY 2006. In: Procs. of 2006 Document Understanding Conference, New York (2006), `http://duc.nist.gov/`
5. Douzidia, F.S., Lapalme, G.: Lakhas, an Arabic Summarization System. In: Procs. of 2004 Document Understanding Conference, Boston, MA (2004), `http://duc.nist.gov/`
6. Evans, D.K., McKeown, K., Klavans, J.L.: Similarity-based Multilingual Multi-document Summarization. Technical Report CUCS-014-05, Department of Computer Science, Columbia University (2005)
7. Fung, P., Ngai, G.: One Story, One Flow: Hidden Markov Story Models for Multilingual Multidocument. ACM Trans. on Speech and Language Processing (TSLP) 3(2) (2006)

8. Hatzivassiloglou, V., et al.: Simfinder: A Flexible Clustering Tool for Summarization. In: Procs. of NAACL 2001 Workshop on Automatic Summarization, Pittsburg, PA (2001)
9. Lin, C.-Y., Hovy, E.: The Automated Acquisition of Topic Signatures for Text Summarization. In: Procs. of 18th Intl. Conference on Computational Linguistics (COLING 2000), Saarbrücken, Germany (2000)
10. Lin, C.-Y., Hovy, E.: Automatic Evaluation of Summaries Using N-gram Cooccurrence Statistics. In: Procs. of North American Chapter of the Association for Computational Linguistics, vol. 1, Edmonton, Alberta (2003)
11. Mani, I., Maybury, M.T. (eds.): Advances in Automatic Text Summarization. MIT Press, Cambridge (1999)
12. Nenkova, A., Vanderwende, L.: The Impact of Frequency on Summarization. Technical Report MSR-TR-2005-101, Microsoft Research (2005)
13. Over, P., Yen, J.: An Introduction to DUC-2004: Intrinsic Evaluation of Generic News Text Summarization Systems. In: Procs. of 2004 Document Understanding Conference, Boston, MA (2004), `http://duc.nist.gov/`
14. Porter, M.: An Algorithm for Suffix Stripping. Program 14(3), 130–137 (1980)
15. Radev, D., et al.: MEAD - A Platform for Multidocument Multilingual Text Summarization. In: Procs. of Fourth International Conference on Language Resources and Evaluation, Lisbon, Portugal (2004)
16. Schiffman, B., Nenkova, A., McKeown, K.: Experiments in Multidocument Summarization. In: San Diego, C.A. (ed.) Procs. of Human Language Technology Conference, San Diego, CA (2002)
17. Document Understanding Conferences Publications.
`http://www-nlpir.nist.gov/projects/duc/pubs.html`.

Lexical Cohesion Based Topic Modeling for Summarization

Gonenc Ercan and Ilyas Cicekli

Dept. of Computer Engineering
Bilkent University, Ankara, Turkey
ercangu@cs.bilkent.edu.tr, ilyas@cs.bilkent.edu.tr

Abstract. In this paper, we attack the problem of forming extracts for text summarization. Forming extracts involves selecting the most representative and significant sentences from the text. Our method takes advantage of the lexical cohesion structure in the text in order to evaluate significance of sentences. Lexical chains have been used in summarization research to analyze the lexical cohesion structure and represent topics in a text. Our algorithm represents topics by sets of co-located lexical chains to take advantage of more lexical cohesion clues. Our algorithm segments the text with respect to each topic and finds the most important topic segments. Our summarization algorithm has achieved better results, compared to some other lexical chain based algorithms.

Keywords: text summarization, lexical cohesion, lexical chains.

1 Introduction

Summary is the condensed representation of a document's content. For this reason, they are low cost indicators of relevance. Summaries could be used in different applications both as informative tools for humans and as similarity functions for information retrieval applications. Summaries could be displayed in search results as an informative tool for the user. The user can measure the relevance of a document that he gets as a result of a search on Internet by just looking its summary. In order to measure similarities between documents, their summaries can be used instead of whole documents, and indexing algorithms can index their summaries instead of whole documents.

Depending on its content, summaries can be categorized into two groups: *extract* and *abstract*. If a summary is formed of sentences that appear in the original text, it is called as an *extract*. A summarization system targeting extracts should evaluate each sentence for its importance. Abstracts are the summaries that are formed from paraphrased or generated sentences. Building abstracts has additional challenges.

Different clues can be exploited to evaluate the importance of sentences. There are extractive summarization systems that take advantage of surface level features like word repetition, position in text, cue phrases and similar features that are easy to compute. Ideally, a summarization system should perform full understanding, which is very difficult and only domain dependant solutions are currently available.

Some summarization algorithms including ours, rely on more sophisticated clues that require deeper analyses of the text. A meaningful text is not a random sequence of words, and it has a semantic integrity to explain one or more topics. In linguistics, coherence is used to define the semantic integrity of a document, and it can be thought as a hidden element which provides the feeling that a document is written intelligently. Since modelling coherence which indicates the semantic structure of a document is difficult, researchers looked other low-cost measures for the semantic structures of documents. Cohesion [8] is simpler than coherence and it can also help to determine the discourse structure in the text. Cohesion is a surface level feature, and it deals with the relationships between text units. Some cohesion relations are lexical cohesion (use of related terms), co-reference, ellipsis and conjunction. Co-reference, ellipsis and conjunction are harder to identify than lexical cohesion.

Modeling the lexical cohesion structure of a text depends on the semantic relations between words in the text. The lexical cohesion structure of the text can be modeled with lexical chains [10]. Lexical chains are connected graphs, where the vertices are intended senses (meanings) of the words and the edges are the semantic relations between these senses. A lexical chaining algorithm needs an ontology to acquire the semantic relations between senses. WordNet is such an ontology, which is used by our algorithm and other lexical chaining algorithms in the literature. In order to find the lexical chains for the text, the intended sense for each word in the text must be determined. This is also known as Word Sense Disambiguation (WSD).

In lexical chaining algorithms, the WSD is done by assuming that the intended sense of a word is more related to other words surrounding the word. Morris and Hirst define the first lexical chaining algorithm, which is a greedy algorithm that tries to disambiguate each word using the context before its occurrence [10]. Barzilay and Elhadad disambiguate the words by checking all possible interpretations of the text [1]. Galley and McKeown improve Barzilay's algorithm both in terms of running time and WSD accuracy [7]. Galley imposes a "one sense per word" constraint and fuses the clues gathered from different occurrences of a word to a single decision for the word's correct sense. In our algorithm, we are using a very similar algorithm to Galley et al.'s algorithm. Our lexical chaining procedure differs only in the WordNet relations used in the algorithm. Our algorithm also uses meronym and holonym relations while their algorithm does not consider these relations.

Barzilay and Elhadad introduce a text summarization algorithm based on lexical chains [1]. They claim that cohesion relations could provide good results in text summarization. Their algorithm uses lexical chains to detect and represent topics. Their lexical chaining algorithm also depends on an explicit text segmentation algorithm. They report that they have experimented with different sentence extraction criterion, and selecting the first sentences of the strongest lexical chains yields the best results. A strength criterion used by Barzilay is shown in Equation 1, and *Homogeneity* is shown in Equation 2. In those equations, *Length* is the number of all members of the lexical chain, and *#DistinctMembers* is the number of the distinct members of the lexical chain.

$$Score(Chain) = Length * Homogeneity \tag{1}$$

$$Homogeneity = 1 - \frac{\#DistinctMembers}{Length} \tag{2}$$

Brunn et al.[2] also use lexical chains for text summarization. Just like Barzilay et al.'s algorithm, they use an explicit text segmenter. Two phase sentence selection procedure is applied. First the segments are ranked with lexical chain scores. From the best scoring text segments, the most scoring sentences are selected. Doran et al. [4] describe a similar summarization algorithm.

In all of these algorithms only lexical cohesion is used. These algorithms treat topics as single lexical chains. We believe that a single lexical chain can not represent a whole topic by itself. Usually a topic receives contributions from several lexical chains. Our algorithm tries to exploit other lexical cohesion clues like substitution, co-reference and ellipsis without detecting them. The lexical chains that tend to co-occur in a text can indicate a context specific relation between these lexical chains. Three lexical chains could correspond to a topic as their members could correspond to ”What”, ”When” and ”Where” portions of the topic. In our algorithm, we try to cluster related lexical chains in order to represent topic in the text.

We present our summarization algorithm in Section 2. In that section, we explain how we clusters the lexical chains and how we select important sentences using these clusters. In Section 3, we evaluate the results of our summarization system by comparing the results of the summarization systems in DUC2004[12]. Finally, we give some concluding remarks in Section 4.

2 Summarization Algorithm

Our algorithm divides the text into segments, according to topics, using the lexical chains extracted from the text. Topics in the text are roughly determined using lexical chains. Through clustering of lexical chains, our algorithm produces more granular segments. In each segment, it is assumed that the first sentence is a general description of the topic, and the first sentence of the segment is included in the summary.

Our algorithm is based on lexical chains, for this reason, our system requires deeper analysis of the text. An outline of our algorithm could be given as:

1. Sentence Detection
2. Part of Speech Tagging
3. Noun Phrase Detection
4. Lexical Chaining
5. Filtering Weak Lexical Chains
6. Clustering Lexical Chains Based on Co-occurrence
7. Extracting Sequences / Segmenting the Text Regard to Clusters.

Part of Speech Tagging is done using the MaxEnt Part of Speech Tagger [11]. We have implemented a noun phrase skimmer that uses the part of speech tags to detect noun phrases. Noun phrases usually end with a head noun. This head noun is accompanied by zero or more pre-modifiers, which usually are nouns or adjectives. The nouns and simple noun phrases of a document are found at the end of the first three steps of our algorithm, and the lexical chains are created for them in the fourth step.

Our lexical chaining algorithm is an implementation of Galley et al.'s algorithm [7], and it is also used in a keyphrase extraction system based on lexical chains [6]. After lexical chains are constructed for the text, there will be some weak lexical chains formed of single word senses. These lexical chains can cause complications in topic identification and segmentation. The formula in Equation 1 is introduced by Barzilay et al. [1], and this formula has been formulated to reflect the strength of lexical chains. Barzilay et al. report that this is the best formula that correlates with the human judges. After lexical chain construction, Barzilay suggests that lexical chains below a certain strength criterion should be filtered. We use strength criterion defined in Equation 3 to filter weak lexical chains before clustering lexical chains. In Equation 3, *Score(Chain)* is the score of the lexical chain, *Avg(Scs)* is the average of the scores of all lexical chains, and *StdDev(Scs)* is the standard deviation of the scores. This equation is first introduced by Barzilay et al. [1] and they report that this criterion correlates with the human judgements.

$$Score(Chain) > Avg(Scs) + 2 * StdDev(Scs) \tag{3}$$

After the weak lexical chains are filtered, the remaining lexical chains are clustered using co-occurrence information. We hope that the remaining strong clusters represent major topics of the text, and important sentences are extracted from these strong clusters. The details of clustering and sentence extraction are discussed in the rest of this section.

2.1 Clustering Lexical Chains

All strong lexical chains in the document are clustered using co-occurrence statistics. A single lexical chain may not be sufficient to represent a single topic. Our summarization algorithm uses clusters of lexical chains in order to represent topics in the text. Figure 2 gives the important clusters of lexical chains constructed for the document in Figure 1[1].

A topic could be formed of words that are not necessarily co-related. For example, in Figure 2 cluster 2 is a good example. This cluster talks about an '*arrest*' in '*London*' on '*Sunday*'. These three sets and their relations with each other can only be determined by the current context. We believe that through clustering, we are forming a relation between these lexical chains. In cluster 2, lexical chains in the cluster are forming up the relations 'what', 'where' and 'when' respectively. Our clustering algorithm uses a very simple assumption: "if two lexical chains tend to appear in same sentences, then there may be a relation between two sets in the given context". It is clear that, this will not hold in all cases. There will be falsely related lexical chains, however, a more accurate algorithm requires deeper semantic analysis. Our approach is just accurate enough for our segmentation algorithm.

In cohesion relations, like reference, substitution and ellipsis, a word is not repeated in each sentence but replaced or omitted. Through clustering, we can be able to account for cohesion clues other than lexical cohesion, for example ellipsis. By forming the link between two or more lexical chains by co-occurrence, it is possible to consider all lexical cohesion relations while segmenting the text.

For each lexical chain LC_i, a sentence occurrence vector V_i is formed. $V_i = \{s_{1_i}, ... s_{k_i} ... s_{n_i}\}$ where n is the number of sentences in the document. Each s_{k_i} is the number

[1] Proper names in the text, 'Pinochet' and 'Frei' are not present in WordNet. We have ignored nouns that are not in WordNet. Thus, 'Pinochet' and 'Frei' are not considered in our algorithm.

Cuban President Fidel Castro said Sunday he disagreed with the arrest in London of former Chilean dictator Augusto Pinochet, calling it a case of 'international meddling.' 'It seems to me that what has happened there (in London) is universal meddling,' Castro told reporters covering the Ibero-American summit being held here Sunday. Castro had just finished breakfast with King Juan Carlos of Spain in a city hotel.He said the case seemed to be 'unprecedented and unusual.' Pinochet, 82, was placed under arrest in London Friday by British police acting on a warrant issued by a Spanish judge. The judge is probing Pinochet's role in the death of Spaniards in Chile under his rule in the 1970s and 80s. The Chilean government has protested Pinochet's arrest, insisting that as a senator he was traveling on a diplomatic passport and had immunity from arrest. Castro, Latin America's only remaining authoritarian leader, said he lacked details on the case against Pinochet, but said he thought it placed the government of Chile and President Eduardo Frei in an uncomfortable position while Frei is attending the summit. Castro compared the action with the establishment in Rome in August of an International Criminal Court, a move Cuba has expressed reservations about. Castro said the court ought to be independent of the U.N. Security Council, because "we already know who commands there," an apparent reference to the United States. The United States was one of only seven countries that voted against creating the court. "The (Pinochet) case is serious ... the problem is delicate" and the reactions of the Chilean Parliament and armed forces bear watching, Castro said. He expressed surprise that the British had arrested Pinochet, especially since he had provided support to England during its 1982 war with Argentina over the Falkland Islands. Although Chile maintained neutrality during the war, it was accused of providing military intelligence to the British. Castro joked that he would have thought police could have waited another 24 hours to avoid having the arrest of Pinochet overshadow the summit being held here. "Now they are talking about the arrest of Pinochet instead of the summit," he said. Pinochet left government in 1990, but remained as army chief until March when he became a senator-for-life.

Fig. 1. An Example News Article

of LC_i members in the sentence k. If sentence k has 3 members of LC_i then s_{k_i} is 3. Two lexical chains LC_i and LC_j goes into the same cluster if their sentence occurrence vectors V_i and V_j are similar. As a result of clustering of lexical chains, we will get the following two properties:

– Lexical chains that co-occur will be in the same cluster. These lexical chains form a set of related topics that talk about a single topic.
– Lexical chains that span different sentences will be in different clusters. Two lexical chains that are in different clusters are considered to be unrelated.

Our clustering algorithm starts from an initial cluster distribution, where each lexical chain is in its own cluster. Thus, our clustering algorithm starts with n clusters, where n is the number of lexical chains. Iteratively, the most similar cluster pair is found and they are merged to form a single cluster. Clustering stops when the similarity between the most similar clusters is lower than a predefined threshold value.

The similarity between two clusters is measured by finding the similarity between the least similar members of the cluster. This is called *complete link clustering*. Since cluster members are lexical chains in our algorithm, a similarity function measuring the co-occurrence between two lexical chains is needed. We have used cosine similarity for this purpose. Lexical chain occurrence vector V_i is a vector in an m dimensional space,

Cluster1 :
LC_1= {Castro, Castro, chief, Castro, Castro, Castro, Castro, Castro, Castro, leader}
V_1 = {1,1,1,0,0,0,0,2,1,1,0,1,0,0,1,0,1}
LC_2 ={establishment, United States, parliament, United States, government, government, government}
V_2 = {0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,1}
Cluster2 :
LC_1= {action, march, meddling, arrest, arrest, arrest, surprise, arrest, meddling, arrest, arrest}
V_1 = {2,1,0,0,1,0,2,0,1,0,0,0,1,0,1,1,1}
LC_2 = {London, London}
V_2 = {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
LC_3 = {Sunday, Sunday}
V_3 = {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
Cluster 3 :
LC_1 = {summit, summit, summit, summit}
V_1 = {0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,1,0}
Cluster 4 :
LC_1 = {Chile, Argentina, Chile, Chile}
V_1 = {0,0,0,0,0,1,0,1,0,0,0,0,1,1,0,0,0}
LC_2 = {war, war}
V_2 = {0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0}
Cluster 5 :
LC_1 = {court, court, court}
V_1 = {0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0}

Fig. 2. Lexical Chain Clusters for the Example in Figure 1

where m is the number of sentences. The angle between two vectors could be used to find the similarity of two vectors. Between two vectors that are in the same direction, there will be an angle of 0 degrees. Cosine of two vectors can be calculated by Equation 4. This value is between 0 and 1, where 1 means most similar.

$$\cos(\theta) = \frac{V_i \cdot V_j}{\|V_i\| \, \|V_j\|} \tag{4}$$

Equation 4 is a well known formula from linear algebra, to find the cosine of the angle between two vectors. In the equation, $\|V_i\|$ represents the Euclidean length for the vector, that is the square root of the sum of squares of vector's dimension values .

2.2 Sequence Extraction or Text Segmentation

Some previous algorithms for lexical chain based summarization such as Brunn et al. [2], and Barzilay et al. [1] use explicit segmentation algorithms that does not take advantage of semantic relations. In our algorithm, the text is segmented from the perspective of each lexical chain cluster, and the hot spots for each topic are found. For each cluster, connected sequences of sentences are extracted as segments. Sentences that are cohesively connected are considered as sentences that are talking about the same topic.

$$V_1 = \{\boxed{1,1,1},0,0,0,\boxed{0,2,1,1,0,1},0,0,\boxed{1},0,\boxed{1}\}$$
$$V_2 = \{\boxed{0,0,0},0,0,0,\boxed{1,1,1,1,1,1},0,0,\boxed{0},0,\boxed{1}\}$$

Fig. 3. Text Segmentation for Cluster 1 Given in Figure 2

For each lexical chain cluster Cl_j, we form sequences separately. For each sentence S_k, if sentence S_k has a lexical chain member in Cl_j, a new sequence is started with sentence S_k or the sentence is added to the current sequence. If there is a current sequence, sentence S_k is added to this sequence; otherwise sentence S_k starts a new sequence. If there is no cluster member for sentence S_k, then the current sequence is ended. By using this procedure, text is segmented with respect to a cluster, identifying topic concentration points.

Figure 3 gives an example of the text segmentation for the document in Figure 1 with respect to cluster 1 that is given in Figure 2. In cluster 1, there are two lexical chains. The sentence occurrence vectors for these lexical chains are plotted in Figure 3, and boxed areas correspond to the sequences in the text. The topic seems to be concentrated in the second sequence, and the second sequence has contributions from both of the lexical chains and spans more than the other sequences.

After finding sequence, each sequence s_i is scored using the formula in Equation 5.

$$S(s_i) = S(Cl_i) * L_i * \frac{(1 + SLC_i) * PLC_i}{f^2} \tag{5}$$

$S(s_i)$ in Equation 5 is the score of segment with respect to cluster i. In Equation 5, L_i is the number of sentences in the sequence s_i, SLC_i is the number of lexical chains that starts in the sequence s_i, PLC_i is the number of lexical chains having a member in the sequence s_i, and f is the number of lexical chains in cluster i. Score of the cluster $S(Cl_i)$, is the average score of the lexical chains in the cluster. Our scoring function tries to model the connectedness of the segment using this cluster score. In order to evaluate this score, the scores of the lexical chains in the cluster are calculated with the formula in Equation 1. The number of sentences in the segment reflects how long the topic is discussed locally. Our algorithm tries to select the segments that lexical chains are starting in, and this will encourage the selection of the segments where the topic is first introduced in.

2.3 Sentence Selection

Humans tend to first explain the topic more generally, and then they give details in the following sentences. With this motivation, our algorithm extracts the first sentence of each sequence. So, if the extracted sequences are truly topic segments for the text, then our algorithm will extract the first sentence of the new topic. This technique depends on the assumption that, first sentences are general descriptions of the topic and this general description does contain sufficient information to represent the text segment in the summary.

The Chilean government has protested Pinochet's arrest, insisting that as a senator he was traveling on a diplomatic passport and had immunity from arrest. Cuban President Fidel Castro said Sunday he disagreed with the arrest in London of former Chilean dictator Augusto Pinochet, calling it a case of international meddling. Castro compared the action with the establishment in Rome in August of an International Criminal Court, a move Cuba has expressed reservations about. He expressed surprise that the British had arrested Pinochet, especially since he had provided support to England during its 1982 war with Argentina over the Falkland Islands.

Fig. 4. Extract of the Text in Figure 1

For a summary of length n sentences, n best scoring sequence's first sentences are included in the summary. However, two different sequences found from different lexical chain clusters can start with the same sentence. A problem with this approach may be that n could be higher than the number of sequences starting with a unique sentence, so the number of sentences to be included in the summary is limited by the number of sequences starting with unique sentences. It is possible for two sequences extracted from different lexical chain clusters to overlap in text area. The order of sentences in the output summary depends on the score of the sequence, which the sentence is extracted from. Sentences selected from the best scoring sequence will be the first sentence in the output summary.

We will try to demonstrate our algorithm using the news article in Figure 1. After lexical chaining and clustering, top ranking clusters are given in Figure 2. In cluster 4, the connection between 'Chile' and 'Argentina' is 'war'. This is discovered from the given context using co-occurrence in the given text. Clustering increases the connectedness of sentences, resulting in granular text segments. Sequences are extracted for these clusters. As a result of this process, summary in Figure 4 is extracted.

3 Evaluation Metrics

Evaluating summarization algorithms is a difficult task and it is a separate research area in Natural Language Processing. A summary's quality can be evaluated in different aspects: selected contents importance, and presentation quality. Presentation quality itself is composed of two aspects: grammatical correctness and coherence. Since we are extracting sentences from the original text, the grammatical correctness in sentences is guaranteed to be as good as the source document's grammatical correctness. Coherence in our solution is a problem as our algorithm does not consider anaphora resolution and information ordering. However, since we extract the first sentences of topic segments, anaphoric references in our extracts are not common.

An evaluation method is the evaluation of summaries by human judges. However, comparing the contents of automatically built summaries with human extracted summaries is a more fair methodology. Automatic evaluation is done using distributed similarity techniques. The similarity between the model summary and the system output reflects the summary quality. The overlap of text units between the system output and the model summaries is used as a quality metric. In the evaluation procedure, it is more appropriate to use multiple model summaries by different summarizers, since summarization is a subjective task.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [9] is one of the most popular summarization evaluation methodologies. ROUGE calculates the recall of text units using N-grams, LCS (Longest Common Subsequences) and Weighted Longest Common Subsequences. All of these metrics are aimed to find the percentage of overlap between the system output and the model summaries. ROUGE-N score is the percentage of overlap calculated using N-grams. ROUGE-L score is calculated using LCS and ROUGE-W score is calculated using Weighted LCS.

3.1 Experiment Setting and Results

We have tested our summarization algorithm with the news article corpus used in DUC2004 [12]. In order to properly evaluate our algorithm, and compare with existing algorithms, we have attempted task 1 of DUC2004. In this task, all summarization systems provide a 75 character summary for each of the 500 articles. Each summary is automatically evaluated against 4 model summaries extracted by professional humans. While calculating ROUGE scores, words in both the model and the system output are stemmed using Porter Stemmer. Weight for calculating the WLCS is assigned as 1.2. These are the values used in DUC2004 and we have used the same values to be compatible with their evaluation.

Table 1 shows the scores for our system, the best system and the worst system of the 40 systems participated in DUC2004. The average score of the participants of DUC2004 is also given in this table. We also included the scores of two systems, which are also participants of DUC2004 and they also use lexical cohesion methods for summarization. Lethbridge University's [3] summarization system also attacks automated summarization problem using lexical chains. Their algorithm uses an explicit text segmenter, and after building lexical chains they score each segment using the lexical chains. From the best segments, they select sentences. This algorithm is derived from Brunn et. al.'s algorithm [2]. Another algorithm using lexical cohesion in DUC2004 is the system developed in Dublin University [5]. This system extracts phrases instead of sentences. System ranks each phrase using TFxIDF (Term Frequency x Inverse Document Frequency), position of word, lexical cohesion score and POS tags. They use C5.0 machine learning algorithm to classify these phrases.

Our implementation of Barzilay et al.'s algorithm uses our lexical chaining procedure, but uses their selection procedure. Their algorithm selects the first sentence where a lexical chain member occurs in. In their algorithm, a strong lexical chain contributes to the summary with only one sentence. They assume that a lexical chain is a topic and the first sentence is the most important sentence.

Since a lexical chaining algorithm's word sense disambiguation accuracy is as low as %63, it is possible that the first member of a lexical chain is an error. In our algorithm, lexical chains are used as an intermediate tool to find topic segments. Segments are identified by combining the cues obtained from co-occurring lexical chains. Co-occurring lexical chains may capture context specific relations and other cohesion patterns. Our segments reflect the lexical cohesion hot spots, while the whole lexical chain reflects a set of related terms that may be scattered to the whole document. We select the first

Table 1. ROUGE Scores of our System and Other Participants of DUC2004

	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-W
Barzilay	0.17861	0.04381	0.15577	0.09508
Lethbridge	0.12135	0.02504	0.10852	0.06604
Dublin	0.22192	0.02543	0.1766	0.10169
Our System	0.19549	0.05247	0.17078	0.1034
Average	0.1858	0.04082	0.15803	0.09470
Best System	0.2511	0.06528	0.20109	0.11953
Worst System	0.12088	0.00731	0.10678	0.06564

sentences of the most lexically cohesive segments. We believe that our sentence selection procedure is more prone to errors in lexical chaining than Barzilay's algorithm.

3.2 Results

Scores of our system is promising as it is above Barzilay's algorithm. Also Lethbridge University's algorithm has obtained results below our system. System by Dublin university is above our algorithm in ROUGE-1 scores. However they have lower scores in other ROUGE scores, this is mainly because their algorithm outputs phrases. In DUC2004 evaluation, stop words are not removed when calculating recall. The model summaries for evaluation are formed of sentences containing stop words, for this reason their system has lower matches of sequences of words.

Table 2. ROUGE Ranks of our System and Other Participants of DUC2004

	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-W
Barzilay	28	15	26	22
Lethbridge	41	38	41	41
Dublin	5	36	7	9
Our System	17	8	9	7

Table 2 shows the rank of each system when compared to participants of DUC2004 single document summarization task. Our system ranked in the first 10 in all of the scores except ROUGE-1 score, which is calculated using uni-grams. In overall, our system achieved very good results. These results reflect that our system has obtained competing results for the algorithms in DUC2004. Since our algorithm outperforms lexical cohesion based algorithms, such as Barzilay's algorithm, Dublin University's algorithm and Lethbridge University's algorithm, we can consider it as a promising attempt.

4 Conclusion

Our motivation for this work is based on the observation that a topic is formed of a group of lexical chains. This is mainly due to the fact that in the current context of the text, words can be related to each other with domain specific relations that can not be acquired from a general ontology. Our algorithm tries to find these relations from the current text. Although this seems to be a weak assumption, we have seen in our experiments that our algorithm achieved better results than other lexical chains based algorithms.

Our system achieves very good results in DUC2004, ranking in the first 10. Our system is purely extractive, some other competing algorithms are using techniques such as: sentence reduction, anaphora resolution and elimination of repetition. In other competing algorithms, there are some systems that focus on news article domain, tracking events. Reduction of sentences could improve ROUGE score as summaries extracted are limited in size, some systems does have similar approaches. Resolving anaphora, improves the performance as model summaries does not usually contain anaphora.

Acknowledgments

This work is partially supported by The Scientific and Technical Council of Turkey Grant "TUBITAK EEEAG-107E151".

References

1. Barzilay, R., Elhadad, M.: Using Lexical Chains for Text Summarization. In: Mani, I., Maybury, M.T. (eds.) Advances in Automatic Text Summarization, pp. 111–121. MIT Press, Cambridge (1999)
2. Brunn, M., Chali, Y., Pinchak, C.J.: Text summarization using lexical chains. In: Proceedings of the Document Understanding Conference (DUC 2001), New Orleans, LA (2001)
3. Chali, Y., Kolla, M.: University of lethridge summarizer at DUC04. In: Proceedings of the Document Understanding Conference (DUC 2004), Boston, USA (2004)
4. Doran, W.P., et al.: Assessing the impact of lexical chain scoring methods and sentence extraction schemes on summarization. In: Gelbukh, A. (ed.) CICLing 2004. LNCS, vol. 2945, pp. 627–635. Springer, Heidelberg (2004)
5. Doran, W., et al.: News story gisting at university college dublin. In: Proceedings of the Document Understanding Conference (DUC 2004), Boston, USA (2004)
6. Ercan, G., Cicekli, I.: Using lexical chains for keyword extraction. Information Processing & Management 43, 1705–1714 (2007)
7. Galley, M., McKeown, K.: Improving word sense disambiguation in lexical chaining. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), pp. 1486–1488 (2003)
8. Halliday, M., Hasan, R.: Cohesion in English. Longman, London (1976)
9. Lin, C.Y., Hovy, E.H.: Automatic evaluation of summaries using n-gram co-occurrence statistics. In: Proceedings of HLT-NAACL-2003, Edmenton, Canada (2003)
10. Morris, J., Hirst, G.: Lexical cohesion computed by thesaural relations as an indicator of the structure of text. Computational Linguistics 17, 21–43 (1991)
11. Toutanova, K., et al.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of HLT-NAACL-2003, Edmenton, Canada (2003)
12. Proceedings of the Document Understanding Conference (DUC 2004), Boston, USA (2004)

Terms Derived from Frequent Sequences for Extractive Text Summarization*

Yulia Ledeneva[1], Alexander Gelbukh[1], and René Arnulfo García-Hernández[2]

[1] Natural Language and Text Processing Laboratory,
Center for Computing Research, National Polytechnic Institute, DF 07738, Mexico
`yledeneva@yahoo.com`
`www.Gelbukh.com`
[2] Instituto Tecnologico de Toluca, Mexico
`renearnulfo@hotmail.com`

Abstract. Automatic text summarization helps the user to quickly understand large volumes of information. We present a language- and domain-independent statistical-based method for single-document extractive summarization, i.e., to produce a text summary by extracting some sentences from the given text. We show experimentally that words that are parts of bigrams that repeat more than once in the text are good terms to describe the text's contents, and so are also so-called maximal frequent sentences. We also show that the frequency of the term as term weight gives good results (while we only count the occurrences of a term in repeating bigrams).

1 Introduction

A summary of a document is a (much) shorter text that conveys the most important information from the source document. There are a number of scenarios where automatic construction of such summaries is useful. For example, an information retrieval system could present an automatically built summary in its list of retrieval results, for the user to quickly decide which documents are interesting and worth opening for a closer look—this is what Google models to some degree with the snippets shown in its search results. Other examples include automatic construction of summaries of news articles or email messages to be sent to mobile devices as SMS; summarization of information for government officials, businessmen, researches, etc., and summarization of web pages to be shown on the screen of a mobile device, among many others.

The text summarization tasks can be classified into single-document and multi-document summarization. In single-document summarization, the summary of only one document is to be built, while in multi-document summarization the summary of a whole collection of documents (such as all today's news or all search results for a query) is built. While we believe that our ideas apply to either case, in this work we have experimented only with single-document summaries.

* Work done under partial support of Mexican Government (CONACyT, SNI, SIP-IPN, COTEPABE-IPN, COFAA-IPN). The authors thank Rada Mihalcea for useful discussion.

The summarization methods can be classified into abstractive and extractive summarization [1]. An abstractive summary is an arbitrary text that describes the contexts of the source document. Abstractive summarization process consists of "understanding" the original text and "re-telling" it in fewer words. Namely, an abstractive summarization method uses linguistic methods to examine and interpret the text and then to find new concepts and expressions to best describe it by generating a new shorter text that conveys the most important information from the original document. While this may seem the best way to construct a summary (and this is how human beings do it), in real-life setting immaturity of the corresponding linguistic technology for text analysis and generation currently renders such methods practically infeasible.

An extractive summary, in contrast, is a selection of sentences (or phrases, paragraphs, etc.) from the original text, usually presented to the user in the same order—i.e., a copy of the source text with most sentences omitted. An extractive summarization method only decides, for each sentence, whether or not it will be included in the summary. The resulting summary reads rather awkward; however, simplicity of the underlying statistical techniques makes extractive summarization an attractive, robust, language-independent alternative to more "intelligent" abstractive methods. In this paper, we consider extractive summarization.

A typical abstractive summarization method consists in several steps, at each of them different options can be chosen. We will assume that the units of selection are sentences (these could be, say, phrases or paragraphs). Thus final goal of the extractive summarization process is *sentence selection*. One of the ways to select the appropriate sentences is to assign some numerical measure of usefulness of a sentence for the summary and then select the best ones; the process of assigning these usefulness weights is called *sentence weighting*. One of the ways to estimate the usefulness of a sentence is to sum up usefulness weights of individual terms of which the sentence consists; the process of estimating the individual terms is called *term weighting*. For this, one should decide what the terms are: for example, they can be words; deciding what objects will count as terms is the task of *term selection*. Different extractive summarization methods can be characterized by how they perform these tasks.

The paper is organized as follows. Section 2 summarizes the state-of-the-art of text summarization methods. In Section 3, some notions used for term selection in our method are introduced. Section 4 presents our experimental setting. Sections 5 and 6 describe the obtained experimental results for different term selection and term weighting schemes, respectively, which are compared in Section 7 with those of existing methods. Section 8 concludes the paper.

2 Related Work

Ideally, a text summarization system should "understand" (analyze) the text and express its main contents by generating the text of the summary. For example, Cristea *et al.* [4] perform sentence weighting according to their proximity to the central idea of the text, which is determined by analysis of the discourse structure.

However, the techniques that try to analyze the structure of the text involve too sophisticated and expensive linguistic processing. In contrast, most of the methods discussed in the literature nowadays represent the text and its sentences as a bag of simple features, using statistical processing without any attempts to “understand” the text.

Supervised learning methods consider sentence selection as a classification task: they train a classifier using a collection of documents supplied with existing summaries. As features of a sentence such methods can consider text units (in which case we can speak of term selection) or other, non-lexical characteristics. Villatoro-Tello *et al.* [7] use as terms *n*-grams found in the text. Kupiec *et al.* [2] use predefined cue phrases (this makes the method language- and domain-dependent) as well as non-lexical features such as the position and length of the sentence; their sentence weighting procedure also includes measuring the overlap of the sentence with the title of the document. HaCohen-Kerner *et al.* [18] consider many other lexical and non-lexical features, such as the position of the sentence in the paragraph.

However, the majority of current methods are purely heuristic: they do not use any learning but directly state the procedure used for term selection, term weighting, and/or sentence weighting (given that sentence selection in most cases consists in selecting the best-weighted sentences).

A very old and very simple sentence weighting heuristic does not involve any terms at all: it assigns highest weight to the first sentences of the text. Texts of some genres—such as news reports or scientific papers—are specifically designed for this heuristic: e.g., any scientific paper contains a ready summary at the beginning. This gives a baseline [12] that proves to be very hard to beat on such texts. However, comparing term-based methods with such position-based baseline is not fair in the sense that this baseline only works on text of specific genres (say, it will not work on official documents, email messages, webpages, or literary novels) and uses information (the position of the sentence) not available to term-based methods. It is worth noting that in Document Understanding Conference (DUC) competitions [12] only five systems performed above this baseline, which does not demerit the other systems because this baseline is genre-specific. Though the method proposed in this paper very slightly outperforms this baseline, such a comparison is unfair.

Of the works devoted to term-based methods, most concentrate on term weighting. Xu *et al.* [6] derives relevance of a term from an ontology constructed with formal concept analysis. Song *et al.* [3] basically weight a word basing on the number of lexical connections, such as semantic associations expressed in a thesaurus, that the word has with its neighboring words; along with this, more frequent words are weighted higher. Mihalcea [15] presents a similar idea in the form of a neat, clear graph-based formalism: the words that have closer relationships with a greater number of “important” words become more important themselves, the importance being determined in a recursive way similar to the PageRank algorithm used by Google to weight webpages.

The latter idea can be applied directly to sentence weighting without term weighting: a sentence is important if it is related to many important sentences, where relatedness can be understood as, say, overlap of the lexical contents of the sentences [15]. The two methods presented in [15] are those that currently give the best results and with which we compare our suggested method.

While in the experiments reported in the papers discussed above were based on words as terms, this is not the only possible option. Liu *et al.* [5] uses pairs of syntactically connected words (basic elements) as atomic features (terms). Such pairs (which can be thought of as arcs in the syntactic dependency tree of the sentence) have been shown to be more precise semantic units than words [19, 20]. However, while we believe that trying text units larger than a word is a good idea, extracting the basic elements from the text requires dependency syntactic parsing, which is language-dependent. Simpler statistical methods (cf. the use of *n*-grams as terms in [7]) may prove to be more robust and language-independent.

In this paper we analyze several options for simple language-independent statistical term selection and corresponding term weighting, based on units larger than one word. In particular, we show that so-called maximal frequent sequences (MFSs), as well as single words that are part of bigrams repeated more than once in the text, are good terms to describe documents.

3 Frequent Sequences

An ngram is a sequence of *n* words. We say that an ngram occurs in a text if these words appear in the text in the same order immediately one after another. For example, a 4-gram (ngram of length 4) *words appear in the text* occurs once in the previous sentence, while *appear immediately after another* does not (these words do not appear on adjusting positions), neither does *the text appear in* (order is different).

The definition of ngram depends on what one considers words. For example, one can consider capitalized (*Mr. Smith*) and non-capitalized (*a smith*) words as the same word or as different words; one can consider words with the same morphological stem (*ask*, *asked*, *asking*), the same root (*derive*, *derivation*), or the same meaning (*occur*, *appear*) as the same word; one can omit the stop-words (*the*, *in*) when counting word positions, etc. Say, one can consider that in our example sentence above there occur the ngrams *we say* (capitalization ignored), *word appear* (plural ignored), *appear text* (*in the* ignored). This can affect counting the ngrams: if one considers *occur* and *appear* as equivalent and ignores the stop-words, then in our example sentence the bigram *appear text* occurs twice.

We call an ngram frequent (more accurately, *β*-frequent) if it occurs more than *β* times in the text, where *β* is a predefined threshold. Frequent ngrams—we will also call them frequent sequences (FSs)—often bear important semantic meaning: they can be multiword expressions (named entities: *The United States of America*, idioms: *kick the basket*) or otherwise refer to some idea important for the text (*the President's speech*, *to protest against the war*).

Our hypothesis is that FSs can express ideas both important and specific for the document. This can be argued in terms of *tf-idf* (term frequency—inverse document frequency, a notion well-known in information retrieval [21]): on the one hand, the idea expressed by an FS is important for the document if it repeatedly returns to it (high term frequency); on the other hand, the corresponding idea should be specific for this document, otherwise there would exist in the language a single word or at least an abbreviation to express it (high inverse document frequency). It is important to note that this argument does not apply to 1-grams, i.e., single words. Therefore, we do not consider 1-grams as ngrams in the rest of this paper.

An ngram can be a part of another, longer ngram. All ngrams contained in an FS are also FSs. However, with the arguments given above one can derive that such smaller ngrams may not bear any important meaning by their own: e.g., *The United States of America* is a compound named entity, while *The United* or *States of America* are not. Exceptions like *The United States* should not affect much our reasoning since they tend to be synonymous to the longer expression, and the author of the document would choose one or another way to refer to the entity, so they should not appear frequently both in the same document.

FSs that are not parts of any other FS are called Maximal Frequent Sequences (MFSs) [10, 11]. For example, in the following text

... *Mona Lisa <u>is the most beautiful</u> picture of Leonardo da Vinci* ...
... *Eiffel tower <u>is the most beautiful</u> tower* ...
... *St. Petersburg <u>is the most beautiful</u> city of Russia* ...
... *<u>The most beautiful</u> church is not located in Europe* ...

the only MFS with $\beta = 3$ is *is the most beautiful*, while the only MFS $\beta = 4$ is *the most beautiful* (it is not an MFS with $\beta = 3$ since it is not maximal with this β). As this example shows, the sets of MFSs with different thresholds do not have to, say, contain one another.

One of our hypotheses was that only MFSs should be considered as bearing important meaning, while non-maximal FSs (those that are parts of another FS) should not be considered. Our additional motivation was cost vs. benefit considerations: there are too many non-maximal FSs while their probability to bear important meaning is lower. In any case, MFSs represent all FSs in a compact way: all FSs can be obtained from all MFSs by bursting each MFS into a set of all its subsequences. García [10] proposed an efficient algorithm to find all MFSs in a text, which we also used to efficiently obtain and store all FSs of the document.

The notions of FSs and MFSs are closely related to that of repeating bigrams; see Section 5. This set is conceptually simpler, but for computational implementation MFSs could be more compact.

4 Experimental Setting

We have conducted several experiments to verify our hypotheses formulated in the previous section.

Algorithm. In each experiment, we followed the standard sequence of steps:

- Term selection: decide which features are to be used to describe the sentences;
- Term weighting: decide how the importance of each feature is to be calculated;
- Sentence weighting: decide how the importance of the features is to be combined into the importance measure of the sentence;
- Sentence selection: decide which sentences are selected for the summary.

The specific settings for each step varied between the experiments and are explained below for each experiment.

Test data set. We used the DUC collection provided [12]. In particular, we used the data set of 567 news articles of different length and with different topics. Each document in the DUC collection is supplied with a set of human-generated summaries provided by two different experts.[1] While each expert was asked to generate summaries of different length, we used only the 100-word variants.

Evaluation procedure. We used the ROUGE evaluation toolkit [13] which was found to highly correlate with human judgments [14]. It compares the summaries generated by the program with the human-generated (gold standard) summaries. For comparison, it uses n-gram statistics. Our evaluation was done using n-gram (1, 1) setting of ROUGE, which was found to have the highest correlation with human judgments, namely, at a confidence level of 95%.

5 Term Selection

For term selection, we compared MFSs with more traditional features such as single words and ngrams. Namely, we considered the following variants of term selection:

- M**:** the set of all MFSs, i.e., an ngram $m \in M$ if it is an MFS with some threshold β (recall that MFSs are of 2 words or longer and $\beta \geq 2$)[2] In the example from Section 3, M = {*is the most beautiful*, *the most beautiful*}. Also, we denote by M_2 the set of all MFSs with $\beta = 2$.
- B**:** repeating bigrams, i.e., bigrams with frequency at least 2. It is easy to show that it is the same set as the set of all bigrams from MFSs: a bigram $b \in B$ iff there exists an MFS $m \in M$ such that $b \subseteq m$. What is more, considering in the latter definition M_2 instead of M also gives the same set. In our example, B = {*is the*, *the most*, *most beautiful*}.
- W**:** single words (unigrams) from elements of B or, which is the same, of M. Namely, a word $w \in W$ if there exists a bigram $b \in B$ such that $w \in b$; it is easy to show that $w \in W$ iff there exists an MFS $m \in M$ such that $w \in m$. Again, considering M_2 in the latter definition also gives the same set. In our example, B = {*is*, *the*, *most*, *beautiful*}.

We give different definitions of the sets B and W to show that they are naturally derived from the notion of MFS and at the same time can be efficiently calculated.

Optionally, stop-words were eliminated at the pre-processing stage; in this case our bigrams (or MFSs) could span more words in the original text, as explained in Section 3.

For term weighting, the frequency of the term was used; for sentence weighting, the sum of the weights of the terms contained in the sentence was used; for sentence

[1] While the experts were supposed to provide extractive summaries, we observed that the summaries provided in the collection were not strictly extractive: the experts considerably changed the sentences as compared with the original text.

[2] In practice, we only considered the MFSs with the thresholds β = 2, 3, and 4, since MFSs with higher thresholds were very rare in our collection, except for those generated by stop-words.

selection, the sentences with greater weight were selected until the desired size of the summary (100 words) is reached.

Table 1 shows the results. Since the size of all summaries is the same (100 words), either measure (recall or precision) can be used for comparison.

Table 1. Recall on 100-words summaries for different term selection options

Terms	With stop-words	Without stop-words
W: words from B or M	0.39421	0.41371
B: repeating bigrams	0.40810	0.42173
M: all MFSs	**0.43066**	**0.44085**

As a kind of statistical significance check, we randomly divided our test data into two halves and ran this (and most of the other) experiments separately on each subset. These experiments confirmed the qualitative observations reported in this paper.

As Table 1 shows, MFSs are a promising choice for term selection. This motivated our further experiments with term selection schemes derived from them, as well as with term weighting options for them.

6 Term Weighting and Sentence Selection

Inspired by the above results, we further experimented with MFSs and other term selection options derived from them. In addition to M and W from Section 5, we considered a generalization of the sets considered in Section 5:

– N: all ngrams from MFSs, i.e., an ngram $n \in N$ if there exists an MFS $m \in M$ such that $n \subseteq m$ (including single words, i.e., 1-grams). Again, considering in the latter definition M_2 also gives the same set, which allows for efficient calculation of the set N in practice. In our example, N = {*is*, *the*, *most*, *beautiful*, *is the*, *the most*, *most beautiful*, *is the most*, *the most beautiful*, *is the most beautiful*}. Note that $W \subset N$, $M \subset N$.

– $N \setminus W$, $N \setminus M_2$, $N \setminus (W \cup M_2)$: same as N but not including 1-grams, the whole MFS, or both; here M_2 is the set of MFSs with $\beta = 2$. In our example, $N \setminus (W \cup M_2)$ = {*is the*, *the most*, *most beautiful*, *is the most*, *the most beautiful*}.

Optionally, stop-words were eliminated at the pre-processing stage. For term weighting, different formulae were considered containing the following values:

– f: frequency of the term in MFSs, i.e., the number of times the term occurs in the text within some MFS. In our example, $f(is) = 3$ since it occurs 3 times in the text within the MFS *is the most beautiful*. If the term itself is an MFS, then this is just the frequency of this term in the text (e.g., for M, f is the same as term weight in Section 5; for W and N it is not). Under certain realistic conditions (MFSs do not intersect in the text, words do not repeat within one MFS) f is the number of times the term occurs in the text as part of a repeating bigram. In our example, $f(is) = 3$ since it occurs 3 times in a repeating bigram *is the* (and one time in a non-repeating context *church is not*).

– *l*: the maximum length of an MFS containing the term. In our example, *l*(*is*) = 4 since it is contained in a 4-word MFS *is the most beautiful*.
– 1: the same weight for all terms.

For sentence weighting, the sum of the weights of the terms contained in the sentence was used. For sentence selection, the following options were considered:

– best: sentences with greater weight were selected until the desired size of the summary (100 words) is reached. This is the most standard method.
– *k*best+first: *k* best sentences were selected, and then the first sentences of the text weight were selected until the desired size of the summary is reached. This was motivated by the very hard-to-beat baseline mentioned in Section 2: only the very best sentences according to our weighting scheme might prove to be above this baseline.

The results are shown in Table 2. We conducted our experiments in three phases. From Table 1 we knew that term selection scheme *M* with stop-words removed gave the best results with other parameters fixed (term weighting, sentence weighting, and sentence selection). So we started from modifying these parameters for the same term selection scheme; see the upper third part of Table 2. The first line of the table represents the best result from Table 1. The best results are highlighted in boldface. From this experiment, we discarded the term weighting options related to *l*.

Table 2. Results for different term selection options

Term Selection		Term weighting	Sentence Selection	Results		
Terms	Stop-words			Recall	Precision	F-measure
M	excluded	*f*	best	0.44085	0.45564	0.44796
		1		**0.44128**	**0.45609**	**0.44840**
		l		0.43977	0.45587	0.44752
		l^2		0.42995	0.44766	0.43847
		$l \times f$		0.43812	0.45411	0.44581
	included			0.43353	0.44737	0.44022
W	included	*f*	best	0.44582	0.45820	0.45181
	excluded			**0.44609**	**0.45953**	**0.45259**
		1		0.38364	0.40277	0.39284
		f^2		0.43892	0.45265	0.44556
N		*f* or 1		0.43711	0.45099	0.44383
W	excluded	*f*	1best+first	**0.46576**	**0.48278**	**0.47399**
			2best+first	0.46158	0.47682	0.46895
M		1	1best+first	0.46354	0.48072	0.47185
			2best+first	0.46028	0.47567	0.46772
		l	1best+first	0.46381	0.48124	0.47223
			2best+first	0.45790	0.47430	0.46583

Then we tried other term selection options, such as *W* and *N*, with the term weighting option 1 and the options related to *f*, which showed good performance in the first experiment. The results are shown in the middle third of Table 2. Term selection *W* gave a slightly better result than *M*. The results for *N* are equal with *f* and 1 as weighting. Other combinations based on *N* did not give good results; see Table 3 (stop-words excluded, *best* sentence selection).

Finally, with the best combinations obtained from the first two experiments, we tried different sentence selection variants; see the last third of Table 2.

One can observe that any *k*best+first sentence selection option outperformed any combination that used the standard sentence selection scheme, with smaller *k* always giving better results—that is, only the slightest correction to the baseline improved it. The best result was obtained with single words derived from MFSs, with their weighting by the frequency of the corresponding MFS.

Table 3. Results for variants of the set *N* (options: *excluded*, *best*)

Terms	Term weighting	Recall	Precisio n	F-measure
N	f or 1	**0.43711**	**0.45099**	**0.44383**
	l	0.42911	0.44324	0.43594
$N \setminus W$	1	0.42009	0.43693	0.42823
	f	0.41849	0.43532	0.42662
$N \setminus M_2$	1	0.42315	0.43806	0.43035
$N \setminus (W \cup M_2)$		0.41084	0.42759	0.41893

7 Comparison

We compared the following results:

- State of the art: The author of [15] provided us with her data, which were evaluated in the same conditions as proposed methods. Specifically, DirectedBackward version of TextRank [15] was evaluated. We also list the results of the original TextRank with implementation of PageRank with DirectedBackward version of TextRank but with some additional data processing to remove noisy data [16] and the modified TextRank with a biased version of PageRank [17]. See details of the preprocessing in [15–17].
- Baseline: We denote *Baseline: first* the baseline mentioned in Section 2, which selects the first sentences of the text until the desired size of the summary is reached [12]. This baseline gives very good results on the kind of texts (news reports) that we experimented with, but would not give so good results on other types of texts. Thus we proposed another baseline, denoted *Baseline: random*, which selects random sentences; the results presented below are averaged by 10 runs. We believe this to be a more realistic baseline for the types of texts other than news reports.

– Our proposal: We compare these methods with the best results obtained with our proposal with the *best* and 1*best+first* sentence selection scheme, as shown in Table 2. In both cases our best results were obtained with the options *W* without stop-words for term selection and *f* for term weighting.

For fair comparison, we separated the methods by the type of information they used in addition to the weighting derived from terms:

– None (text is considered as a bag of sentences, sentence as a bag of terms, terms as strings),
– Order of sentences (say, first sentences are treated specially),
– Sophisticated pre-processing to obtain the terms.

We believe that in the future combination of these types of additional information can give even better results. The comparison is given in Table 4.

Table 4. Results with other methods

Additional info used	Method	Recall	Precision	F
None	Baseline: *random*	0.37892	0.39816	0.38817
	TextRank: [15]	**0.45220**	0.43487	0.44320
	Proposed: *W*, *f*, *best*	0.44609	**0.45953**	**0.45259**
Order of sentences	Baseline: *first*	0.46407	0.48240	0.47294
	Proposed: *W*, *f*, 1*best+first*	**0.46576**	**0.48278**	**0.47399**
Pre-processing	TextRank: [16]	0.46582	0.48382	0.47450
	TextRank: [17]	**<u>0.47207</u>**	**<u>0.48990</u>**	**<u>0.48068</u>**

We could not apply our method with the pre-processing option because we did not have access to the specific details of the pre-processing procedure used in [16] and [17]. However, in the other two categories our method outperformed the others. Possibly with the same type of pre-processing our method would outperform the others in the last category, too.

8 Discussion and Conclusions

We have tested different combinations of term selection, term weighting, and sentence selection options for language- and domain-independent extractive single-document text summarization on a news report collection.

We observed that words from repeating bigrams are good terms, and so are MFSs (we can speculate that MFSs are still better semantic units but splitting them into single words gives a more flexible and less sparse comparison). For term weighting, we observed that a good weighting scheme is the number of occurrences of the term in the text as part of a repeating bigram. With these settings, we obtained the results superior to the existing state-of-the-art methods.

Most of the state-of-the-art methods perform worse than the baseline method that takes into account a special ordering of sentences in news reports, which contain a nearly ready abstract in their first sentences. However, our methods can select one sentence better than this baseline method (while already the second-best sentence selected by our method proves to be worse than the baseline). This gives a hybrid method (one sentence our and then back-off to the baseline) superior to both the baseline and other state-of-the-art methods.

In our experiments we did not apply pre-processing that was shown to be beneficial for other methods, so our results are below those of other methods when they do apply it, though above them when they do not. The latter makes us believe that when we apply pre-processing we will obtain results superior to all existing methods. This will be the topic of our future work.

On the other hand, our experiments show that very different options (some of them rather absurd) only slightly affect the overall result, at least on the collection we used for our experiments. This can probably be explained by the nature of the texts in this collection (short news reports) and maybe by the behavior of the ROUGE evaluation scheme: the completely random selection baseline is rather high (so nearly any method would give at least similar results) while what seems to be almost top-line—selecting the first sentences of the text[3]—is quite low and quite near to the random baseline. This makes us rather pessimistic about much further progress in the results unless another data collection is used and probably better evaluation schemes are developed.

References

1. Lin, C.Y., Hovy, E.: Automated Text Summarization in SUMMARIST. In: Proc. of ACL Workshop on Intelligent, Scalable Text Summarization, Madrid, Spain (1997)
2. Kupiec, J., Pedersen, J.O., Chen, F.: A Trainable Document Summarizer. In: Proceedings of the 18th ACM-SIGIR Conference on Research and Development in Information Retrieval, Seattle, pp. 68–73 (1995)
3. Song, Y., et al.: A Term Weighting Method based on Lexical Chain for Automatic Summarization. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, Springer, Heidelberg (2006)
4. Cristea, D., et al.: Summarization through Discourse Structure. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, Springer, Heidelberg (2006)
5. Liu, D., et al.: Multi-Document Summarization Based on BE-Vector Clustering. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, Springer, Heidelberg (2006)
6. Xu, W., Li, W., et al.: Deriving Event Relevance from the Ontology Constructed with Formal Concept Analysis. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, Springer, Heidelberg (2006)
7. Villatoro-Tello, E., Villaseñor-Pineda, L., Montes-y-Gómez, M.: Using Word Sequences for Text Summarization. In: Sojka, P., Kopeček, I., Pala, K. (eds.) TSD 2006. LNCS (LNAI), vol. 4188, pp. 293–300. Springer, Heidelberg (2006)
8. Chuang, T.W., Yang, J.: Text Summarization by Sentence Segment Extraction Using Machine Learning Algorithms. In: Proc. of the ACL 2004 Workshop, Barcelona, España (2004)
9. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information processing & Management 24, 513–523 (1988)

[3] We believe this to be nearly top-line because the first lines of a news report are intended by its author to serve as a ready summary of the whole report, and are probably the best summary for it.

10. García-Hernández, R.A., Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A.: A Fast Algorithm to Find All the Maximal Frequent Sequences in a Text. In: Sanfeliu, A., Martínez Trinidad, J.F., Carrasco Ochoa, J.A. (eds.) CIARP 2004. LNCS, vol. 3287, pp. 478–486. Springer, Heidelberg (2004)
11. García-Hernández, R.A., Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A.: A New Algorithm for Fast Discovery of Maximal Sequential Patterns in a Document Collection. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 514–523. Springer, Heidelberg (2006)
12. DUC. Document understanding conference 2002 (2002), `http://www-nlpir.nist.gov/projects/duc`
13. Lin, C.Y.: ROUGE: A Package for Automatic Evaluation of Summaries. In: Proceedings of Workshop on Text Summarization of ACL, Spain (2004)
14. Lin, C.Y., Hovy, E.: Automatic Evaluation of Summaries Using N-gram Co-Occurrence Statistics. In: Proceedings of HLT-NAACL, Canada (2003)
15. Mihalcea, R.: Random Walks on Text Structures. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 249–262. Springer, Heidelberg (2006)
16. Mihalcea, R., Tarau, P.: TextRank: Bringing Order into Texts. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain (2004)
17. Hassan, S., Mihalcea, R., Banea, C.: Random-Walk Term Weighting for Improved Text Classification. In: Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2007), Irvine, CA (2007)
18. HaCohen-Kerner, Y., Zuriel, G., Asaf, M.: Automatic Extraction and Learning of Keyphrases from Scientific Articles. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878, pp. 645–657. Springer, Heidelberg (2006)
19. Bolshakov, I.A.: Getting One's First Million...Collocations. In: Gelbukh, A. (ed.) CICLing 2004. LNCS, vol. 2945, pp. 229–242. Springer, Heidelberg (2004)
20. Koster, C.H.A.: Transducing Text to Multiword Units. In: Workshop on Multiword Units MEMURA at the fourth International Conference on Language Resources and Evaluation, LREC-2004, Lisbon, Portugal (2004)
21. Baeza Yates, R., Ribeiro Neto, B.: Modern Information Retrieval. ACM Press, New York (1999)

Real-Word Spelling Correction with Trigrams: A Reconsideration of the Mays, Damerau, and Mercer Model

Amber Wilcox-O'Hearn, Graeme Hirst, and Alexander Budanitsky[⋆]

Department of Computer Science, University of Toronto
Toronto, Ontario, Canada M5S 3G4
{amber,gh,abm}@cs.toronto.edu

Abstract. The trigram-based noisy-channel model of real-word spelling-error correction that was presented by Mays, Damerau, and Mercer in 1991 has never been adequately evaluated or compared with other methods. We analyze the advantages and limitations of the method, and present a new evaluation that enables a meaningful comparison with the WordNet-based method of Hirst and Budanitsky. The trigram method is found to be superior, even on content words. We then show that optimizing over sentences gives better results than variants of the algorithm that optimize over fixed-length windows.

1 Introduction

Real-word spelling errors are words in a text that, although correctly spelled words in the dictionary, are not the words that the writer intended. Such errors may be caused by typing mistakes or by the writer's ignorance of the correct spelling of the intended word. Ironically, such errors are also caused by spelling checkers in the correction of non-word spelling errors: the "auto-correct" feature in popular word-processing software will sometimes silently change a non-word to the wrong real word (Hirst and Budanitsky 2005), and sometimes when correcting a flagged error, the user will inadvertently make the wrong selection from the alternatives offered. The problem that we address in this paper is the automatic detection and correction of real-word errors.

Methods developed in previous research on this topic fall into two basic categories: those based on human-made lexical or other resources and those based on machine-learning or statistical methods. An example of a resource-based method is that of Hirst and Budanitsky (2005), who use semantic distance measures in WordNet to detect words that are potentially anomalous in context — that is, semantically distant from nearby words; if a variation in spelling[1] results in a word that was semantically closer to the context, it is hypothesized that the original word is an error (a "*malapropism*")

[⋆] This research was supported by a grant from the Natural Sciences and Engineering Research Council of Canada. We are grateful to Bryce Wilcox-O'Hearn for his assistance.

[1] In this method, as in the trigram method that we discuss later, any consistent definition, narrow or broad, of what counts as the spelling variations of a word may be used. Typically it would be based on edit distance, and might also take phonetic similarity into account; see our remarks on Brill and Moore (2000) and Toutanova and Moore (2002) in section 5 below.

and the closer word is its correction. An example of a machine-learning method is that of Golding and Roth (1999), who combined the Winnow algorithm with weighted-majority voting, using nearby and adjacent words as features. (An extensive review of the prior research is given by Hirst and Budanitsky (2005), so we do not revisit it here. The problem of spelling correction more generally is reviewed by Kukich (1992).)

Typically, the machine learning and statistical approaches rely on pre-defined **confusion sets**, which are sets (usually pairs) of commonly confounded words, such as {*their, there, they're*} and {*principle, principal*}. The methods learn the characteristics of typical context for each member of the set and detect situations in which one member occurs in context that is more typical of another. Such methods, therefore, are inherently limited to a set of common, predefined errors, but such errors can include both content and function words. By contrast, the resource-based methods are not limited in this way, and can potentially detect a confounding of any two words listed in the resource that are spelling variations of one another, but these methods can operate only on errors in which both the error and the intended word are content words. The two methods are thus complementary; a complete system could use confusion sets to find common confounds and a resource-based method to look for other errors.

However, there is one method that is statistical and yet does not require predefined confusion sets: using word-trigram probabilities, which were first proposed for detecting and correcting real-word errors many years ago by Mays, Damerau, and Mercer (1991) (hereafter, *MDM*). Conceptually, the method is simple: if the trigram-derived probability of an observed sentence is lower than that of any sentence obtained by replacing one of the words with a spelling variation, then hypothesize that the original is an error and the variation is what the user intended.[2] In other words, relatively low probability of a sentence is taken as a proxy for semantic anomaly. Despite its apparent simplicity, the method has never, as far as we are aware, been applied in practice nor even used as a baseline in the evaluation of other methods. In this paper, we show why MDM's algorithm is more problematic than it at first seems, and why their published results cannot be used as a baseline. We present a new evaluation of the algorithm, designed so that the results can be compared with those of other methods, and then construct and evaluate some variations of the algorithm that use fixed-length windows.

2 The MDM Method and Its Characteristics

2.1 The Method

MDM frame real-word spelling correction as an instance of the noisy-channel problem: correcting the signal *S* (the observed sentence), which has passed through a noisy

[2] Trigram models have also been proposed for the simpler problem of correcting *non-word* spelling errors, most notably by Church and Gale (1991) and Brill and Moore (2000). Such models simply *presume* the presence of an error that has already been detected by another process (for example, by the failure of lexical look-up), and merely try to correct it within the trigram window. The real-word problem, by contrast, presumes the *absence* of an error, and the model is responsible not just for correcting errors but also for detecting them in the first place; this leads to considerations such as optimizing over sentence probabilities that have no counterpart in the simpler non-word trigram models. See also section 5 below.

channel (the typist) that might have introduced errors into it, by finding the most likely original signal S' (the intended sentence, generated by a language model). The probability that the typist types a word correctly is a parameter α, which is the same for all words.[3] A typical value for α could be .99. For each word, the remaining probability mass $(1-\alpha)$, the probability that the word is mistyped as another real word, is distributed equally among all its spelling variations.[4] So the probability that an intended word w is typed as x is given by

$$P(x\,|\,w) = \begin{cases} \alpha & \text{if } x = w \\ (1-\alpha)/|SV(w)| & \text{if } x \in SV(w) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $SV(w)$ is the set of spelling variations of word w (not including w itself).

The language model uses trigram probabilities; that is, the probability of an intended word w_i is given by $P(w_i\,|\,w_{i-1}w_{i-2})$, where $w_0 = w_{-1} = BoS$ (the beginning-of-sentence marker) and $w_{n+1} = w_{n+2} = EoS$ (the end-of-sentence marker). Thus the probability of an intended sentence $S' = w_1 \dots w_n$ is given by

$$P(S') = \prod_{i=1}^{n+2} P(w_i\,|\,w_{i-1}w_{i-2}). \tag{2}$$

So given an observed sentence S, the corrected sentence S' is the one in the search space $\mathscr{C}(S) \cup \{S\}$ that maximizes the probability $P(S'\,|\,S) \propto P(S') \cdot P(S\,|\,S')$, where $P(S\,|\,S')$ is given by the model of the noisy channel, i.e., the typist, and the set $\mathscr{C}(S)$ of candidate corrections is the set of all sentences in which exactly one word in S has been replaced by one of its real-word spelling variations.

2.2 Discussion of the Method

MDM's method has an advantage over the resource-based "open-ended" methods in being able to detect errors in both content words and function words. But it also has the complementary disadvantage that effort is spent on errors that would also be found by a grammar checker (which would presumably be included in any writer's-aid system of which the spelling checker were a part), rather than concentrating on the errors that could not be thus detected. Another disadvantage is the size of the trigram model; a model covering a usefully large vocabulary might be impractically large. Data sparseness is also a serious problem: many correct trigrams that are observed will not occur in the model, even if it is built from a very large corpus.

An undesirable property of the method is that the likelihood that a real-word error x will be corrected depends on the number of spelling variations of the intended word w: the larger $SV(w)$ is, the smaller $P(w\,|\,x)$ is and hence the smaller the chance of correction

[3] No mention is made of words mistyped as non-words; but we can regard α as the probability that the word is either typed correctly or is typed as a non-word and then correctly amended.

[4] MDM refer to this as the word's confusion set; but unlike the confusion sets of, e.g., Golding and Roth (1999), it includes all spelling variations, not just those selected by a human as likely confounds.

is. This is a consequence of the division of the constant probability mass $(1-\alpha)$ among all members of $SV(w)$ in equation 1.

Because each member of $\mathscr{C}(S)$ contains exactly one changed word, the method is unable to correct more than one error per sentence. (Including in $\mathscr{C}(S)$ sentences with more than one change would be combinatorially explosive; but see section 4.2 below.) This limitation would usually not be a problem; that is, we expect that for most typists, α is considerably greater than the reciprocal of the mean sentence length, and so sentences would only very rarely contain more than one real-word error. Nonetheless, MDM seemingly violate their own assumption by considering typists with α values as low as .9 (one word in every ten is a real-word error); see section 2.3 below.

2.3 The Limitations of MDM's Evaluation

MDM's evaluation of their method used trigram probabilities for a 20,000-word vocabulary; they do not say what corpus the probabilities were derived from,[5] nor what smoothing method, if any, was used.[6] The test set was only 100 sentences, containing no words outside the 20,000-word vocabulary, chosen from newswire and English Canadian Hansard. For each sentence, a set of erroneous sentences was generated by replacing each word in turn with each of its possible spelling variations in the vocabulary; that is, each erroneous sentence contained exactly one error. There was an average of 86 erroneous sentences S for each original sentence S'.

In each set of sentences, each erroneous sentence was tested to determine whether, if it were observed, some other sentence in the set would be preferred, and if so whether that would be the original sentence; in addition, each original sentence was tested to see whether some erroneous variation would be preferred. The experiments were carried out with four different values of α, from .9 (an extremely error-prone typist) to .9999 (an extraordinarily accurate typist).

MDM did not present their results in terms of per-word accuracy or precision and recall, nor did they give the data necessary to calculate these values (true and false positives), so it is not possible to compare their results with other methods, such as those of Golding and Roth (1999) or Hirst and Budanitsky (2005), for which data are so presented. They do not include data on sentence lengths, and moreover, they classify their results according to *(a)* whether an erroneous sentence was detected as such and, if so, whether the appropriate correction was made, and *(b)* whether an actually correct sentence was wrongly selected for change. Thus, erroneous sentences in which the method incorrectly changes a true positive are conflated with those in which it chooses a false positive and a false negative. Hence only *per-sentence* accuracy, precision, and recall, incommensurate with other methods, can be derived from MDM's data; but in any case such measures are meaningless because of the extreme artificiality and bias of the

[5] By a citation to Bahl, Jelinek, and Mercer (1983), MDM imply that the corpus they used was the IBM Laser Patent Corpus. But this cannot be so, as that corpus had a vocabulary of only 12,000 words (Bahl et al. 1978); and in any case trigram probabilities derived from such a corpus would be completely inappropriate for use with newswire and Hansard text.

[6] In their example data, MDM show the seemingly unlikely trigram *a submit that* as having a much higher probability than the trigram *what is happening*.

test set. With the original sentences outnumbered by erroneous sentences 86 to 1, the number of false positives that are possible is extremely small compared to the number of true positives, with the consequence that per-sentence precision exceeds .99 in all cases and per-sentence recall varies from .618 for a very high value of α to .744 for a low value. Moreover, a model that performs well for MDM's test data may actually be prone to overcorrection in real data, which would translate into a loss of precision. There may be additional unpredictable effects of this bias too.

3 Re-evaluating the MDM Method

Because of these problems, we re-implemented and re-evaluated the MDM method in order to be able to make direct comparisons with other methods. As the original MDM data are not available, we followed Hirst and Budanitsky (2005) in using the 1987–89 Wall Street Journal corpus (approximately 30 million words), which we presume to be essentially free of errors. We reserved 500 articles (approximately 300,000 words) to create test data (see below). With the remainder of the corpus, using the CMU–Cambridge Statistical Language Modeling Toolkit (Clarkson and Rosenfeld 1997), we created a trigram model whose vocabulary was the 20,000 most frequent words in the corpus; all other words were mapped to the token *OOV* ("out of vocabulary"). We incorporated standard tokenization, and the Good–Turing smoothing and Katz backoff techniques of the toolkit.

To create more-realistic test sets, we automatically inserted real-word errors in the reserved articles by replacing one word in approximately every 200 with a random spelling variation — that is, we modeled a typist whose α value is .995; we chose this value simply to match the density of errors used by Hirst and Budanitsky (2005). And like both those authors and MDM, we defined a spelling variation to be a single-character insertion, deletion, or replacement, or the transposition of two characters that results in another real word. We created three test sets, each containing 15,555 sentences, which varied according to which words were candidates for replacement and for substitution:

T20: Any word in the 20,000-word vocabulary of the trigram model could be replaced by a spelling variation from the same vocabulary; this replicates MDM's style of test set.

T62: Any word in the 62,000 most frequent words in the corpus could be replaced by a spelling variation from the same vocabulary; this reflects real typing errors much better than **T20**.

Mal: Any content word listed as a noun in WordNet (but regardless of whether it was used as a noun in the text; there was no syntactic analysis) could be replaced by any spelling variation found in the lexicon of the *ispell* spelling checker; this replicates Hirst and Budanitsky's "malapropism" data.

Observe that in **T62** and **Mal**, the errors (the replacement words) are not limited to the vocabulary of the model. Thus one factor in our re-evaluation of the method is the adequacy of a 20,000-word vocabulary in the face of more-realistic data.

Table 1. Results of our replication of the MDM method on Wall Street Journal data with a 20,000-word vocabulary on three different test sets (see text for description), and the results of Hirst and Budanitsky (2005) on similar data (last row)

	Detection			Correction		
α	P	R	F	P	R	F
Test set **T20**:						
.9	.334	.847	.479	.327	.818	.467
.99	.574	.768	.657	.567	.747	.645
.995	.646	.736	.688	.639	.716	.675
.999	.794	.658	.719	.790	.643	.709
Test set **T62:**						
.9	.235	.537	.327	.229	.519	.318
.99	.447	.478	.462	.441	.466	.453
.995	.523	.460	.490	.517	.450	.481
.999	.693	.400	.508	.690	.395	.502
Test set **Mal**:						
.9	.145	.367	.208	.140	.352	.200
.99	.306	.320	.313	.299	.310	.304
.995	.371	.304	.334	.365	.296	.327
.999	.546	.261	.353	.543	.257	.349
Hirst and Budanitsky's best results (on **Mal**):						
–	.225	.306	.260	.207	.281	.238

We ran our re-implementation of the MDM method with this data. Only test-data words that were in the 20,000-word vocabulary were candidates for correction, and words outside the vocabulary were mapped to *OOV* when determining trigram probabilities. We used four values of α, from .9 to .999, including the .995 value of the "typist" of our test data. We computed results in terms of per-word precision, recall, and F-measure, which we show separately for detection of an error and correction of an error; see Table 1.

The performance of the method is quite impressive. On the **T20** test set (all errors are in the vocabulary of the model) at $\alpha = .995$, which is perhaps the most realistic level, correction recall (the fraction of errors correctly amended) is .716 and correction precision (the fraction of amendments that are correct) is .639 ($F = .675$). On the **T62** test set (errors are not limited to the vocabulary of the model), performance naturally drops, but correction recall and precision are .450 and .517, respectively ($F = .481$), which is a level that would still be helpful to a user. Some examples of successful and unsuccessful corrections are shown in Table 2.

On the malapropism test set (all errors are in content words), the results are poorer; at $\alpha = .995$, correction recall is .296 and correction precision is .365 ($F = .327$). The difference between these results and those on **T62** shows that MDM's method performs better on function-word errors than on content-word errors. This is not surprising;

Table 2. Examples of successful and unsuccessful corrections. Italics indicate observed word, arrow indicates correction, square brackets indicate intended word.

SUCCESSFUL CORRECTION:

Exxon has made a *loot* → lot [lot] of acquisitions of smaller properties, though the pace slowed last year after oil prices fell.

FALSE POSITIVE:

... Texaco's creditors *would* → could [would] breathe a sigh of relief ...

... the Conservative Party ... has been *last* → lost [last] in political polls.

FALSE NEGATIVE:

Like many schools, Lee's prospective kindergarten uses a readiness *teat* [test], designed to screen out children considered too immature.

TRUE POSITIVE DETECTION, FALSE POSITIVE CORRECTION:

"I'm uncomfortable *tacking* → talking [taking] a lot of time off work," he says.

intuitively, function-word errors are more likely to result in syntactic ill-formedness, and hence a much lower probability sentence, than the content-word errors. Nonetheless, these results are noticeably better than the best results of Hirst and Budanitsky's WordNet-based method, which achieved $F = .238$ on very similar data (last row of Table 1); in particular, the MDM method has superior correction precision.

4 Variations and Attempted Improvements on the MDM Method

4.1 A Better Language Model

Although MDM's method already does well compared to Hirst and Budanitsky's method, it is clear that it can be improved further. One obvious improvement is to increase the size of the language model. Table 3 shows that a 62,000-word model results in a large improvement over the 20,000-word model; for example, at $\alpha = .995$, correction F increases by 43% on test set **T62** and 45% on **Mal**. (Results on **T20** are roughly the same as before, of course; the slight reduction in performance is primarily due to the greater number of spelling variations that many words now have in the model.) The cost of the improvement is an increase in the size of the model from 17.9 million trigrams to 20.8 million. (Despite the exponential increase in the space of trigrams, the number actually observed in the corpus grows quite mildly.) Because of these results, we drop the 20,000-word model (and the **T20** test set) from further consideration.

4.2 Permitting Multiple Corrections

As we noted in section 2.2, the MDM algorithm can make at most one correction per sentence, because it would be combinatorially explosive to include sentences with more than one correction in the set $\mathscr{C}(S)$ of possible corrections of sentence S. We also noted that such an ability would, in any case, be of use only to very unskilled typists. Nonetheless, for the benefit of such typists, a possible method of making multiple corrections

Table 3. Results of our replication of the MDM method on Wall Street Journal data with a 62,000-word vocabulary on three different test sets

	Detection			Correction		
α	P	R	F	P	R	F
Test set **T20**:						
.9	.318	.828	.460	.311	.801	.448
.99	.532	.742	.619	.525	.724	.609
.995	.592	.708	.645	.587	.691	.635
.999	.738	.627	.678	.734	.614	.669
Test set **T62**:						
.9	.325	.846	.469	.318	.820	.458
.99	.544	.774	.639	.538	.758	.629
.995	.608	.750	.672	.603	.736	.663
.999	.756	.678	.715	.753	.667	.707
Test set **Mal**:						
.9	.212	.596	.313	.205	.571	.302
.99	.398	.536	.457	.390	.519	.445
.995	.459	.510	.483	.453	.497	.474
.999	.620	.444	.517	.616	.436	.510

Table 4. Results of the method permitting multiple corrections in the same sentence

	Detection			Correction		
α	P	R	F	P	R	F
Test set **T62**:						
.9	.270	.869	.411	.263	.840	.400
.99	.505	.783	.614	.499	.765	.604
.995	.578	.756	.655	.573	.740	.646
.999	.739	.680	.708	.736	.668	.701
Test set **Mal**:						
.9	.179	.614	.277	.172	.586	.266
.99	.372	.543	.442	.364	.525	.430
.995	.437	.515	.473	.431	.502	.464
.999	.610	.448	.516	.605	.440	.510

in a sentence while avoiding a combinatorial explosion is this: Instead of choosing the single sentence $S' \in \mathscr{C}(S) \cup S$ that maximizes the probability $P(S' \mid S)$, choose *all* sentences that give a probability exceeding that given by S itself, and then combine the corrections that each such sentence implies. (If conflicting corrections are implied then the one with the highest probability is chosen.) In other words, we apply all corrections that, taken individually, would raise the probability of the sentence as a whole, rather than only the single most probable such correction. It is important to note, however, the price that is paid here for avoiding the complete search space: The sentence that results from the combination of corrections might have a lower probability than others with fewer corrections — possibly even lower than that of the original sentence.

We experimented with this method using the 62,000-word model of section 4.1. We expected that the method would lead to improved correction only in the poor-typist condition where $\alpha = .9$ (one word in ten is mistyped). The results are shown in Table 4. Contrary to our expectations, despite an increase in recall compared to Table 3, F values were distinctly poorer for all values of α, *especially* the lower values, because the number of false positives went up greatly and hence precision dropped markedly. The number of sentences in which multiple corrections were hypothesized far exceeded the number of sentences with multiple errors; even for $\alpha = .9$ there were actually very few such sentences in the test data.

4.3 Using Fixed-Length Windows

The MDM method optimizes over sentences, which are variable-length and potentially quite long units. It is natural, therefore, to ask how performance changes if shorter, fixed-length units are used. In particular, what happens if we optimize a single word at a time in its trigram context? In this section, we consider a variation of the method that optimizes over relatively short, fixed-length windows instead of over a whole sentence (except in the special case that the sentence is smaller than the window), while respecting sentence boundaries as natural breakpoints. To check the spelling of a span of d words requires a window of length $d+4$ to accommodate all the trigrams that overlap with the words in the span. The smallest possible window is therefore 5 words long, which uses 3 trigrams to optimize only its middle word.

Assume as before that the sentence is bracketed by two *BoS* and two *EoS* markers (to accommodate trigrams involving the first two and last two words of the sentence). The window starts with its left-hand edge at the first *BoS* marker, and the MDM method is run on the words covered by the trigrams that it contains; the window then moves d words to the right and the process repeats until all the words in the sentence have been checked.[7]

Observe that because the MDM algorithm is run separately in each window, potentially changing a word in each, this method as a side-effect also permits multiple corrections in a single sentence. In contrast to the method of section 4.2 above, the combinatorial explosion is avoided here by the segmentation of the sentence into smaller windows and the remaining limitation of no more than one correction per window. This limitation evaporates when $d = 1$, and the method becomes equivalent in its effect to that of section 4.2.

[7] If the number of words in the sentence is not an exact multiple of d, and the final window would contain no more than $d/2$ words, some preceding windows are enlarged to distribute these extra words; if the final window would contain more than $d/2$ but fewer than d words, then some preceding windows are reduced to distribute the extra space. For example, if $d = 5$ and the sentence is 22 words long, then the lengths of the windows are 6,6,5,5; if the sentence is 18 words long, then they will be 5,5,4,4.

Table 5. Results of adapting the MDM method to a fixed window of size $d+4$ that corrects d words

	Detection			Correction		
α	P	R	F	P	R	F
Test set **T62**, $d = 3$:						
.9	.275	.867	.418	.269	.838	.407
.99	.507	.783	.615	.501	.765	.605
.995	.579	.756	.656	.574	.740	.646
.999	.740	.680	.709	.737	.668	.701
Test set **Mal**, $d = 3$:						
.9	.184	.614	.283	.177	.586	.272
.99	.373	.543	.442	.366	.525	.431
.995	.439	.515	.474	.432	.502	.465
.999	.611	.448	.517	.607	.440	.510

	Detection			Correction		
α	P	R	F	P	R	F
Test set **T62**, $d = 6$:						
.9	.283	.864	.426	.276	.835	.415
.99	.512	.780	.618	.507	.762	.608
.995	.584	.755	.659	.579	.739	.649
.999	.743	.679	.710	.740	.668	.702
Test set **Mal**, $d = 6$:						
.9	.188	.610	.287	.181	.583	.276
.99	.377	.541	.445	.370	.523	.433
.995	.442	.513	.475	.436	.500	.466
.999	.612	.446	.516	.607	.438	.509

	Detection			Correction		
α	P	R	F	P	R	F
Test set **T62**, $d = 10$:						
.9	.292	.860	.436	.285	.832	.425
.99	.521	.780	.625	.515	.762	.615
.995	.593	.755	.664	.588	.739	.655
.999	.747	.679	.711	.744	.667	.703
Test set **Mal**, $d = 10$:						
.9	.193	.609	.293	.186	.581	.282
.99	.384	.541	.449	.376	.524	.438
.995	.448	.514	.479	.442	.501	.470
.999	.614	.447	.518	.610	.439	.511

This, in turn, suggests a variation in which the window slides across the sentence, moving one word to the right at each iteration, overlapping its previous position, and then checking the words it contains in its new position. This would permit unrestricted multiple corrections for values of d larger than 1, but at the price of rather more computation: If the sentence length is l words (plus the *BoS* and *EoS* markers), then $l-d+1$ iterations will be required to check the complete sentence instead of just $\lceil l/d \rceil$.[8]

We experimented with these methods for d = 3, 6, and 10, with the 62,000-word model. (We also tried $d = 1$, and verified that the results were identical to those of

[8] Some additional complexities arise in this method from the overlapping of the positions that the window takes. Except for the case when $d = 1$ (where this method becomes identical to the simple fixed-window method), words will be candidates for change in more than one window, with possibly conflicting results. We took a very simple approach: we never changed words in the middle of the analysis, and the opinion of the rightmost window always prevailed. For a discussion of the issues, see Wilcox-O'Hearn (2008).

Table 4.) The performance of the simple fixed-window method is shown in Table 5. We observe that in most conditions, as with our first approach to multiple corrections, this method increases recall somewhat compared to the whole-sentence model (Table 3), but precision drops markedly, especially for lower values of d and α, resulting in F values that are mostly poorer than, and at best about the same as, those of the whole-sentence model. Results are not shown for the sliding-window variation, whose performance in all conditions was the same as, or poorer than, the simpler method. We conclude that taking a unit of analysis smaller than the sentence is deleterious to the MDM method.

5 Related Work

As noted in footnote 2 above, noisy-channel trigram models have also been used in the simpler problem of non-word spelling correction. The emphasis in this work has generally been on the development of better channel models, i.e., better models of the typist. For example, at the level of keyboard errors, a substitution error involving keys that are adjacent on the keyboard is more likely than one involving two random keys; Church and Gale (1991) use complete character-based confusion matrices of typing errors. At the level of cognitive errors, the substitution of, for example, *a* for *e* is more likely (in English) in the context of *-ent* at the end of a word; Brill and Moore (2000) develop a model that accounts for this, which Toutanova and Moore (2002) extend to include phonetic similarity. Clearly, these channel models could also be used as the model of the typist in the MDM method; in equation (1), the probability mass $(1-\alpha)$ would be distributed among the spelling variations not equally but in accordance with their relative likelihood as given by the new model. We intend to do this in future work (Wilcox-O'Hearn 2008). However, such models will not account for errors introduced by miscorrection of non-word errors, for which our present equal-probability assumption is a better model.

The only other trigram-based method that we are aware of for real-word errors is that of Verberne (2002), who does not use (explicit) probabilities nor even localize the possible error to a specific word. Rather, her method simply assumes that any word trigram in the text that is attested in the British National Corpus (without regard to sentence boundaries!) is correct, and any unattested trigram is a likely error; when an unattested trigram is observed, the method then tries the spelling variations of all words in the trigram to find attested trigrams to present to the user as possible corrections. Her evaluation was carried out on only 7100 words of the Wall Street Journal corpus, with 31 errors introduced (i.e., a density of one error in every approximately 200 words, the same as used by Hirst and Budanitsky and the present study); she obtained a recall of .33 for correction and a precision of just .05 ($F = .086$).[9]

Since we began this research, Microsoft has released Office Word 2007, which includes a "contextual spelling checker" capable of detecting a number of real-word

[9] Verberne also tested her method on 5500 words of the BNC with 606 errors introduced (an average density of one word in nine) by inserting all possible instances from a pre-compiled list of 134 error types; this achieved correction recall of .68 and precision of .98. But this was a subset of her training data and the error density is quite unrealistic, so the results are not meaningful.

errors; the underlying method is proprietary and not disclosed. In future work, we will evaluate this system in comparison with the MDM model. An informal preliminary evaluation, with 5000 words of our **Mal** test data containing 25 errors, found a trade-off of low recall for high precision: Word 2007 found just 4 of the 25 errors and marked a fifth (*cation* for *nation*) as a non-word error, but it made no false-positive errors ($R = 0.2, P = 1.0, F = 0.33$).

6 Conclusion

We have shown that the trigram-based real-word spelling-correction method of Mays, Damerau, and Mercer is superior in performance to the WordNet-based method of Hirst and Budanitsky, even on content words ("malapropisms") — especially when supplied with a realistically large trigram model. Our attempts to improve the method with smaller windows and with multiple corrections per sentence were not successful. Rather, we found that there is little need for multiple corrections; indeed, the constraint of allowing at most one correction per sentence is useful in preventing false positives.

References

Bahl, L.R., et al.: Recognition of a continuously read natural corpus. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1978), Tulsa, vol. 3, pp. 422–424 (1978)

Bahl, L.R., Jelinek, F., Mercer, R.L.: A maximum likelihood approach to continuous speech recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 5(2), 179–190 (1983)

Brill, E., Moore, R.C.: An improved error model for noisy channel spelling correction. In: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, pp. 286–293 (2000)

Church, K.W., Gale, W.A.: Probability scoring for spelling correction. Statistics and Computing 1, 93–103 (1991)

Clarkson, P., Rosenfeld, R.: Statistical language modeling using the CMU–Cambridge Toolkit. In: Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech), Rhodes, pp. 2707–2710 (1997)

Golding, A.R., Roth, D.: A Winnow-based approach to context-sensitive spelling correction. Machine Learning 34(1–3), 107–130 (1999)

Hirst, G., Budanitsky, A.: Correcting real-word spelling errors by restoring lexical cohesion. Natural Language Engineering 11(1), 87–111 (2005)

Kukich, K.: Techniques for automatically correcting words in text. Computing Surveys 24(4), 377–439 (1992)

Mays, E., Damerau, F.J., Mercer, R.L.: Context based spelling correction. Information Processing and Management 23(5), 517–522 (1991)

Toutanova, K., Moore, R.C.: Pronunciation modeling for improved spelling correction. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, pp. 144–151 (2002)

Verberne, S.: Context-sensitive spell [sic] checking based on trigram probabilities. Master's thesis, University of Nijmegen (2002)

Wilcox-O'Hearn, L.A.: Applying trigram models to real-word spelling correction. MSc thesis, Department of Computer Science, University of Toronto (forthcoming, 2008)

Non-interactive OCR Post-correction for Giga-Scale Digitization Projects

Martin Reynaert

Induction of Linguistic Knowledge, Tilburg University, The Netherlands

Abstract. This paper proposes a non-interactive system for reducing the level of OCR-induced typographical variation in large text collections, contemporary and historical. Text-Induced Corpus Clean-up or TICCL (pronounce 'tickle') focuses on high-frequency words derived from the corpus to be cleaned and gathers all typographical variants for any particular focus word that lie within the predefined Levenshtein distance (henceforth: LD). Simple text-induced filtering techniques help to retain as many as possible of the true positives and to discard as many as possible of the false positives. TICCL has been evaluated on a contemporary OCR-ed Dutch text corpus and on a corpus of historical newspaper articles, whose OCR-quality is far lower and which is in an older Dutch spelling. Representative samples of typographical variants from both corpora have allowed us not only to properly evaluate our system, but also to draw effective conclusions towards the adaptation of the adopted correction mechanism to OCR-error resolution. The performance scores obtained up to LD 2 mean that the bulk of undesirable OCR-induced typographical variation present can fully automatically be removed.

1 Introduction

This paper reports on efforts to reduce the massive amounts of non-word word forms created by OCRing large collections of printed text in order to bring down the type-token ratios of the collections to the levels observed in contemporary 'born-digital' collections of text. We report on post-correction of OCR-errors in large corpora of the Cultural Heritage. On invitation by the National Library of The Netherlands (Koninklijke Bibliotheek - Den Haag) we have worked on contemporary and historical text collections. The contemporary collection comprises the published Acts of Parliament (1989-1995) of The Netherlands, referred to as 'Staten-Generaal Digitaal' (henceforth: SGD)[1]. The historical collection is referred to as 'Database Digital Daily Newspapers' (henceforth: DDD)[2], which comprises a selection of daily newspapers published between 1918 and 1946 in the Netherlands. The historical collection was written in the Dutch spelling 'De

[1] URL: http://www.statengeneraaldigitaal.nl/

[2] URL: http://kranten.kb.nl/ In actual fact, this collection represents the result of a pilot project which is to be incorporated into the far more comprehensive DDD.

Vries-Te Winkel', which in 1954 was replaced by the more contemporary spelling used in the SGD. Both collections should be seen as pilot projects for extensive digitization projects underway in which the full newspaper col lection present in the National Library will be made publicly available online in the course of the next few years. A nice consequence of the fact that both collections we have worked on here are already available online is that any example given in this paper can be independently verified. If we claim that the English word 'restoring' is in fact an OCR-misrecognition of the word 'regeering' (i.e. De Vries-Te Winkel spelling for the contemporary word 'regering' ('government'), any reader can look it up on the DDD website and see the actual word highlighted in yellow in the digital image from which the OCRed text version was created by a company, using the OCR-software Abbyy FineReader version 6.

In the next Section we discuss the nature of lexical errors in large text collections. In Section 3 we present previous work in relation to the aims of this paper and in Section 4 we introduce our approach based on anagram hashing. Section 5 deals with the evaluation. We conclude in Section 6.

2 OCR-Errors and other Lexical Variation in Corpora

Commercially available Optical Character Recognition (OCR) systems boast high accuracy these days. Given that a system reaches 99% word accuracy, it should nevertheless not be lost from view that this in fact means that one word will have been misrecognized out of every hundred words processed.

Fig. 1 shows the Vocabulary Growth Curves (VGCs) [1] obtained with the zipfR package due to [2] for the three text collections we deal with in this paper. VGCs show how many new words are seen the more text is read. The attendant growth lines for hapax legomena are also plotted. TWC02 is a contemporary one-year newspaper corpus, covering the year 2002, composed mainly of 5 national Dutch newspapers. We use the corpus here as a kind of reference for the 'normal' vocabulary growth one would expect to see. Its curve at first shows a greater vocabulary growth than the SGD. This is easily explained by the fact that in newpapers far more topics are touched upon than in a typical debate in Parliament, which is likely centered around far fewer topics, calling upon a smaller vocabulary. At around 60M words the vocabulary growth of the TWC02 starts to slow down, and the SGD vocabulary continues to rise. We take this to be the effect of OCR-errors within the SGD. In sharp contrast to both these curves, the VGC of 'Het Volk', one of the newspapers in the DDD, exhibits a markedly sharp ascent. We list more statistics on the corpora in Table 1. Note the tremendous type-token ratio observed for 'Het Volk' (articles - 1918).

As concerns hapax legomena, words observed only once in a particular corpus, represented in the plot by the finer lines mimicking the lines representing the full vocabulary, the TWC02 displays a fairly normal rate of 44,2% overall. Around 50% of words in text are typically expected to be hapax legomena [3] (p.199). The SGD has a slightly higher rate at 55,2% overall, but again 'Het Volk' tops with a rate of hapaxes at 86,4% overall. This exceedingly high ratio of hapaxes

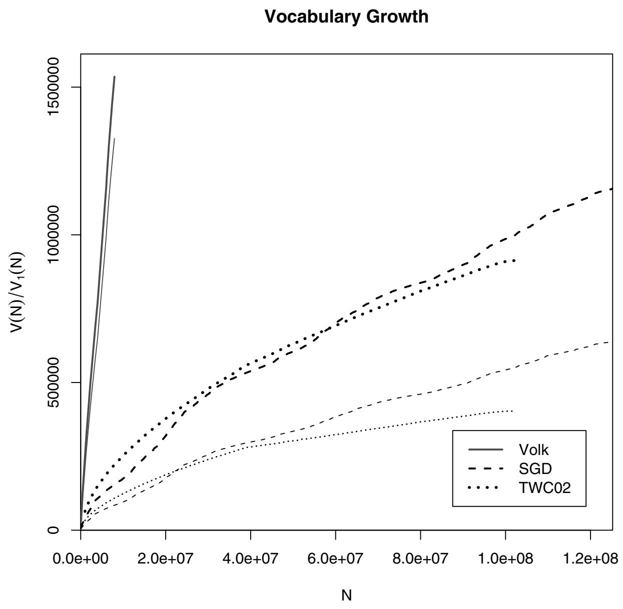


Fig. 1. Vocabulary Growth Curves for the contemporary reference corpus Twente Newspaper Corpus 2002 or TWC02, for the contemporary OCRed parliamentary debates corpus or SGD and the historical newspaper corpus 'het Volk, articles, 1918'. Attendant finer lines depict the growth in terms of the hapax legomena.

to types in 'Het Volk' shows that OCR-errors, which are often considered to be highly systematic, are anything but. High systematicity in particular confusions is present, allowing for some variants to accrue high token frequencies. On the whole, however, just about anything may happen, although less frequently. To illustrate this, Table 2 lists twenty variants for 'regeering' which differ only in the last character, the substitutions observed covering nearly the full alphabet.

[4] presented statistics on large collections of typographical errors (English: 12,094 typos and Dutch: 9,152) collected in large corpora. The bulk of the errors there were shown to be single-character insertions, deletions and substitutions, in line with the findings of previous studies, the largest of which was [5]. In Table 3 and Table 4 we list comparable statistics obtained from the OCRed corpora we here work with: statistics on 5,047 mainly OCR-errors from the SGD and 3,799 from the DDD. For the SGD we focused on the word 'belasting' ('tax'), a common topic in parliamentary debate, and strove to identify all variants in all morphological and compound forms of the word. In all, we identified 1,577 variants for the various guises of the noun 'belasting'. For the DDD we opted to identify all the variants for the lemma 'regeering', i.e. 'government'. This lemma yielded 1,468 variants in a single newspaper in the DDD, the 1918 edition of 'Het Volk' alone.

Table 1. Corpora Statistics: Corpus, language (CD: Contemporary Dutch, HD: Historical Dutch), origin: born-digital (BD) or OCRed (OCR), number of word tokens, number of word types, type-token ratio (TTR)

Corpus	Lang.	Origin	Tokens	Types	TTR
TWC2	CD	BD	92,793,519	914,026	0.985%
SGD	CD	OCR	125,209,007	1,156,998	0.924%
DDD	HD	OCR	7,950,950	1,535,529	19.31%

Table 2. Twenty variants (multiple non-contiguous errors) for the focus word 'regeering' produced by apparent random substitutions of the focus word's last character(s), besides the recurring substitution of an 'e' by 'c'

regecrin	regecrinc	regecring'	regecrinj	regecrino	regecrins	regecrinz
regecrin-	regecrincr	regecrini	regecrink	regecrinp	regecrint	regecrinü
regecrina	regecrinf	regecrinic	regecrinn	regecrinr	regecrinx	

Table 3. SGD 1989-1995: overview and statistics per LD of error-types encountered in a sample of 5,047 non-word variants

Category	LD 1	LD 2	LD 3	LD 4	LD 5	LD 6	LD 7	Total	%
deletion	221	10	3	1				235	4.66
insertion	1,980	27	6	11				2,024	**40.10**
substition	1,065	49	37	3		1		1,155	**22.89**
transposition		26						26	0.52
multi-C		722	30	10	1	1		779	**15.46**
multi-NC		303	271	101	22	5	2	710	**14.09**
run-on words	67							67	1.33
split word	32							32	0.63
TOTAL	3,380	1,138	347	126	23	7	2	5,047	
%	66.98	22.55	6.88	2.50	0.46	0.14	0.04		100.00

Table 4. DDD 'Het Volk' 1918: overview and statistics per LD of error-types encountered in a sample of 3,799 non-word variants

Category	LD 1	LD 2	LD 3	LD 4	LD 5	LD 6	Total	%
deletion	31	27	1	12			71	1.87
insertion	133	25	3	4			165	4.34
substition	575	276	109	2			962	**25.32**
transposition		3					3	0.08
multi-C		203	193	9	2	1	412	**10.85**
multi-NC		810	1,277	77	15	3	2,182	**57.44**
run-on words	2						2	0.05
split word	2						2	0.05
TOTAL	743	1,344	1,583	104	17	4	3,799	
%	19.56	35.38	41.67	2.74	0.45	0.11		100.0

The statistics clearly show the qualitative differences between typewritten, i.e. born-digital, and OCRed text collections. The bulk of the errors found in the latter are substitutions and multiple errors. A multiple error cannot be described by reference to just one of the 4 categories of error, i.e. either to insertion, deletion, transposition or substitution [6], alone. A multiple contiguous error (multi-C) would be the OCR-error 'regeermg' for 'regeering' (6 hits on the DDD website[3], i.e. the multiple error consisting of deletion of an 'i' and substitution of the 'n' by 'm' is situated in one location within the word. A multiple non-contiguous error (multi-NC) would be the OCR-error 'rcgecring' for 'regeering' (243 hits). These statistics further show us what is expected from a system geared at correcting OCR-errors. While the smaller edits required for typewritten text are present, an OCR-oriented correction mechanism has to be able to deal with greater edits, often located in several locations within a particular word string.

3 Related Work

3.1 Historical Text Collections: Prior Work

Most approaches to historical spelling variation for correction and/or text retrieval attempt to model the historical typographical variation observed by means of rules, either created manually or derived (semi-)automatically ([7] for English, [8], [9], [10] for German and [11] for Dutch). These authors typically work on older, pre-standardization era language variants than we do here. The spelling De Vries-Te Winkel can be seen as one of the first attempts at spelling standardization for Dutch. In this prior work very little attention, if any, is devoted to the typographical variation due to the OCR-process. This is either because the corpus was rekeyed or because the text underwent thorough editing after OCRing. [10] mention that OCR-errors are a problem, but do not address it. The work of [7] discusses adapting Aspell for work with 18th century English. Other interactive systems for post-correction of OCRed texts are described in [12], which was based on Ispell and [13], who describe an elaborate system for training an interactive OCR post-correction tool.

The digitization project underway at the National Library comprises an estimated 8 million pages of newspaper text, good for an estimated 25 billion words of running text. The sheer scale of digitization envisaged there and by similar institutions around the globe precludes even considering interactive post-correction of the OCRed corpora.

3.2 Large Dictionary Word Variant Retrieval: Prior Work

[14] propose a Universal Levenshtein Automata based approach to fast approximate search in large dictionaries. We note they evaluate exclusively on average retrieval times of the sets of correction candidates per length class of focus words. The system is claimed to perform exhaustive variant retrieval up to LD 3, but

[3] URL: http://kranten.kb.nl/index4.html (deselect: 'Het Centrum' and 'De NRC')

this is not supported by evaluation. The method is shown to be largely language-independent, but requires training and is sensitive to the size of the dictionary and the number of correction candidates, i.e. variants within the LD limit (the 'bound' in their terms), present. Given that the authors work with randomly introduced errors for all three languages, an assessment of the system's performance on both recall and precision would have been straightforward: if one considers the original list to be the gold standard and one introduces errors, one has a benchmark set against which to gauge one's system. We note the omission with wonder. This is remedied to some extent in [15] where recall scores are in fact given. The system was extended to return only restricted candidate sets. The results suggests that in fact the system does not exhaustively return all the correction candidates. This is particularly evident in the scores on LD 1, which shows the desirability of evaluating one's system not only globally, but also looking more specifically at specific ranges, here sets of variants sharing the same LD to their canonical form. Testing on lists containing only erroneous word forms, they could not perform an evaluation in terms of precision, although this is not mentioned. [16] apply the system to classify English and German webpages according to the level of lexical errors they contain. The paper discusses in full how the error dictionaries that are used for this purpose are built. We note that the system is strictly limited to LD 3. Further that by their own calculations over one third of the actual errors present in webpages is not detected: on lists of 1,000 real-world errors collected from webpages, over 62% for English and over 63% for German are not accounted for by the error dictionaries.

4 OCR Post-correction

4.1 Preliminaries

In contrast to the above cited recent work on historical text collections, we here do not focus almost exclusively on the historical variation, but almost exclusively on the variation caused by the OCR-process. This is because the spelling variation due to historical change is almost negligible compared to the OCR-variation in the DDD.

The system we propose, TICCL, is an unsupervised, scalable, fully automatic solution which requires no training (apart from some unavoidable text preprocessing) and which is largely language-independent. In fact, this approach should work for most alphabetical languages. TICCL does not try to account for unknown word types, i.e. words not present in the lexicon, in terms of their likely in-vocabulary counterpart. Rather, it tries to exhaustively gather all the likely non-word variants for a given known or, if unknown, given high frequency word type. This entails the system does not operate on the tokens in running text, but rather on the word type list derived from the corpus to be post-corrected. The system can be run with or without an extra validated word lexicon for the language. If no validated word lexicon is available, a word type list derived from a background corpus for the language may be used instead. The work by [17] is relevant in this respect because it shows that existing OCR-systems can be used to bootstrap resources for languages for which no customised OCR-systems exist or for which resources are scarce. Historical Dutch text can today be seen as an under-resourced language. For the historical spelling we work with here, we do not have a validated lexicon at our disposal nor was the OCR-system used adapted for older variants of Dutch.

4.2 Anagram Hashing

We propose an adaptation of the core correction algorithm we have described in depth in [18]. Anagram Hashing first uses a bad hashing function to identify all word strings in the corpus at hand that consist of the same subset of characters and assigns a large natural number to them, to be used as an index. Informally, the numerical value for a word string is obtained by summing the ISO Latin-1 code value of each character in the string raised to a power n, where n is empirically set at: 5. In effect, all anagrams, words consisting of a particular set of characters and present in the list, will be identified through their common numerical value. As the collisions produced by this function identify anagrams, we refer to this as an **anagram hash** and to the numerical values obtained as the **anagram values** (henceforth: AVs) and **anagram keys**, when we discuss these in relation to the hash. Based on a word form's anagram key it thus becomes possible to systematically query the list for any variants present, be they morphological, historical, typographical or orthographical.

The 'alphabet' used when we allow the system to search up to three (and more) character edits, contains the AVs for single characters and all possible two-character and three-character combinations. We call this the AnagramValueAlphabet. Note that a single value in this alphabet represents a single character or any combination of two (2 combinations) or three (6 combinations) particular characters, allowing for efficient look-up. The focus word is not likely to contain all the characters in the AnagramValueAlphabet. The subset of values from the AnagramValueAlphabet derivable from the characters actually present in the focus word forms the FocusWordAlphabet.

The lexicon is a regular hash built up at run-time having the AVs as keys and chained anagrams as values. We use the AV for the focus word and the FocusWordAlphabet and AnagramValueAlphabet to query the lexicon for variants of the focus word. These variants can all be seen as variations and combinations of the usual error type taxonomy. In our implementation, all four edit operations are handled as substitutions. For **substitutions**, a value from the FocusWordAlphabet is subtracted and a value from the AnagramValueAlphabet added. A single query on the AV for 'regeering' minus the AV for an 'e', plus the AV for a 'c' may thus retrieve the three OCR-errors: 'rcgeering', 'regcering' and 'regecring'. **Insertions** are substitutions where a value from the FocusWordAlphabet is subtracted and zero added. **Deletions** are substitutions where zero is subtracted and a value from the AnagramValueAlphabet added. To find **transposition** errors nothing needs to be added or subtracted.

By systematically querying the lexicon hash we retrieve all possible variants that fall within reach. The actual reach is defined by the alphabet used, i.e. depends on whether the alphabet contains the AVs for single characters only, or also for the full set of character bigrams, or even trigrams. The actual number of hash look-ups required is defined by the number of unique values in the AnagramValueAlphabet and by the number of unique values for all the character combinations in the focus word.

The anagram-based core correction system proposed by us in [18] was here extended on the basis of conclusions we were allowed to draw concerning the nature of OCR-errors on the basis of the statistics gathered from lexical variants culled from the SGD and DDD. In fact, only relatively minor extensions proved necessary. Since our earlier system had been developed to handle typos only, it had no good provisions for handling multiple non-contiguous errors. This shortcoming towards OCR-errors was remedied by extending the focus word derived anagram values to allow for all character n-grams, where n <= 3, to be derived and used in the search for variants, rather than only the contiguous character n-grams. Further, the algorithm's reach in terms of Levenshtein distance was extended by allowing for every transformation to occur twice. This is motivated by the fact that the OCR-process is likely to misrecognize e.g. the character bigram 'in' and to render it as 'm'. If 'in' occurs twice within a particular word, this may then very well happen twice. It is possible to further extend the algorithm towards exhaustive LD 4 and higher retrieval. This necessarily comes at a cost.

The pseudo-code for our adapted and extended algorithm is presented next:

```
Set LDlimit to 3
  Foreach CounterOne in (1,2)
    Foreach CounterTwo in (1,2)
      For each AlphabetValue in Alphabet
        For each FocuswordAlphabetValue in FocuswordAlphabet
        NewValue = FocuswordAnagramValue − (FocuswordAlphabetValue
        × CounterOne) + (AlphabetValue × CounterTwo)

          If NewValue defined in LexiconHash
          VARIANTS = list of string variants associated with NewValue
          Get WordLength of FocusWord
          If CounterOne = 2 or CounterTwo = 2 and WordLength > 9
          LDlimit = LDlimit × 2
          Endif
          If WordLength < 7
          LDlimit = 2
          Endif
          Else
          LDlimit = 3
          Endelse
            For each variant in VARIANTS
            Calculate Levenshtein Distance between focus and variant
              If LD <= LDlimit
              Return variant
Endif ; Endfor ; Endif ; Endfor ; Endfor ; Endfor ; Endfor
```

4.3 Character Normalization

Preliminary experiments were performed on the SGD. The SGD's higher OCR-accuracy (largely due to better print quality, modern fonts more suited to the OCR-system, clearer page layout) estimated by the National Library at 99% provided a gentler introduction to the peculiarities of post-correction of OCR-errors than the DDD, estimated at 70%, would have. In these initial experiments we worked with the full alphabet encountered within the corpus. In the SGD we observed that 187 characters were actually in use. Given the far larger search space created by the OCR-process in the DDD, we in later stages decided to opt for extensive character normalization. This was performed according to the following scheme: non-visible, non-printing characters were simply discarded. All text is lowercased. All digits and numbers are rewritten as a single '3'. All punctuation marks, except hyphens and apostrophes, are rewritten as a single '2'. Uppercased characters bearing diacritics are rewritten as '4'. Lowercased characters bearing diacritics are rewritten as '5'. This leaves us an alphabet of just 32 characters. Normalization is not a necessary step, but it constricts the search space and promotes the scalability of the system.

Tokenization on low-accuracy OCRed text is not a good idea since all words into which the process inserted punctuation are then split. For performance evaluations for text processing of noisy inputs, please refer to [19]. The frequency lists are in this new scheme obtained by regarding any string delimited by spaces as a word string, instead of after tokenizing.

4.4 Processing Steps

The system first reads in the validated lexicon, if available. Words in the validated lexicon are currently invariably rejected as variants during further processing, i.e. we do not do real-word correction. For the tests reported here, this validated lexicon was primarily based on a new open-source spelling checking dictionary[4] containing not only lemmas, but also expanded word forms. TICCL next reads in the word unigram frequency list derived from the corpus to be cleaned. The validated lexicon has no frequency information, but any word type already logged there inherits the frequency information from the corpus-derived list. Next, TICCL reads the list of focus words to be processed, typically a range of frequency/word length values covering a part or the whole of the corpus to be cleaned. If available, TICCL further reads in a list of pre-processed variants (merged and split words, variants containing superfluous hyphens, etc. obtained by a script that uses word uni- and bigrams to tackle these specific problems) in order to add these to the other variants retrieved for a particular focus word during normal processing. While TICCL reads in the accumulated lexicons to be processed, it applies the character normalization steps, retaining all surface forms with their normalized version. TICCL next gathers all the variants present within the lexicon for the focus words it is set to work on and applies the Levenshtein distance limit filter to discard variants exceeding the LD limit and the

[4] URL: http://www.opentaal.org/

validated lexicon filter and text-induced morphological variants filter to not retain real-word variants. TICCL finally returns the retained variants to an output file in the form of pairs: (focus word, retrieved variant).

5 Evaluation

5.1 Test Sets - How We Evaluate

We evaluate on a subset of the paired lists of variants and focus word for which we presented error distribution statistics in Section 2. The subsets involved all the variants for the 20 SGD focus words in Table 5, 890 in all, and all the variants for the 17 focus words for 'Het Volk', 3,102 in all. Listed next to the focus words are the numbers of variants found. We have had to rely on sampling the typographical variation present within the corpora, exhaustively gathering all the typographical variants for the focus words. As can be seen there is some overlap in the common words between the SGD and 'Het Volk'. Names, especially names of historical figures, being more tied to their era, provide less opportunity for such overlap.

We evaluate in terms of recall and precision, resulting in the combined F-score [20]. These metrics are derived from the numbers of True Positives (TPs), False Positives (FPs) and False Negatives (FNs) returned by the system. **True**

Table 5. Overview of the SGD and DDD focus words and their observed numbers of variants which constitute the evaluation sets. Capitalized words are proper names.

Focus SGD	#	Focus 'Het Volk'	#
Achtienribbe-Buijs	23	Amsterdam	307
Amsterdam	43	Annexionisten	20
Bolkestein	18	België (Belgium)	104
Jorritsma-Lebbink	33	Bismarck	10
Nieuwenhoven	22	Compiègne	3
Rotterdam	47	Hindenburg	32
Wolffensperger	25	Nederlandsche (Dutch)	572
belasting (tax)	36	Posthuma	264
belastingen (taxes)	56	Richthofen	7
belastingplichtige (taxable person)	41	Trotzky	45
belastingplichtigen (taxable persons)	37	Wilhelmina	42
doelstelling (aim)	82	Zeeuwsch-Vlaanderen	19
doelstellingen (aims)	58	belasting (tax)	102
evaluatie (evaluation)	44	belastingen (taxes)	34
faciliteiten (facilities)	27	distribueeren (to distribute)	52
goedkeuring (approval)	36	eenheidsworst (unity sausage)	21
inkomstenbelasting (income tax)	81	regeering (government)	1468
motorrijtuigenbelasting (motor vehicle tax)	70		
studiefinanciering (study financing)	93		
vennootschapsbelasting (corporate tax)	52		

Positives are defined by what constitutes the **target** of our exercise. The target for TICCL are the non-word variants (be they typos or OCR-errors) present in the corpus-derived list to be processed. **False Positives** are non-word variants, or real-words absent from the lexicon, that are erroneously reported to be variants for a particular focus word. **False negatives** are those items in the list of known, annotated variants[5] for the particular focus word that are absent from the list of variants returned, i.e. that the system was not able to retrieve or 'correct'.

The scores we present in Table 6 and Table 7 are the scores obtained per LD. The sum of True Positives and False Negatives gives the total amount of variants on which we evaluated. We list the Recall R, Precision P and F-scores F obtained at each LD and complement these with the Cumulative Recall CR, Cumulative Precision CP and Cumulative F-scores CF to the particular LD. The score at a particular LD is thus measured for each variant retrieved which is 1 or 2 or 3 or more edits removed from the focus word. The cumulative score is based on the accumulated numbers of TPs, FNs and FPs observed at each LD up to the LD under scrutiny. The formulae used are as follows:

$$\text{Recall} = \text{R} = \frac{TP}{TP+FN} \qquad \text{Precision} = \text{P} = \frac{TP}{TP+FP}$$

Since we deem recall and precision to be equally important, the harmonic mean of R and P, the simplified F measure, F, is given by:

$$\text{F-score} = \text{F} = \frac{2\times R\times P}{R+P}$$

Recall expresses to what extent TICCL has been able to identify the non-words (typos and OCR-errors) in the full corpus derived list. **Precision** then expresses to what extent TICCL has been able to assign an identified variant to its proper canonical form. In other words, we desire the list of possible variants returned by TICCL to contain as many as possible of the actual variants present in the corpus-derived word type list for a particular focus word and as few as possible of the variants actually belonging to another focus word. Note that in these evaluations we are very strict: a variant reported for e.g. the singular form 'tax' which in fact in the original text read 'taxes', will be counted as an FP. Note too that precision scores provide information about the coverage of the lexicon used. In the SGD we find the hapax legomenon 'gelasting' which should be understood as 'court order', i.e. a domain specific kind or 'order'. The word is absent from the validated lexicon we used, is returned as a variant for 'belasting ('tax') and consequently counted as an FP. The Zipfian nature of the distribution of errors over the LDs entails that the cumulative scores at higher LDs remain relatively high: there being fewer errors at higher LDs means that the relative proportion of these in the cumulative score is smaller, i.e. they have less impact on the cumulative score. It should be noted that our scores do not take into account the token frequencies of the variants retrieved. The scores reported here are scores on word types only, so each retrieved variant has an equal share in the

[5] This motivates why we only evaluate on subsets of our collections of annotated variants: all variants for a particular focus word need to be annotated, exhaustively.

Table 6. Overview of the SGD performance scores

Measured at	Items retrieved			At LD			Cumul. to LD		
LD	TP	FN	FP	R	P	F	CR	CP	CF
LD 1	466	4	7	0.991	0.985	0.988	0.991	0.985	0.988
LD 2	284		129	1.000	0.688	0.815	0.995	0.847	**0.915**
LD 3	106	1	525	0.991	0.168	0.287	0.994	0.564	0.720
LD 4	11	11	133	0.500	0.076	0.133	0.982	0.522	0.682
LD 5	1	6	22	0.143	0.043	0.067	0.975	0.515	0.674

Table 7. Overview of the DDD performance scores

Measured at	Items retrieved			At LD			Cumul. to LD		
LD	TP	FN	FP	R	P	F	CR	CP	CF
LD 1	380	6	4	0.984	0.990	0.987	0.984	0.990	0.987
LD 2	1112	9	114	0.992	0.907	0.948	0.990	0.927	**0.957**
LD 3	1558	3	613	0.998	0.718	0.835	0.994	0.807	0.891
LD 4	25	9	46	0.735	0.352	0.476	0.991	0.798	0.884

scores obtained. Weighting the scores by token rate would in fact likely make the scores look better, in so far that [21] predicts that small LD accidents happen far more often than larger LD accidents.

5.2 Discussion of Evaluation Results - Future Work

In collecting the evaluation sets we did not strive at obtaining a full balance between the proportion of names included (SGD: 35%, DDD: 70.6%). This discrepancy explains why our system apparently performs better on lesser quality OCRed text than on far better quality OCRed text. In fact, performance on at least some of the names is markedly better than on common words. For some names, there simply are no other words resembling them to the extent that these would fall within the LD of 1, 2 and even more edits. As such, words with a higher neighbourhood density [22], especially short words and words derived from a stem and highly common pre- and/or affixes such as e.g. 'belasting', are far more likely to incur more False Positives.

We have here reported solely on TICCL's capabilities of retrieving a focus word's variants and have reported these in terms of both Recall and Precision. We believe TICCL in this guise constitutes a usable and useful system: if it is set to work within the limits of LD 2, given our statistics on the actual distribution of OCR-errors within these text collections, our scores in effect mean that for the SGD almost 89% and for 'Het Volk' almost 55% of the less desirable typographical variation in terms of non-words present can now fully automatically be removed, as these are the summed percentages of Levenshtein distance 1 and 2 errors observed in our 5,047 SGD and 3,799 'Het Volk' error samples. These results are obtained by using only very simple but efficient and effective retrieval and filtering strategies. In a future paper we hope to report on how by extending our system with substring processing capabilities (Dutch compounds being written as single words) and by taking account of context, we achieve higher precision scores on higher LD errors.

6 Conclusion

We believe to have demonstrated that we have a built a competitive system which can greatly enhance the overall quality of large corpora of OCRed text aimed at text retrieval. The system being inherently simple, sufficiently efficient to scale to very large corpora and yet not language-dependent in se, we hope to see it adopted in large scale digitization programmes around the world and adapted to local needs. To further this, we intend to release TICCL to the Open Source community in due course.

Acknowledgments. This work was funded by the Netherlands Organisation for Scientific Research (NWO).

References

1. Baayen, R.H.: The effects of lexical specialization on the growth curve of the vocabulary. Computational Linguistics 22, 455–480 (1996)
2. Evert, S., Baroni, M.: zipfR: Word frequency distributions in R. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, Posters and Demonstrations Session, Prague, Czech Republic (2007)
3. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
4. Reynaert, M.: Corpus-Induced Corpus Clean-up. In: LREC 2006: Fifth International Conference on Language Resources and Evaluation, Magazzini del Cotone Conference Center – Genova, Italy, Paris, ELRA, European Language Resources Association (2006)
5. Pollock, J., Zamora, A.: Collection and characterization of spelling errors in scientific and scholarly text. Journal of the American Society for Information Science 34, 51–58 (1983)
6. Damerau, F.J.: A technique for computer detection and correction of spelling errors. Communications of the ACM 7, 171–176 (1964)
7. Schneider, P.: Computer assisted spelling normalization of 18th century English. Language and Computers 36, 199–211(13) (2001)
8. Ernst-Gerlach, A., Fuhr, N.: Retrieval in text collections with historic spelling using linguistic and spelling variants. In: JCDL 2007: Proceedings of the 2007 conference on Digital libraries, pp. 333–341. ACM Press, New York (2007)
9. Pilz, T., et al.: Rule-based search in text databases with nonstandard orthography. Literary and Linguistic Computing 21, 179–186 (2006)
10. Hauser, A., et al.: Information access to historical documents from the Early New High German Period. In: Burnard, L., et al. (eds.) Digital Historical Corpora - Architecture, Annotation, and Retrieval. Dagstuhl Seminar Proceedings, Dagstuhl, Germany, IBFI (2007)

11. Adriaans, F., et al.: A cross-language approach to historic document retrieval. In: Lalmas, M., et al. (eds.) ECIR 2006. LNCS, vol. 3936, pp. 407–419. Springer, Heidelberg (2006)
12. Taghva, K., Stofsky, E.: OCRSpell: an interactive spelling correction system for OCR errors in text. International Journal on Document Analysis and Recognition 3, 125–137 (2001)
13. Strohmaier, C.M., et al.: A visual and interactive tool for optimizing lexical post-correction of OCR-results. IEEE Computer Society, Los Alamitos (2003)
14. Mihov, S., Schulz, K.U.: Fast approximate search in large dictionaries. Journal of Computational Linguistics 30, 451–477 (2004)
15. Mihov, S., et al.: Tuning the selection of correction candidates for garbled tokens using error dictionaries. In: Finite State Techniques and Approximate Search, Proceedings of the First Workshop on Finite-State Techniques and Approximate Search, Borovets, Bulgaria, pp. 25–30 (2007)
16. Ringlstetter, C., Schulz, K.U., Mihov, S.: Orthographic errors in web pages: Toward cleaner web corpora. Computational Linguistics 32, 295–340 (2006)
17. Kolak, O., Resnik, P.: OCR post-processing for low density languages. In: HLT 2005: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp. 867–874. Association for Computational Linguistics, Morristown, NJ, USA (2005)
18. Reynaert, M.: Text-Induced Spelling Correction. PhD thesis, Tilburg University (2005)
19. Lopresti, D.: Performance evaluation for text processing of noisy inputs. In: SAC 2005: Proceedings of the 2005 ACM symposium on Applied Computing, pp. 759–763. ACM Press, New York (2005)
20. van Rijsbergen, C.J.: Information Retrieval. Butterworths, London (1975)
21. Zipf, G.K.: The psycho-biology of language: an introduction to dynamic philology, 2nd edn. The MIT. Press, Cambridge (1935)
22. Frauenfelder, U., et al.: Neighbourhood density and frequency across languages and modalities. Journal of Memory and Language 32, 781–804 (1993)

Linguistic Support for Revising and Editing

Cerstin Mahlow and Michael Piotrowski

University of Zurich
Institute of Computational Linguistics
Binzmühlestrasse 14
8050 Zürich, Switzerland
{mahlow,mxp}@cl.uzh.ch

Abstract. Revising and editing are important parts of the writing process. In fact, multiple revision and editing cycles are crucial for the production of high-quality texts. However, revising and editing are also tedious and error-prone, since changes may introduce new errors.

Grammar checkers, as offered by some word processors, are not a solution. Besides the fact that they are only available for few languages, and regardless of the questionable quality, their conceptual approach is not suitable for experienced writers, who actively create their texts. Word processors offer few, if any, functions for handling text on the same cognitive level as the author: While the author is thinking in high-level linguistic terms, editors and word processors mostly provide low-level character oriented functions. Mapping the intended outcome to these low-level operations is distracting for the author, who now has to focus for a long time on small parts of the text. This results in a loss of global overview of the text and in typical revision errors (duplicate verbs, extraneous conjunctions, etc.).

We therefore propose functions for text processors that work on the conceptual level of writers. These functions operate on linguistic elements, not on lines and characters. We describe how these functions can be implemented by making use of NLP methods and linguistic resources.

1 Introduction

Writing a text involves several steps and various tasks, starting from planning activities to writing a first draft and then revising and editing[1] to get to the final version. Revising and editing are typically recursive processes, continuing until an acceptable state is achieved.

Writing means creating a coherent text from linguistic elements, such as words, phrases, clauses and sentences. When revising and editing texts, authors are working with these elements, arranging and rearranging them, exchanging them for others, maybe even "playing" with them.

In this paper we will try to develop the idea of tools based on linguistics to support writers in the writing process, especially during revising and editing.

[1] In composition research, a distinction is typically made between *revising*, which takes place on the discourse level, and *editing*, which takes place at the sentence and word level (see [1] for a discussion).

First, to get an idea of the abstraction level on which writers are thinking about their texts, we will have a look at recommendations for writers and editors: What are the concepts and the metalanguage used to talk about textual elements as well as revision and editing tasks?

Next, we will analyze functions in state-of-the-art word processors to find out on which conceptual level they operate and which support they offer for revising and editing.

Finally, we will develop ideas for software functions for revising and editing, which use linguistic knowledge to provide writers with functions operating on a conceptual level closer to their own. As examples we will describe the possible mode of operation for some functions based on linguistic concepts.

2 Writing: Composing, Editing, Revising

In this section we will explore two aspects: The language used by researchers when they talk about what people are doing when writing and the abstraction level or metalanguage used by composition teachers for recommendations.

2.1 Writing as a Process – What Do Writers Do When Revising?

Research over the last 30 years has shown that writing should be regarded as a process leading to a text, i.e., the focus in research has moved from the resulting product to the process. In the U.S. and Canada this shift in view started in the 1970s; in the German-speaking part of Europe it was around 10 years later (cf. [2]) that researchers started to focus on what people actually do when they write, revise, and edit and no longer on what people *should* do.

Experiments (e.g., [3,4,5,2]) have shown, that writers of all ages work in loops and cycles, acting as composer, revisor and editor of their own text. Most of the revising and editing takes place in a conscious phase *after* composing a text or a text fragment.

Writers perform different actions during revising; according to [4] they correct mistakes, amend elements of style, and restructure or rewrite portions of the text. Revising and editing takes place at different levels of the text (cf. [2]):

- Text (structure, logic, comprehensibility)
- Paragraph and sentence (grammar, including conjunctions, tense, syntax, etc.)
- Word (conciseness, diction, expression)
- Layout and spelling

Thus, the language used by researchers to describe the writing process, is clearly influenced by linguistic terms – they do not talk about, say, characters or lines.

2.2 Recommendations for Revising and Editing

Recommendations for writers are made both on the basis of experiments monitoring writers doing a certain task, as well as based on the daily experience of expert writers.

For example, based on research on comprehensibility, Langer, Schulz von Thun, and Tausch [6] postulate four requirements for texts: They should be reader-friendly, logical, concise, and stimulating. These are high-level goals to be strived for when revising a text. Each of them are to be achieved on the levels of the word, the sentence, and the text as a whole. Most of the practical recommendations eventually refer to linguistic elements, such as noun phrases, compounds, mode and tense of verbs, modal verbs, word order, construction of phrases and clauses, etc. (see also [7]).

Similarly, experienced writers also refer to linguistic elements when describing what to do with a text to revise it to achieve specific goals like targeting a specific audience or communicating a certain message.

3 Support for Writers in Word Processors

The considerations outlined above lead us to inquire whether word processors offer any functions *on the same level of abstraction* to support writers in these revising and editing tasks. To answer this question, we will look at two aspects of word processors: First, the automatic checkers for spelling and grammar and style, and second, the general editing functions offered to writers by word processors. We are not concerned with text properties such as organization or discourse-level structures.

As a representative we will consider Microsoft Word because of its ubiquity and richness of functions and add-ons.

3.1 Checkers

The automatic checkers in Word can be used in two modes: During writing ("as you type") or upon manual invocation (e.g., to run it on the current state of a text). The checkers flag textual elements which they consider problematic and give the user information on the nature of the problem and a suggestion for remedial. The spelling checker flags individual misspelled words, while the grammar and style checker marks words, phrases or whole sentences. Both the spelling checker and the grammar checker can be set to the language of the document. The grammar and style checker can also be configured with regard to different styles or genres.

Grammar and style checkers have been developed since the 1980s, evolving from research systems to inexpensive commercial packages or add-ons for word processors today (cf. [3, p. 332]). Today, Microsoft's Grammar and Style Checker is the most widely used checker. Studies from composition teachers (e.g., [3,8]) have shown that grammar checkers don't work accurately and reliably, but that there are educational scenarios where you can make good use of them; for example, one can use a flagged element and the explanation and proposed revision offered by the grammar checker as a starting point for classroom discussions about grammar, thereby gaining insights about grammar and style.

However, this type of scenarios was obviously not one intendend by the developers. With respect to their original purpose of helping writers with producing texts conforming to grammar and style or genre rules, authors agree in discouraging their use (e.g., [3,8]). Studies have shown that grammar checkers detect only a small portion of problems in a text, that they flag false positives, that they frequently detect problematic sentences but incorrectly identify the issues, that they give misleading or misunderstandable hints for revising, etc.

Being aware of these problems, experienced writers therefore tend to ignore the flags or try to ignore some flags, or, on the other hand, give up and adapt their writing to avoid getting criticized by the grammar checker [8, p. 462]. Particularly the latter behavior is obviously problematic. It is probably caused by the fact that most users of word processors do not know that they can, in fact, turn off or configure the grammar checker. Restricting themselves to certain sentence models or phrases takes away the aspect of "playing" with language and severely restricts their range of expression (cf. [8, p. 463f]). This is an even greater problem with basic writers; Heilker [9, p. 65] reports that basic writers tend to consider the checkers' suggestions as authoritative – maybe even more authoritative than those of a human teacher.

The checkers tell writers about isolated problems found in the text, in a linear fashion from the beginning to the end, if the checking is started manually. There is no way to get an overview over all detected problems, concerning, for instance, a high number of consecutive nouns.

To summarize: Automatic checkers should be used with care. It is thus questionable whether they are useful for revising and editing, especially for experienced writers.

3.2 Processing Functions

Word processors offer writers a number of functions for editing: Select, cut, copy, and paste, insert and delete, search and replace, etc. However, all of these functions operate on characters or lines, not on lingustic units, i.e., there are no functions for performing actions like *mark the sentence in which the cursor is placed* or *insert the content of the clipboard after the word on which the cursor is placed*.

There are some functions in Microsoft Word, where it seems that there is a concept like *word*. For example, a "word" can be selected by double-clicking on it. In fact, however, this includes all characters between two delimiting spaces, so this function does not select a word in the linguistic sense. Another example is the "Smart Cut and Paste" option, which adds or removes spaces when cutting or pasting text. The idea behind "Smart Cut and Paste" is to make sure that spaces are added around the word so that it does not run into a neighboring word. Likewise, when cutting a word from a sentence, the surrounding spaces are removed, so that there is only a single space left. However, the insertion – including spaces – will be made exactly at the position of the cursor, regardless of whether it is in a word or between two words; see fig. 1 for an illustration.

1. Cursor positioned inside a word	2. Result after pasting "the"
`said wo|rd`	`said wo the |rd`

Fig. 1. Behavior of "Smart Cut and Paste" in Microsoft Word; "|" indicates the cursor position

So, even though some functions may seem to operate on linguistic units, in fact they are still based on characters: A "word" is an alphanumeric string delimited by spaces, and a "sentence" is a sequence of alphanumeric strings, ending with a punctuation mark. These definitions are not only used in word processors, but also in programmer-oriented text editors like XEmacs or **vi**. Both offer more functions operating on "words" and "sentences" than Microsoft Word, but these are not based on linguistic knowledge either.

3.3 Problems

Spelling checkers and grammar and style checkers put the writer into what is essentially a passive position. The decision about whether something is grammatical or not is made by the system, and what is more, it offers only negative feedback.

Inside the grammar checker, some knowledge of linguistic concepts like words and sentences, and even nouns and verbs, is certainly available. But these concepts are not available to the editing functions, and the output of the grammar checker is not linked to the editing functions. For example, an element flagged as problematic can only be replaced as a whole (either with the system's suggestion or with a replacement entered by the user), but it is not possible to make additional (or alternative) amendments *outside* the flagged region.

Using the suggestion box of the grammar checker also forces the writer to edit similiar problems at different points in the text point by point. There is no way to link from the checker to the "search and replace" function of the word processor to edit all occurences at once. Or, if the writer is not satisfied with a suggestion for a flagged element, but changing the word order would solve the problem, it is not possible to select additional words and/or reorder them by mouse click or keystroke.

Experienced writers know the mistakes they frequently make, but they cannot search for specific errors. Using a checker means relying on a system where a writer cannot be sure whether it will actually find all problematic elements with respect to specific rules. It is therefore not surprising that McGee and Ericsson [8, p. 462] observed that experienced writers tend to ignore all or some of the flags.

On the other hand, word processors have no functions which implement operations on a level comparable to the suggestions made by the grammar checkers. For example, there is no function for changing all passive sentences into active ones. Writers would often like to change the word order, whether for grammatical, semantic, or stylistic reasons. This operation is very error-prone, since the writer has to mark the exact number of characters to select a word, cut it out, move the cursor to the right position, and then insert it from the clipboard. Even worse, when reordering words from or to the beginning of the sentence, capitalization must be corrected by changing the first letters of each word after reordering. There are no functions for all of these common revising and editing tasks.

The lack of functions operating on linguistic elements is disappointing for writers writing in any language. However, while functions like *search and replace* can be used to a certain degree to make global edits in analytic languages like English, writers have to be very careful and reedit the result of such operations in inflectional languages like German. Even worse is the situation with grammar and spell checkers: The results for English are already unsatisfying; according to Vernon [3, p. 340 and following], checkers only detect one third of typical mistakes. For highly inflectional languages like German, the quality is probably much worse, primarily due to the more complex morphology and syntax.

4 Concepts for Linguistically Supported Revising and Editing

We have seen that current word processors offer writers only very restricted functions for editing and revising. While grammar checkers are based on linguistic knowledge, they are cumbersome to use and only of limited use to experienced authors. Thus, if writers want to achieve certain goals while revising or editing, they have to break down their high-level linguistic ideas into a large number of character-level operations. We think that this may cause attentional disruption, which could be another reason for the tendency to primarily make relatively localized surface corrections when editing on screen, as noted by various studies (cf. [10, p. 259], [11, p. 567], [12, p. 102 and following][2]).

As Piolat points out, "to improve a text, writers must successively make a series of corrections, while checking to see that each one is compatible with others, often located at different linguistic levels" [10, p. 266]. The tool used for composing, revising, and editing should offer support appropriate for these tasks, i.e., it should be aware of linguistic phenomena occuring in the text, it should help the writer control the text, and it should be aware of consequences that changes may have. Thus we think that tools for writers should have functions based on linguistic elements, supporting writers on different levels. Functions on each level should enable the writer to work on linguistic elements – words, phrases, clauses and sentences. However, while word processors should support authors when writing, revising, and editing their texts, the word processor should not control the writing.

We envision two basic types of desirable functions, which we will describe in the rest of this section.

4.1 Highlighting

In the same way that spelling checkers or grammar checkers highlight "incorrect" words or phrases, writers should be able to ask for specific linguistic elements to be highlighted, e.g., conjunctions, verbs in a specific mode, all verbs, sentences without verb, sentences with more than one finite verb, etc. The goal of highlighting is to give the writer a quick overview of these constructions. The interpretation, however, should be left to the author and must not be made by the system (in contrast to the checkers currently available).

When teaching composition, there are pedagogical scenarios where students are asked to mark specific linguistic constructions in their own text by hand to get an overview about the constructions they have used, as described by Eyman and Reilly [12, p. 106]:

> An instructor can ask students to change active verbs to boldface, highlight passive constructions in italics, use larger fonts for descriptive words, underline the thesis statement, or select particular font colors for topic sentences in each paragraph. This kind of visual marking presents a striking image of the text and can show the writer elements that may be overused or missing. Obviously, this sort of exercise requires instruction in identifying these constructions within a text, which may also help students gain control of their prose by providing them with the tools needed to analyze and discuss it.

We think that this kind of highlighting – provided automatically – is not only useful for basic writers: Even experienced writers do not always have an overview of the

[2] Most of the available studies rely on factors like screen size and legibility to explain this phenomenon; we do not know of any studies analyzing editing functions *per se*.

constructions they have used; having this information available during the process of revising and editing would be useful.

4.2 Support for Editing Actions

Tools should also provide functions that support writers in performing certain actions while revising and editing, e.g., changing the word order, replacing words, changing the mode or tense of verbs, replacing pronouns with noun phrases, etc. Such actions can (often) be performed without affecting elements other than the focused element. This type of action will therefore be called a *restricted action*.

In a more advanced scenario, there could be support for more complex actions, which affect words or phrases not directly involved: Changing the number of the subject requires changing the number of the finite verb and vice versa, replacing the noun in a complex noun phrase may require other changes to ensure congruency, etc. We will call this type of actions *actions with side-effects*.

Depending on the language, replacing a word by another can be a restricted action or one with side-effects. Ideally, the writer should not be forced to distinguish between both variants; the word processor should deal with this problem and offer additional options and highlighting for actions with side-effects.

5 An Outline of Editor Functionality Based on Linguistics

To prove our concept of word processor functions based on linguistic elements we are currently working to implement various functions for *highlighting* and a number of *actions*, as described in section 4, and evaluate them with experienced writers. These functions will be selected on the basis of research in composition, see, e.g., [13], [2].

We will implement these functions in the XEmacs[3] text editor. We have chosen XEmacs for the following reasons: It is open-source, new functions can easily be added using Emacs Lisp, either as additional functions or replacing existing functions, all predefined functions are available in source form and can therefore be analyzed and adjusted, and XEmacs and the similar GNU Emacs are text editors preferred by many "power users" and experienced writers (who are using markup languages like LaTeX, **troff**, or XML). For some functions we can adapt existing XEmacs functions, e.g., for highlighting or changing elements. By implementing the new functions in an existing text editor, we also hope to be able to show that word processors in general can be adapted to use concepts beyond characters and lines without the need to rewrite them from scratch.

In this section, we will describe the functions `show-conjunctions`, `transpose-words-consider-case`, `transpose-conjuncts`, and `query-replace-word`. The basic approach of these functions should at least be applicable to most European languages, as the underlying linguistic concepts are similar; of course, appropriate linguistic resources are required for each language.

Using `query-replace-word` as an example, we will show that for highly inflectional languages, like German, the complexity of some revising and editing actions is considerable and requires linguistic support.

[3] `http://xemacs.org/`

For each function we will describe the required linguistic resources. In general, the processes enabling the linguistic support should be as shallow as possible for them to be usable in interactive mode. For example, there is generally no need to parse the whole text syntactically or morphologically; restricted concepts of "word" and "sentence", for tokenizing around the current position of the cursor, morphological analyses of a few words at a time, and generation of a few words at a time will be sufficient.

5.1 Show Used Conjunctions

Highlighting of key elements is known from editors for programming languages. For XEmacs, there are specialized editing modes for numerous programming languages, which provide the highlighting of key elements (so-called *syntax highlighting*). We will use the highlighting functionality of XEmacs to show the conjunctions used in a text.

For most languages, conjunctions are a closed word class consisting of invariable words. Executing the command `show-conjunctions` will give a quick overview of the use of conjunctions. In addition, the command `show-conjunctions-frequency` will display a frequency list of conjunctions, which allows writers to see whether they have a preference for certain conjunctions; the command `show-conjunctions-sequence` lists the conjunctions in the order in which they appear in the text.

For these functions, only minimal linguistic resources are required. As conjunctions are typically invariable, there is no need to look for different wordforms, and since conjunctions are not linguistically productive (i.e., no new conjunctions are produced using derivation or composition), it is not necessary to consider morphological processes. Thus, only a list of conjunctions is needed.

5.2 Reorder Words

XEmacs has the built-in function `transpose-words` (or keystroke `M-t`) to reorder words; in most cases, this function is used to interchange two words. Each word will keep its case. However, when a word is moved to or from the beginning of a sentence, it may be necessary to change the case of the involved words.

The function `transpose-words-consider-case` will have the same effect as `M-t`, but it will consider the case of the words and adjust it if necessary. For German, there is an additional challenge: If a word is a noun it is always capitalized, regardless of its position. Thus, for German, transposing two words with respect to case includes several aspects:

- If a word is at the beginning of a sentence, and it is moved to a non-sentence-initial position, it is lowercased, unless it is a proper name or a noun. To make this decision, morphologic analysis is required. The word that is now in sentence-initial position must be capitalized.
- If a word is moved to the beginning of a sentence, it is capitalized. The word that used to be at the beginning of this sentence must now be lowercased, unless it is a proper name or a noun (see above).
- If a word is moved between non-sentence-initial positions, the function will behave like `transpose-words`.

For the purpose of this function the standard XEmacs notions of "word" and "sentence" are sufficient. As a linguistic resource morphologic analysis is required.

5.3 Reorder Conjuncts

In this article, we are using the coordination "revising and editing" several times. If you look carefully, you will find no occurrence of "editing and revising". However, our first draft of this paper contained both versions. Often one version of such a coordination is preferred and should be used consistently. If the `transpose-words` function (described above) is used, at least three operations have to be performed to change "editing and revising" to "revising and editing", as shown in fig. 2.

Step	Command	State
		editing and revising|
1.	C-u -2 M-t	revising| editing and
2.	M-f	revising editing| and
3.	M-t	revising and editing|

Fig. 2. Editing operations necessary to change "editing and revising" to "revising and editing" with XEmacs

When using a word processor that does not offer a function like `transpose-words` (e.g., Microsoft Word), at least 8 steps are necessary (cf. fig. 3).

Step	Command	State
		editing and revising|
1.	Alt Shift ←	editing and revising
2.	Cmd X	editing and |
3.	Alt ←	|editing and
4.	Cmd V	revising |editing and
5.	Alt Shift →	revising editing and
6.	Cmd X	revising |and
7.	Alt →	revising and |
8.	Cmd V	revising and editing|

Fig. 3. Editing operations necessary to change "editing and revising" to "revising and editing" in Microsoft Word with "Smart Cut and Paste" (Mac key combinations; the shaded background indicates the current selection)

The function `transpose-conjuncts` will allow writers to interchange the conjuncts of a coordination with a single command. The cursor has to be placed on the conjunction. This function requires the same linguistic resources as `transpose-words-consider-case`. Extending this function to cover cases with compounds written as separate words (as in English), e.g., "word processors and editors" will require additional linguistic knowledge.

5.4 Replace Words

XEmacs has the built-in function `query-replace` (or keystroke `M-%`). Replacing one string with another will (by default) keep capitalization – essential for proper names and nouns in German, or words at the beginning of a sentence. However, in inflectional languages, like German, words can have many *word forms*, i.e., inflected forms, and each word form can typically express more than one category, see table 1.

Table 1. Word forms of *Haus*, *Zelt*, and *Hütte*

Word	Word forms	Categories
Haus (*n*, (e)s/er decl.)	Haus	Nom Sg, Dat Sg, Acc Sg
	Hauses	Gen Sg
	Hause	Dat Sg
	Häuser	Nom Pl, Gen Pl, Acc Pl
	Häusern	Dat Pl
Zelt (*n*, (e)s/e decl.)	Zelt	Nom Sg, Dat Sg, Acc Sg
	Zeltes	Gen Sg
	Zelts	Gen Sg
	Zelte	Dat Sg, Nom Pl, Gen Pl, Acc Pl
	Zelten	Dat Pl
Hütte (*f*, –/en decl.)	Hütte	Nom Sg, Gen Sg, Dat Sg, Acc Sg
	Hütten	Nom Pl, Gen Pl, Dat Pl, Acc Pl

Manually replacing all occurrences of *Haus* ‘house’ with the corresponding word form of *Zelt* ‘tent’ is therefore a complex task: First, one has to find all word forms of *Haus* – with the usual search functions this will require to search for each word form individually. Then, one has to determine the category of a specific occurrence; note that the word form may be ambiguous, and the exact category can only be found by looking at the syntactic context. Finally, one must manually replace the word form of *Haus* with the corresponding word form of *Zelt*.

As *Haus* and *Zelt* are of the same gender, congruency with respect to determiners, adjectives, and pronouns is not affected. However, replacing all word forms of *Haus* with the corresponding word forms of *Hütte* ‘hut’ will compound problems with congruency since *Haus* is neuter while *Hütte* is feminine.

It is clear that having to make these changes while revising a text is very distracting, and support by a writing tool would therefore be desirable. We are thus proposing a function `query-replace-word`, which would operate as follows: After calling the function, the writer is prompted to enter the word to replace (*from-word*) and its replacement (*to-word*). The function then checks the word classes to ensure that they are identical, otherwise it falls back to the standard `query-replace` function. The function then searches for all forms of the paradigm of *from-word*; when a form of *from-word* is found, the user can choose a word form from the paradigm of *to-word* from a list, or the replacement can be skipped. To ease the selection, the replacement forms could be ordered according to their likelihood by comparing categorial features; for example, when the word form *Haus* is found, it is known that this form is singular and nominative, dative, or accusative. Thus, only the replacement forms *Zelt* or *Zelte* are possible (cf. 1) and can be presented as first choices.

As a further enhancement, the function could try to determine the exact category by partially parsing the immediate context. This would make it possible to present the exact replacement form as default choice.

The best handling of side-effects, as caused by a change of gender, will have to be determined experimentally. We are currently considering the following possibilities:

– The editor could correct the side-effects automatically. This would be desirable, but, unfortunately, quite difficult.
– Potential trouble spots are highlighted, and the writer gets the chance to immediately correct the problems in the context of the current replacement manually, possibly with the aid of a grammar checker.
– Potential trouble spots are highlighted, and the writer can correct them after the end of the replacement process, possibly with the aid of a grammar checker.

For a basic implementation, morphologic analysis and generation are required. To find the exact replacement and to detect side-effects, syntactic information is required, which could be provided by parsing of the context.

In the description above, we have been concerned with nouns. For other word classes, other considerations may be neccessary. One example is the replacement of a simple verb with a separable-prefix verb in German, as in: “Er *notierte* sich die Nummer” vs. “Er *schrieb* sich die Nummer *auf*”.[4]

6 Conclusion

We have shown that today’s word processors give writers only little support when revising and editing text. By looking at research on composition, revising and editing, we have seen that the concepts writers use to reason about their texts, and the operations they perform are predominantly on a linguistic level. Based on that insight, we have determined two groups of functions that word processors should have to better support the writing process: (1) Specific views for highlighting linguistic phenomena, and (2) functions to perform operations on linguistic units. As a proof of concept, we have specified four functions for the editor XEmacs and have outlined the required linguistic resources.

The functions described in section 5 (`show-conjunctions`, `transpose-words-consider-case`, `transpose-conjuncts`, `query-replace-word`) may seem trivial and require only relatively few linguistic resources. However, we think that they can relieve writers from many low-level operations, which distract writers from the actual revising and editing. Despite its relative simplicity, we are not aware of any word processor or editor implementing this type of functionality. Furthermore, we describe possible extension to the functions, and further functions can be derived from these functions.

[4] Both sentences could be translated as ‘He wrote down the number’.

We are currently working on the actual implementation of the functions described in this paper. We will then evaluate their usefulness in experimental settings with experienced writers.

References

1. Haar, C.: Definitions and distinctions. [14], ch. 2
2. Kruse, O., et al. (eds.): Prozessorientierte Schreibdidaktik. Schreibtraining für Schule, Studium und Beruf. Haupt, Bern, Stuttgart, Wien (2006)
3. Vernon, A.: Computerized grammar checkers 2000: capabilities, limitations, and pedagogical possibilities. Computers and Composition 17, 329–349 (2000)
4. Blatt, I., Hartmann, W. (eds.): Schreibprozesse im medialen Wandel. Ein Studienbuch. Diskussionsforum Deutsch. Schneider, Hohengehren (2004)
5. Merz-Grötsch, J.: Schreiben als System. Band 1: Schreibforschung und Schreibdidaktik. Ein Überblick. 2 edn. Filibach, Freiburg i. Breisgau (2005)
6. Langer, I., Schulz von Thun, F., Tausch, R.: Sich verständlich ausdrücken. Reinhardt, München (2002)
7. Hajnal, I., Item, F.: Schreiben und Redigieren, auf den Punkt gebracht! Huber, Frauenfeld (2003)
8. McGee, T., Ericsson, P.: The politics of the program: MS Word as the invisible grammarian. Computers and Composition 19, 453–470 (2002)
9. Heilker, P.: Revision worship and the computer as audience. Computers and Composition 9, 59–69 (1992)
10. Piolat, A.: Effects of word processing on text revision. Language and Education 5, 255–272 (1991)
11. Piolat, A., Roussey, J., Thunin, O.: Effect of screen presentation on text reading and revising. International Journal of Human-Computer Studies 47, 565–589 (1997)
12. Eyman, D., Reilly, C.: Revising with word processing/technology/document design. In: [14], ch. 7
13. Björk, L., Bräuer, G., Rienecker, L., Jörgensen, P.S. (eds.): Teaching Academic Writing in European Higher Education (Studies in Writing). Springer, Heidelberg (2003)
14. Horning, A., Becker, A. (eds.): Revision: History, Theory, and Practice (Reference Guides to Rhetoric and Composition). Parlor Press (2006)

The Role of PP Attachment in Preposition Generation

John Lee[1] and Ola Knutsson[2]

[1] Spoken Language Systems
MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, USA
jsylee@csail.mit.edu
[2] School of Computer Science and Communication
Royal Institute of Technology, Stockholm, Sweden
knutsson@csc.kth.se

Abstract. This paper is concerned with the task of preposition generation in the context of a grammar checker. Relevant features for this task can range from lexical features, such as words and their part-of-speech tags in the vicinity of the preposition, to syntactic features that take into account the attachment site of the prepositional phrase (PP), as well as its argument/adjunct distinction. We compare the performance of these different kinds of features in a memory-based learning framework. Experiments show that using PP attachment information can improve preposition generation accuracy on Wall Street Journal texts.

1 Introduction

Preposition usage is among the more frequent types of errors made by non-native speakers of English. In an analysis of texts [1], written by students in English-as-a-Second-Language classes, errors involving prepositions form the largest category, at about 29%[1]. A system that can automatically detect and correct preposition usage would be of much practical and educational value. Research efforts towards building such a grammar checking system have been described in [2], [3], and [4].

When dealing with preposition errors, the system typically makes two decisions. First, a *preposition generation* model needs to determine the best preposition to use, given its context in the input sentence. It should, for example, predict the preposition "*in*" to be the most likely choice for the input sentence:

Input: *He participated* **at?** *the competition.*
Corrected: *He participated* **in** *the competition.*

If the predicted preposition differs from the original one, a *confidence* model would then need to decide whether to suggest the correction to the user. In this case, confidence in the predicted preposition "*in*" should be much higher than the original "*at*", and correction would be warranted.

The focus of this paper is on the preposition generation task. Table 1 provides some examples. In particular, we are interested in comparing the effectiveness of different kinds of features for this task.

[1] As cited in [2].

Table 1. Example sentences for preposition generation. The lexical head of the PP is in *italics* and the prepositional complement is **bolded**.

Sent #	Input Text	Output
1	Pierre Vinken *joined* the board ___ a nonexecutive **director**.	as
2	The $2.5 million Byron plant was *completed* ___ **1985**.	in
3	The average maturity for funds *open* only ___ **institutions**, ...	to
4	Newsweek announced new advertising *rates* ___ **1990**.	for

Our research question is made precise and motivated in the rest of this section. Previous work is summarized (§2) and contrasted with our proposed features (§3). After a brief description of the learning algorithm (§4), experimental results are presented (§5 and §6).

1.1 Research Question

The features considered in previous research on preposition generation may be divided into three main types. Lexical features, such as word *n*-grams within a window around the preposition; the parts-of-speech (POS) tags of these words; and syntactic features, such as the word modified by the prepositional phrase (PP), or grammatical relations between pairs of words.

Unfortunately, no direct comparison has been made between these different kinds of features. Intuitively, syntactic features should be helpful in choosing the preposition. How much gain do they offer? Does their utility vary for different kinds of PP, or depend on the size of the training set? This paper seeks to fill this gap in the literature by comparing a lexical baseline feature set with a syntactic feature set that incorporates PP attachment information.

Our key finding is that PP attachment information can improve generation performance. In a memory-based learning approach, this improvement is especially notable when the training data is sparse.

1.2 Theoretical Motivations

Linguistic analyses suggest that the attachment site of the PP, as well as the argument/adjunct distinction, play significant roles in the choice of preposition. This section provides some linguistic background to motivate our research question, and also defines some terminology to be used in the rest of the paper. The material in this section is based on [5], unless otherwise stated.

Attachment. A preposition "expresses a relation between two entities, one being that represented by the prepositional complement, the other by another part of the sentence." The *prepositional complement* is, in most cases, a noun phrase[2]. That "another part of the sentence" can be a verb-, noun- or adjectival

[2] Some prepositions function as particles in phrasal verbs, e.g., "give *up*" or "give *in*". We view these particles as part of the verb and do not attempt generation.

phrase. The PP is said to be *attached* to this phrase, and the head word of this phrase is called the *lexical head* of the PP.

For example, in sentence #1 in Table 1, the preposition "*as*" expresses the relation between the prepositional complement "*director*" and its lexical head, the verb "*joined*". Knowing that the PP is attached to "*joined*", rather than to "*board*", would clearly help predict the preposition "*as*".

Argument/Adjunct Distinction. The relevance of the lexical head for the choice of preposition may depend on its relation with the prepositional complement. One aspect of this relation is the argument/adjunct distinction. In principle, "arguments depend on their lexical heads because they form an integral part of the phrase. Adjuncts do not."[6]. The preposition in an argument PP is thus more closely related to the lexical head than one in an adjunct PP. The distinction can be illustrated in two of the syntactic functions of PPs:

- **Complementation:** The preposition marks an argument of the lexical head. The prepositions "*as*" in sentence #1 in Table 1 is such an example. In this usage, the PP is said to be an *argument*.
- **Adverbial:** The PP serves as a modifier to its lexical head. The phrase "*in 1985*" in sentence #2 is one example. In this usage, the PP is an *adjunct*.

The argument/adjunct distinction has been shown to be helpful in PP attachment [6]; it may also be relevant in preposition generation.

1.3 Practical Motivations

In addition to the linguistic motivations discussed above, the use of PP attachment and the argument/adjunct distinction can also improve the user experience of a grammar checking system.

For a language learner, the system should serve not merely a practical, but also an educational, purpose. Besides having a wrong preposition detected and corrected, the user would also like to learn the reason for the correction, such as, "the verb *X* requires the preposition *Y*". Without considering PP attachment, this kind of feedback is difficult.

By making known its assumptions on the attachment site, the grammar checker also enhances its transparency. If the user spots an attachment error, for example, s/he may choose to inform the system and can then expect a better prediction of the preposition.

2 Previous Work

Previous research on preposition generation and error detection has considered lexical, part-of-speech (POS) and syntactic features.

2.1 Lexical and POS Features

A rule-based approach using lexical features is employed in [3] for Swedish prepositions. The system can identify insertion, deletion and substitution errors, but does not offer corrections.

A variety of lexical and POS features, including noun and verb phrases in the vicinity of the preposition, as well as their word lemmas and POS tags, are utilized in [2]. The evaluation data consist of newspaper text and a corpus of essays written by 11th and 12th grade students, covering 34 prepositions. A maximum entropy model achieved 69% generation accuracy. Differences in the data set genre, however, prevent a direct comparison with our results.

2.2 Syntactic Features

To our best knowledge, the only work on preposition generation that utilizes syntactic features is [4]. In addition to a variety of POS features and some WordNet categories, it also considers grammatical relations (e.g., direct or indirect object) extracted from a parser. The grammatical relation feature is identified as a strong feature. A voted perceptron algorithm, trained on five prepositions, yielded 75.6% accuracy on a subset of the British National Corpus.

3 Features

Despite the variety of features explored in previous work, no analysis on their relative effectiveness has been performed. The main goal of this paper is to make a direct comparison between lexical and syntactic features. We thus propose two feature sets, Lexical and Attach. They are restricted to the same types of features except one difference: the former contains no information on the PP attachment site; the latter does. Some examples of these features are given in Table 2.

Table 2. Two sets of features are to be contrasted. The Lexical feature set does not specify the PP attachment site; the Attach set does so via the Lexical Head feature. Features extracted from the sentences in Table 1 are shown below.

Sent #	Lexical			Attach		
	VP Head (`V`)	NP/ADJP Head (`N1`)	Complement (`N2`)	Lexical Head (`H`)	NP/ADJP Head (`N1`)	Complement (`N2`)
1	joined	board	director	joined	board	director
2	completed	*null*	1985	completed	*null*	1985
3	*null*	open	institutions	open	*null*	institutions
4	announced	rates	1990	rates	*null*	1990

3.1 Lexical Feature Set

Three words in the vicinity of the preposition are extracted[3]:

– **Verb Phrase Head** (`V`) Head of the verb phrase preceding the preposition.

[3] We follow the naming convention in the literature on PP attachment disambiguation (e.g., [7]). Our Lexical feature set is similar to theirs, with one crucial difference: the preposition *itself* is not included as a feature here, for obvious reasons.

– **Noun or Adjectival Phrase Head** (`N1`) Head of the noun phrase or adjectival phrase occurring between `V` and the preposition.
– **Prepositional Complement** (`N2`) Head of the noun phrase or nominal *-ing* following the preposition.

For example, for sentence #1 in Table 2, `V` is "*joined*", `N1` is "*board*", and `N2` is "*director*".

Since the PP may be attached to `V` or `N1`, its attachment site cannot be inferred from this feature set. However, either `V` or `N1` can be missing; for example, in sentence #2, `N1` is *null* because the verb "*completed*" is immediately followed by the PP "*in 1985*". In such a case, then, there is no PP attachment ambiguity.

3.2 Attachment Feature Set

In the Lexical feature set, the PP attachment site is left ambiguous. We hypothesize, on linguistic grounds presented in §1.2, that it can serve as an informative feature. To test this hypothesis, the Attach feature set re-labels the features in Lexical based on the PP attachment site given by the parse tree:

– **Lexical Head** (`H`) If the PP is attached to a verb phrase, the lexical head is `V`; if the PP is attached to a noun- or adjectival phrase, it is `N1`.
– **Noun or Adjectival Phrase Head** (`N1`) Similarly, this could be one of two values. If the PP is attached to a noun- or adjectival phrase, this is *null*; if it is attached to a verb phrase, this is the same as the `N1` in Lexical. In the latter case, the noun may still play an important role in the choice of preposition. Consider the expressions "*keep the pressure* **on** *someone*" and "*keep pace* **with** *someone*". Under the same lexical head "*keep*", the `N1` nouns "*pressure*" and "*pace*" provide strong clues about the different prepositions.
– **Prepositional Complement** (`N2`) Same as in the Lexical feature set.

4 Memory-Based Learning

The memory-based learning framework has been shown to perform well on a benchmark of language learning tasks [8]. In this framework, feature vectors are extracted from the training set and stored as a database of instances, called the *instance base*. For each test instance, the set of nearest neighbors is retrieved from the instance base. The majority label of this set is returned.

One strength of this approach is that irregular and low-frequency events are preserved in the instance base. This may prove advantageous for our task, as the choice of preposition can be highly context-specific and idiosyncratic.

Of critical importance is the distance metric between two instances, since it determines who the nearest neighbors are. We utilized ib1-ig [9], an algorithm that uses information gain to define this metric. The following section is a brief summary taken from [8].

4.1 IB1-IG

When there are n features, the distance Δ between two instances X and Y is:

$$\Delta(X,Y) = \sum_{i=1}^{n} w_i \delta(x_i, y_i)$$

where δ, the distance per feature, is defined by:

$$\delta(x_i, y_i) = \begin{cases} 0 \text{ if } x_i = y_i \\ 1 \text{ otherwise} \end{cases}$$

The weight w_i is intended to reflect the salience of the feature i. In IB1-IG, w_i is the information gain (IG) of feature i, i.e. the amount of entropy (H) reduced by the feature. In order not to favor features with more values, the "split info" ($si(f)$) is used as a normalizing factor. Formally,

$$w_i = \frac{H(C) - \sum_{v \in V_f} P(v) H(C|v)}{si(f)}$$
$$si(f) = - \sum_{v \in V_f} P(v) \log_2 P(v)$$

where C is the set of class labels (i.e., the prepositions), and V_f is the set of values for feature f.

4.2 Example

The distance metric could be understood as defining "buckets" of neighbors for each test instance. These buckets, from the nearest ones to the furthest, form the steps of the back-up sequence to be followed by the algorithm, as it searches for the set of nearest neighbors. As an illustration, we now apply the IB1-IG algorithm to the LEXICAL feature set (see §3.1).

The information gain of each feature in consideration, `V`, `N1` and `N2`, is computed on the training set. The information gain for `N1` turns out to be the greatest, followed by `N2` and then `V`. By linguistic intuition, `N1` and `N2` should be most informative for preposition generation when the lexical head is a noun. Since nouns constitute the majority among the lexical heads in our training set (see §5), it is natural that `N1` and `N2` yield the most information gain.

Table 3 shows the complete back-off sequence. Given a test instance, its closest neighbors are those training instances that match all three features (`N1+N2+V`). If such instances exist, the majority label (preposition) of these neighbors is returned. Among our test data whose lexical heads are nouns, 1111 fall into this category, and the predicted preposition is correct 78.1% of the time.

If no training instances match all three features, then the algorithm searches for training instances that match both `N1` and `N2` (`N1+N2`), since this combination yields the next largest information gain. The process continues down the back-off sequence in the left column of Table 3.

Table 3. The back-off order of the nearest-neighbor "buckets" in the Lexical feature set. The size of each bucket and its corresponding accuracy are listed below for two types of lexical heads: nouns, and verbs with argument PPs.

Nearest Neighbor	Lexical Head			
Back-off	Noun		Verb (argument PP)	
Sequence	Size	Accuracy	Size	Accuracy
`N1+N2+V`	1111	78.1%	395	82.3%
`N1+N2`	621	68.4%	243	24.3%
`N1+V`	471	57.5%	45	51.1%
`N2+V`	35	54.3%	14	78.6%
`N1`	14	21.4%	3	0%
`N2`	0	n/a	0	n/a
`V`	2	100%	0	n/a
Total	2254	71.8%	700	59.7%

5 Data

We restrict our attention to the ten most frequently occurring prepositions in the Penn Treebank [10]: *of, in, to, for, on, by, at, with, from*, and *as.*

Our test data consists of 3990 occurrences[4] of these ten prepositions in section 23 of the Penn Treebank. Statistics of the test data are presented in Table 4.

Table 4. Distribution of lexical heads in our test set, section 23 of the Penn Treebank

Lexical Head	Percentage
Verb (argument PP)	17.5%
Verb (adjunct PP)	22.9%
Noun	56.5%
Adjective	3.0%

Our training data is the Aquaint Corpus of English News Text, which consists of 10 million sentences drawn from New York Times, Xinhua News Service, and the Associated Press. Parse trees for these sentences are obtained automatically from a state-of-the-art statistical parser [11]. The distributions of the prepositions are shown in Table 5.

Correctness of the PP attachment in the training data could have been ensured by using a manually parsed corpus, such as the Penn Treebank. However, the parser is reasonably accurate with PP attachments[5], and allows us to take statistical advantage of a much larger training corpus such as Aquaint. This advantage is especially significant for the memory-based learning framework. Our

[4] Some prepositions occur in constructions such as "**as ... as**", "*because* **of**" and "*such* **as**", where their usage is quite predictable. To avoid artificially boosting the generation accuracy, we exclude such cases from our experiments.

[5] The parser achieves 82.29% recall and 81.51% precision [11] for PP modifications.

Table 5. The five most frequently occurring prepositions in the training set, tabulated according to their lexical heads

Verbs		Nouns		Adjectives	
Prep.	Frequency	Prep.	Frequency	Prep.	Frequency
in	25%	*of*	55%	*to*	27%
to	16%	*in*	15%	*of*	14%
for	11%	*for*	8%	*for*	14%
on	10%	*on*	5%	*as*	13%
with	10%	*to*	4%	*with*	11%

results may also be more realistic, since treebanks may not be available in other domains.

6 Evaluation

We conducted experiments to compare the two feature sets described in §3: Lexical and Attach. Results are summarized in Table 6.

Table 6. Preposition generation accuracy on the Lexical and Attach feature sets. The majority baseline is 28.5% (always choosing "*of*"). Results from §6.2 are upper bound estimations; results from §6.4 is our best without assuming correct attachment information in the test input. For detailed comments, see the individual sections listed in the left column.

Section	Train Set	Test Set	Verbs (argument PP)	Verbs (adjunct PP)	Nouns	Adjectives	Overall
§6.1	Lexical	Lexical	59.7%	58.6%	71.8%	75.8%	66.8%
§6.4	Attach	Lexical	**72.3%**	**60.2%**	**71.7%**	**77.5%**	**69.3%**
§6.2	Attach	Attach	75.3%	62.8%	72.5%	81.7%	71.1%
§6.2	Attach	Arg	75.3%	65.9%	n/a	n/a	n/a

6.1 Lexical Feature Set

As discussed in §4.2, information gain is greatest for the NP/ADJP Head feature (`N1`) in the Lexical feature set, followed by Prepositional Complement (`N2`), and lastly Verb Phrase Head (`V`). This sequence produces the back-off steps of nearest neighbors shown in Table 3. Please refer to this table for the rest of this section.

Nouns and Adjectives. When very similar training instances (`N1+N2+V`) are available, generation accuracy reaches a relatively high 78.1%. Performance gradually degrades as the nearest neighbors become less similar. The overall accuracy is 71.8% for nouns. The same general trend is observed for adjectives.

Verbs. Our discussion on verbs will focus on those with argument PPs. Generation accuracy is relatively high (82.3%) when similar neighbors (`N1+N2+V`) are available. However, at the next back-off level, `N1+N2`, the accuracy sharply decreases to 24.3%. This drags the overall accuracy down to 59.7%.

The poor performance when backing off to `N1+N2` is not accidental. The VP Head (`V`) feature is most relevant when an argument PP is attached to a verb. Consider the sentence "*They've never shown any inclination to spend money* **on** *production*". Among the `N1+N2` neighbors, the preposition "*for*" is the most common, due to expressions such as "*money* **for** *production*". However, the verb "*spend*", coupled with a direct object "*money*", should have signaled a strong preference for the preposition "*on*".

In other words, backing off to `V+N2` would have been more appropriate, since the word "*production*" is related more to the verb than to the `N1` noun. An obvious remedy is to use a different back-off sequence when the lexical head is a verb. However, there is no way of making this decision, precisely because the PP attachment site is not known. The ATTACH feature set is designed to address this shortcoming.

6.2 Attachment Feature Set: With Treebank

Without the benefit of attachment information, the LEXICAL feature set is limited to one back-off sequence, ignoring the underlying differences between PPs with verb and noun lexical heads. In contrast, the ATTACH feature set creates an instance base for each kind of lexical head. Each instance base can then optimize its own back-off sequence.

Performance of the ATTACH feature set depends critically on the quality of the PP attachment information. We therefore performed evaluation on the test set under three conditions. In this section, the features were extracted from the manually parsed Penn Treebank; in §6.3, they were extracted from automatically produced parse trees; in §6.4, no parse tree was assumed to be available.

Nouns and Adjectives. Information gain is greatest for Lexical Head (`H`), then Prepositional Complement (`N2`). Accuracies for both nouns and adjectives (third row in Table 6) compare favorably with the LEXICAL set, likely due to the fact that `N2` counts are no longer skewed by verb-specific usage.

Verbs. Information gain is highest for `H`, followed by `N2` and `N1`, yielding the back-off order shown in Table 7. Generation accuracy is 75.3% for verbs with argument PPs, substantially higher than the LEXICAL feature set, at 59.7%.

For those test instances with very similar training counterparts (`H+N1+N2`), the accuracy is 82.3%. This performance is comparable to the analogous category (`N1+N2+V`) in the LEXICAL feature set. The gain over the LEXICAL feature set is mainly due to the appropriate back-off to `H+N2`, which yields 66.4% accuracy. This back-off decision, in contrast to the one with the LEXICAL set, recognizes the importance of the identity of the verb.

Overall, when assuming perfect attachment information, the generation accuracy for the ATTACH feature set is 71.1% (third row in Table 6).

Table 7. Back-off order of the nearest-neighbor "buckets" for verb lexical heads in the Attach feature set. Performance of verbs with argument PPs are listed.

Nearest Neighbor	Verb (argument PP)	
Back-off Sequence	Size	Accuracy
`H+N1+N2`	389	82.3%
`H+N2`	143	66.4%
`H`	167	67.1%
`N2`	1	0%
Total	700	75.3%

Argument/Adjunct Distinction. For verbs[6], further gain in accuracy is still possible if the argument/adjunct distinction is known. Preposition generation tends to be more difficult for verbs with adjunct PPs than those with argument PPs. Since adjuncts depend less on the verb than arguments, their performance naturally suffers at the back-off to `H`. At this back-off level, arguments achieve 67.1% accuracy (see Table 7). The analogous figure for adjuncts is only 31.8%.

One case in point is the sentence "*... other snags that infuriated some fund investors* **in** *October 1987*". As an adjunct, the preposition "*in*" should be highly likely in front of the word "*October*". The back-off to `H`, however, wrongly predicts "*by*" based on statistics associated with the verb "*infuriated*".

Suppose the argument/adjunct distinction is known in the test data, and that the back-off from `H+N2` is changed from `H` to `N2` when the PP is an adjunct. The performance for adjuncts would then rise to 65.9% (last row in Table 6), an absolute improvement of 3%.

6.3 Attachment Feature Set: With Automatically Derived Parse Trees

In the previous section, where perfect attachment information is available, the overall generation accuracy reaches 71.1%. This section considers the use of automatically parsed sentences [11] rather than the Penn Treebank. The result should still be interpreted as an upper bound, since the parsing was performed on sentences with the correct prepositions in place.

When the Attach features are extracted from these parse trees, the overall generation accuracy decreases to 70.5%. It would be interesting to observe how much further the accuracy would degrade if sentences with preposition errors are fed to the parser. Making a meaningful comparison might prove difficult, however, since one needs to simulate how the test sentences would have been written by non-native speakers of English.

Instead, we now discuss some techniques which, without relying on attachment annotation in input sentences, could still help improve the accuracy.

[6] We consider verbs only, since "it is difficult to consistently annotate an argument/adjunct distinction" [12] for nouns in the Penn Treebank.

6.4 Attachment Feature Set: Without Parse Trees

For texts with lots of grammatical errors, parsing could be challenging, making it difficult to obtain attachment information. Lexical features, however, can be extracted more robustly. Could test instances with only Lexical features still leverage an instance base with Attach features?

A significant portion of prepositional phrases, in fact, have no ambiguity in their attachment site; for example, when a verb is immediately followed by a preposition, or when an `N1` noun occurs at the beginning of the sentence. The unambiguous test instances, then, can take advantage of the Attach instance base, while the rest are processed as usual with the Lexical instance base. This simple mechanism improves the overall accuracy from 66.8% to 68.7%.

For the ambiguous instances[7], their performance on the Lexical instance base still has room for improvement. As we have seen in §6.1, the back-off decision is crucial when fully matched instances (`N1+N2+V`) are not available. Instead of always backing off to `N1+N2`, entropy statistics can help make more informed choices.

Three sets of nearest neighbors — `N1+N2`, `N1+V` and `N2+V` — are the back-off options. If the lexical head is a verb, for example, one may expect the back-off sets involving `V` to have relatively low entropy, since the distribution of their prepositions should be more constrained. One reasonable approach is to back-off to the set with the lowest entropy. This procedure raises the overall accuracy to 69.3% (second row in Table 6), which is within 2% of the upper bound.

7 Conclusions

We have shown that knowledge of the PP attachment site can improve accuracy in preposition generation. In a memory-based learning framework, the improvement is especially substantial when similar training instances are not available and a back-off decision must be made.

For noisy texts, such as input to a grammar checker, PP attachment sites may not be readily available. In these cases, attachment information in training data can still boost generation accuracy to within 2% of the upper bound.

Acknowledgments

We thank Viggo Kann, Stephanie Seneff, Martin Volk, as well as the two anonymous reviewers, for their helpful discussions.

[7] Another potential approach is to first disambiguate the PP attachment site, i.e., determine whether `V` or `N1` is to be assigned as the lexical head `H`. The instance base with Attach features can then be used as before. We have not explored this approach, since literature on PP attachment disambiguation suggests that the preposition identity is one of the most important features [13].

References

1. Bitchener, J., Young, S., Cameron, D.: The effect of different types of corrective feedback on esl student writing. Journal of Second Language Writing 14, 191–205 (2005)
2. Chodorow, M., Tetreault, J., Han, N.R.: Detection of grammatical errors involving prepositions. In: Proc. ACL-SIGSEM Workshop on Prepositions, Prague, Czech Republic (2007)
3. Eeg-Olofsson, J., Knutsson, O.: Automatic grammar checking for second language learners — the use of prepositions. In: Proc. NoDaLiDa, Reykjavík, Iceland (2003)
4. De Felice, R., Pulman, S.G.: Automatically acquiring models of preposition use. In: Proc. ACL-SIGSEM Workshop on Prepositions, Prague, Czech Republic (2007)
5. Quirk, R., et al.: A comprehensive grammar of the english language. Longman (1985)
6. Merlo, P., Esteve Ferrer, E.: The notion of argument in prepositional phrase attachment. Computational Linguistics 32, 341–378 (2006)
7. Ratnaparkhi, A., Reynar, J., Roukos, S.: A maximum entropy model for prepositional phrase attachment. In: Proc. ARPA Workshop on Human Language Technology, Plainsboro, NJ (1994)
8. Daelemans, W., van den Bosch, A., Zavrel, J.: Forgetting exceptions is harmful in language learning. Machine Learning 34, 11–41 (1999)
9. Daelemans, W., van den Bosch, A., Weijters, A.: Igtree: Using trees for compression and classification in lazy learning algorithms. Artificial Intelligence Review 11, 407–423 (1997)
10. Marcus, M., et al.: The penn treebank: Annotating predicate argument structure. In: Proc. ARPA Workshop on Human Language Technology, Plainsboro, NJ (1994)
11. Collins, M.: Head-driven statistical models for natural language parsing. Computational Linguistics 29, 589–637 (2003)
12. Bies, A., et al.: Bracketing guidelines for treebank ii style. Technical Report, University of Pennsylvania, Philadelphia, PA (1995)
13. Collins, M., Brooks, J.: Prepositional phrase attachment through a backed-off model. In: Proc. 3rd Workshop on Very Large Corpora, Cambridge, MA (1995)

EFL Learner Reading Time Model for Evaluating Reading Proficiency

Katsunori Kotani[1,2], Takehiko Yoshimi[3,2], Takeshi Kutsumi[4], Ichiko Sata[4], and Hitoshi Isahara[2]

[1] Kansai Gaidai University
16-1 Nakamiya Higashino-cho, Hirakata, Osaka, Japan
kat@khn.nict.go.jp
[2] National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto, Japan
[3] Ryukoku University
1-5 Yokotani, Seta Oe-cho, Otsu, Shiga, Japan
[4] Sharp Corporation
492 Minosho-cho, Yamatokoriyama, Nara, Japan

Abstract. We propose a reading time model for learners of English as a foreign language (EFL) that is based on a learner's reading proficiency and the linguistic properties of sentences. Reading proficiency here refers to a learner's reading score on the Test of English for International Communications (TOEIC), and the linguistic properties are the lexical, syntactic and discourse complexities of a sentence. We used natural language processing technology to automatically extract these linguistic properties, and developed a model using multiple regression analysis as a learning algorithm in combining the learner's proficiency and linguistic properties. Experimental results showed that our reading time model predicted sentence-reading time with a 22.9% error rate, which is lower than the models constructed based on linguistic properties proposed in previous studies.

1 Introduction

One of the critical issues in learning or teaching a foreign language is learners' individual differences in proficiency. Unlike first language acquisition, proficiencies in acquiring a foreign language vary greatly. Thus, a language teacher has to understand each learner's problems and help the learner contend with them. The learners' problems principally arise from lack of lexical or syntactic knowledge. For instance, if a learner encounters a lexical item the meaning of which the learner does not know, he or she has to guess the meaning based on contextual information. Reading such a sentence should take more time than reading a sentence without unknown lexical items. Given this, some learners' problems can be identified by measuring his or her reading time, because encountering unfamiliar lexical or syntactic items will interrupt the reading process [1].[1]

[1] Reading process refers to a series of understanding tasks from word meaning to sentence/ discourse meaning. It involves word recognition, syntactic parsing and semantic composition.

The reading process is typically measured with the following metrics: (i) reading time, (ii) eye-movement and (iii) brain activity [7]. Of these metrics, reading time is more easily applicable to language classrooms than the others from the viewpoint of cost. Reading time tests, which use reading time as an evaluation metric, seem to be unpopular as pedagogical tests. Recent research, however, has confirmed the reliability and validity of reading time tests as a metric of foreign language reading proficiency [15, 10].[2] Based on these findings, we decided to use reading time as a measure of a learner's reading problems.

In this paper, we present our reading time model (RT model), which functions as the baseline in identifying a sentence which might include reading problems. The RT model predicts the time required for learners of English as a foreign language (EFL) to read a sentence based on both the linguistic properties of a sentence and a learner's reading proficiency. In our approach, a learner's level of reading problems is identified by comparing a learner's actual reading time with the reading time predicted by the RT model. A remarkable difference between a learner's actual reading time and the predicted time might indicate reading problems. Without RT model, a teacher has to manually set up the reading time for a learner to read. This task is costly and time consuming, especially when there are a lot of sentences. The RT model automatically sets up the model reading time, thereby assisting a language teacher in identifying a learner's problems.

In our experiment, the RT model was derived with a multiple regression analysis, which took reading time as a dependent variable and linguistic and learners' properties as independent variables (discussed in § 2). This model was able to predict sentence-reading time with a 22.9% error rate.

2 Features

Sentence-reading time should vary depending on the linguistic properties of a sentence and a learner's reading proficiency, as previous linguistic/psycholinguistic studies have reported [2, 8]. Linguistic properties include lexical, syntactic and discourse factors. Of these factors, we picked linguistic factors, which can be automatically derived with state-of-the-art natural language processing tools, because the goal of this study is to implement the RT model into the Computer Assisted Language Learning (CALL) system. In the rest of this section, we will review the features used to construct the RT model.

2.1 Lexical Factors

The RT model uses word length and lexical difficulty as lexical factors that should affect sentence-reading time. It is supposed that the length of a word is positively correlated with its lexical difficulty, as research on readability often uses word length to determine readability [4, 16]. Based on this idea, we speculated that the length of a word should affect reading time and used word length as a lexical feature. We defined word length as the number of characters in a word.

[2] Reading time-based evaluation should not exclude comprehension test-based evaluation, and these methods are fully compatible in identifying a learner's reading problems.

Word length was not the sole factor in lexical difficulty, because some short words are hard for EFL learners to understand [12]. Therefore, we added lexical difficulty scores heuristically derived based on JACET 4000 Basic Words [6], which classifies the difficulty of English vocabularies based on the empirical observations of English teachers working with Japanese EFL learners. Lexical difficulty was then rated using a ranking tool [17]. We measured lexical difficulty based on the scores derived with the tool [17], and summed the scores of words in a sentence as another lexical feature.

2.2 Syntactic Factor

The RT model uses sentence length and the number of branching nodes as syntactic factors that may affect sentence-reading time. Sentence length is supposed to be negatively correlated with readability [4, 16]. Therefore, we used sentence length as a syntactic feature in the RT model. A sentence's length was defined as the number of words it contained.

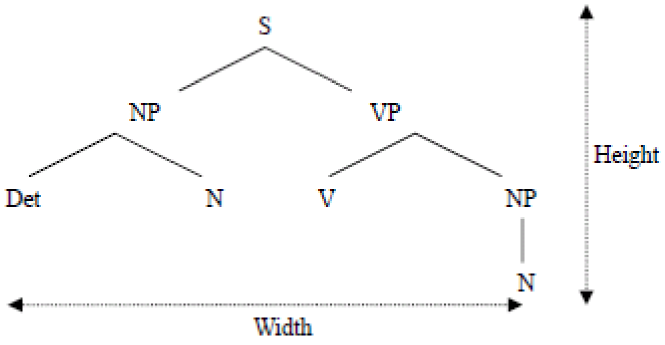


Fig. 1. Syntactic Tree

Sentence length is equivalent to the width of a syntactic tree, as shown in Figure 1. In addition to the width of the tree, we speculated that the height of a tree also indicates syntactic difficulty. To take into account both the width and height of a tree, we decided to use the number of branching nodes as another syntactic factor. Previous research [8] showed that the number of branching nodes was closely correlated with EFL learners' reading times. In addition to this empirical support, the number of branching nodes should be supported from the viewpoint of research on the garden-path sentence presented by [5]. We used the Apple Pie Parser to generate syntactic trees [14], and measured the number of branching nodes based on the trees.

2.3 Discourse Factor

In understanding the meaning of discourse, identifying anaphors is a crucial problem, and a pronoun is a typical anaphoric expression. Although there are other anaphoric expressions, such as definite expressions, we decided to use the number of pronouns as our discourse feature because it is relatively easy to measure.

The RT model picked the number of pronouns as a discourse factor. In understanding a pronoun, a learner has to identify the referent of a pronoun. Hence, the number of pronouns should indicate discourse complexity.

2.4 Learner Factors

The RT model also takes into account a learner's reading proficiency. Among various metrics for measuring the reading proficiency of EFL learners, such as comprehension test scores, word recognition time or grammatical judgment time, we used a learner's reading score on the Test of English for International Communications (TOEIC) as a learner factor.

Learner factors should involve other factors, such as non-verbal factors, e.g., the degree of interest, motivation, and background knowledge. If one wants to construct a reading model employing these non-verbal factors, these factors have to be surveyed. In this study, we focus on learners' linguistic proficiency as a learner factor, so the RT model does not take non-verbal factors into account. Some research has also suggested that non-verbal factors less clearly affect the reading time of EFL learners [10]. We leave non-verbal factors for further studies.

3 Data Set

The RT model was built with the multiple regression analysis using linguistic and EFL learners' features (stated in §2) as dependent variables. The independent variable was the reading time required for Japanese EFL learners to read English passages from TOEIC textbooks.

EFL learners' reading times were collected in the following procedure. Reading materials were extracted from a TOEIC preparation textbook [9]. Eighty-four passages were chosen and classified into two test groups, consisting either of 7 or 14 passages.

The participants were recruited from a job information website on the conditions that (i) the participant submit a TOEIC score sheet, (ii) the participant was an English learner, and (iii) the participant should live close to the site of the experiment. Of those who responded, 64 participants were chosen based on their having taken a TOEIC test in the year preceding the experiment. Each participant was randomly provided with either a 7-passage text set or a 14-passage text. 31 participants took the seven-passage text test, and 33 participants took the 14-passage text test.

Sentence-reading time was recorded with a reading process recording tool [19]. Participants read sentences displayed on a computer monitor and answered comprehension questions after reading a passage. The rate of correct answers to comprehension questions was used to determine outliers.

The reading process recording tool measures sentence-reading time in 10-millisecond units. As shown in Figure 2, a sentence appears when a participant puts the cursor on the corresponding numbered icon. A comprehension question appears

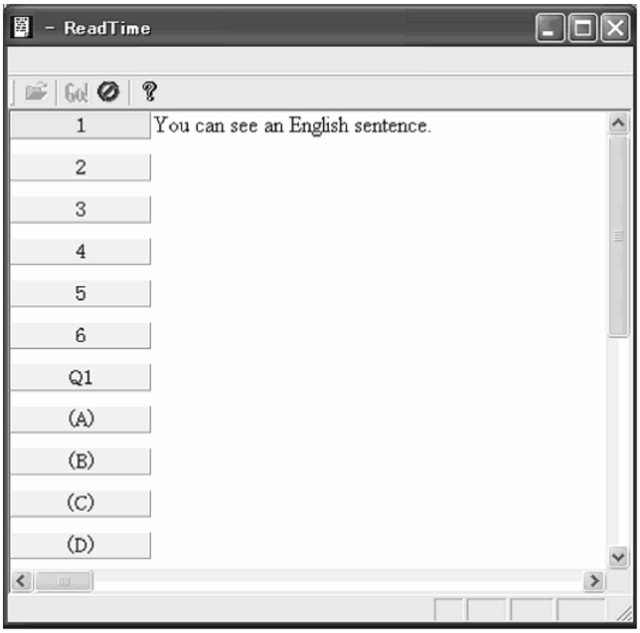


Fig. 2. Screenshot of a Reading Process Recording Tool

when the cursor is put over a Q (question) icon. In answering a comprehension question, a participant has only to click on one of the four answer icons ((A) through (D)).

Before the experiment, participants were instructed (i) to read a passage first and then answer comprehension questions, (ii) to try to understand a passage well enough to answer the comprehension questions correctly, (iii) to take as long as they needed (there was no time restriction) and (iv) to practice the reading process recording tool with several practice passages and questions.

We eliminated outliers from the collected reading time data. The data were excluded (i) if the rate of correct answers to comprehension questions was less than 70%, and (ii) if the participant's reading speed (in terms of words read per minute (WPM)) was faster than 200 WPM or slower than 70 WPM. We decided to omit the low comprehension rate data because the RT model should predict the sentence-reading time of EFL learners with a reasonable level of comprehension.[3] We excluded slow reading speed data because of the possibility of irregular reading, i.e., that a learner might have read too carefully in order to correctly understand passages. The fast reading speed data were also regarded as irregular reading for EFL learners because 200 WPM is as fast as native English speakers.[4] Hence, we supposed that such fast reading speed did not appropriately represent EFL learners' reading speed.

[3] However, there is still the problem whether the 70% correct rate was adequate for further study.

[4] The average speed for native English speakers is reported to be 200-300 WPM [3].

As a result, a total of 1807 reading times were obtained. The data consisted of 80 passages, 448 sentences, read by 61 participants. Mean age of the participants was 29.8 years old (S.D. 9.5). Of the participants, 8 were male and 53 were female. Table 1 presents the participants' TOEIC reading score (SCR) distribution. The mean TOEIC reading score of participants is 318.0, which is higher than the mean score of Japanese EFL learners, i.e., 254 to 270, according to the TOEIC technical manual [18].

Table 1. Frequency Distribution of Participants' TOEIC Reading Scores

Intervals	Frequency
$0 \leq \text{SCR} \leq 50$	0
$50 \leq \text{SCR} \leq 100$	0
$100 \leq \text{SCR} \leq 150$	3
$150 \leq \text{SCR} \leq 200$	4
$200 \leq \text{SCR} \leq 250$	11
$250 \leq \text{SCR} \leq 300$	10
$300 \leq \text{SCR} \leq 350$	6
$350 \leq \text{SCR} \leq 400$	11
$400 \leq \text{SCR} \leq 450$	11
$450 \leq \text{SCR} \leq 500$	5

4 Experiment

The RT model was built using multiple regression analysis based on all the features discussed in § 2. This section discusses the methods and results of our experiment, which tested the effectiveness of the RT model.

4.1 Methods

Of the 1807 reading times, 1627 were used as learning data for the multiple regression analysis to develop the RT model, and 180 were used as test data to verify the model. Multiple regression analysis was carried out for reading times with all factors shown in § 2 entered simultaneously.

The RT model was evaluated based on the error rate derived from Formula (1). In Formula (1), predicted value refers to reading time calculated with the RT model, and observed value is defined as actual reading time measured with the tool shown in § 3.

$$E(rror)R(ate) = \frac{|predicted \quad value \ - \ observed \quad value|}{observed \quad value} \times 100\% \tag{1}$$

First, we evaluated the RT model derived from all of the features in § 4.2, and then we compared the accuracy of the RT models built with different linguistic feature combinations in § 4.3. Finally, we compared our RT model with other models that use syntactic features as in previous studies [11] and [13] in § 4.4.

4.2 Prediction Performance of Our Model

Table 2 shows the error rate of the RT model. From the table 2 we found that most of the 180 test data showed low error rates. This tendency is clearly observed, because the relative frequency is the highest at the interval between 0% and 10%. Moreover, as the right tail is longer, the distribution of the error rate is positively skewed. The normality of the error rate was examined using the Kolmogorov-Smirnov test, and it was found that the error rate did not follow the normal distribution. The median error rate was 22.9%, and the range was 99.2.

Table 2. Error Rate (ER) Frequency Distribution Table

Intervals	Frequency	Relative Frequency (%)	Cumulative Frequency	Cumulative Relative Frequency (%)
0% ≤ ER ≤ 10%	43	23.9	43	23.9
10% ≤ ER ≤ 20%	34	18.9	77	42.8
20% ≤ ER ≤ 30%	37	20.6	114	63.3
30% ≤ ER ≤ 40%	19	10.6	133	73.9
40% ≤ ER ≤ 50%	20	11.1	153	85.0
50% ≤ ER ≤ 60%	10	5.6	163	90.6
60% ≤ ER ≤ 70%	9	5.0	172	95.6
70% ≤ ER ≤ 80%	5	2.8	177	98.3
80% ≤ ER ≤ 90%	1	0.6	178	98.9
90% ≤ ER ≤ 100%	2	1.1	180	100.0

4.3 Prediction Performance and Features of a Reading Time Prediction Model

The RT model predicts sentence-reading time based on linguistic and learner factors. To examine the linguistic effects on the RT model, we constructed RT models that used different combinations of linguistic features. All models employed weighted learner factors equally.

Table 3. Constituent Features and Error Rates of Reading Models

RT Model	Constituent Features	Error Rate (%)
RT Model 1	All Features	22.9
RT Model 2	Lexical & Learner Features	25.7
RT Model 3	Syntactic & Learner Features	24.6
RT Model 4	Discourse & Learner Features	37.2
RT Model 5	Lexical & Syntactic & Learner Features	24.2
RT Model 6	Lexical & Discourse & Learner Features	27.3
RT Model 7	Syntactic & Discourse & Learner Features	24.1

Table 3 lists the error rates of each RT model as defined by the median error rate of the model. RT Model 1, which is derived from all of the features, has the lowest error rate. The error rate seems to increase when a model lacks syntactic features, e.g., RT Models 2, 4 and 6. Given this, we suppose that syntactic features affect RT models more strongly than any other features.

4.4 Comparison of Our Reading Model with other Reading Models Built with Syntactic Features

RT models account for syntactic complexity as indicated by sentence length and the number of branching nodes. Previous studies defined syntactic complexity using other syntactic factors. A previous study [13] developed a reading model based on (i) the height of a syntactic tree, (ii) the number of noun phrases, (iii) the number of verb phrases and (iv) the number of subordinate conjunctions in a sentence. Another study [11] built a reading model using the presence or absence in a sentence of (v) relative clauses, (vi) participle clauses and (vii) *to*-infinitive clauses. We constructed two reading models using the syntactic features proposed by these previous studies [11, 13], and compared our RT model with these models regarding prediction accuracy.[5]

We found that the RT model based on features (i)-(iv) [13] had an error rate of 23.9%, and the RT model using features (v)-(vii) [11] had an error rate of 24.9%. The error rate of our RT model was 22.9%. Thus, the error rate of our RT model was 4.2% ($=\frac{23.9-22.9}{23.9}\times 100\%$) lower than that of the model [13], and 8.0% ($=\frac{24.9-22.9}{24.9}\times 100\%$) lower than that of Nagata et al. [11].

5 Related Work

Our study shares a problem on features with a study for predicting EFL learners' reading time [11]. A previous study [11] developed an RT model that computed text reading time by summing up the word recognition time of each word in a passage. Word recognition time is weighted for words appearing in particular constructions such as relative clauses, participle clauses and *to*-infinitive clauses. This is based on an assumption that these constructions are more difficult for EFL learners than other constructions. This RT model is derived with a neural network learning algorithm.

Both Nagata et al.'s model [11] and our RT model encounter an empirical problem that the prediction performance depends on the performance of natural language processing techniques because both models use syntactic features derived with a syntactic parser, which is not free from technical error. The error effect of parsing should be limited as much as possible. From this viewpoint, our syntactic features should involve fewer errors than those of Nagata et al.'s model [11]. Since our syntactic features are sentence length and the number of branching nodes, our model does not need to label syntactic nodes. By contrast, the syntactic features of Nagata et al.'s model [11] have to undergo labeling, e.g.,

[5] Note that the reading model [13] does not predict sentence-reading time but text readability, and the reading model [11] predict text reading time by summing up the word recognition time in a sentence which is weighted for words appearing in particular constructions.

relatives, participles and *to*-infinitives. This creates the possibility of technical errors due to labeling. For instance, a relative clause might be parsed as a non-relative clause, or vice versa. This kind of parsing error might degrade the prediction accuracy of the RT model. Regarding the technical errors for the labeling, our RT model should also be more robust than the model using labeled features to be.

6 Conclusion

We presented an RT model that predicts EFL learners' sentence-reading times based on linguistic properties of a sentence and a learner's reading proficiency. This is the first step toward identification of sentences that have lexical or syntactic problems that are challenging to a learner. The results of our experiment show that the RT model predicts a learner's sentence-reading time with an error rate of 22.9%.

We must still examine the prediction performance of the RT model in more detail. Then, we will pursue a more accurate reading time model.

References

1. Alderson, J.C.: Assessing Reading. Cambridge University Press, Cambridge (2000)
2. Bell, T.: Extensive reading: Speed and comprehension. The Reading Matrix, 1(1) (2001)
3. Carver, R.P.: Optimal rate of reading prose. Reading Research Quarterly 18(1), 56–88 (1982)
4. Flesch, R.: A new readability yardstick. Journal of Applied Psychology 32, 221–233 (1948)
5. Frazier, L., Rayner, K.: Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. Cognitive Psychology 14, 178–210 (1982)
6. JACET. JACET 4000 Basic Words. The Japan Association of College English Teachers, Tokyo (1993)
7. Just, M.A., Carpenter, P.A.: The Psychology of Reading and Language Comprehension. Allyn and Bacon, Newton (1987)
8. Kotani, K., et al.: Effects of syntactic factors on EFL learners' reading time. Information Technology Letters 6, 457–460 (2007)
9. Lougheed, L.: How to Prepare for the TOEIC Test: Test of English for International Communication. Barron's Educational Series, Inc., Hauppanuge, New York (2003)
10. Naganuma, N., Wada, T.T.: Measurement of English reading ability by reading speed and text readability. JLTA Journal 5, 34–52 (2002)
11. Nagata, R., et al.: A method of rating English reading skill automatically: Rating English reading skill using reading speed. Computer & Education 12, 99–103 (2002)
12. Sano, H., Ino, M.: Measurement of difficulty on English grammar and automatic analysis. IPSJ SIG Notes 117, 5–12 (2000)
13. Schwarm, S.E., Ostendorf, M.: Reading level assessment using support vector machines and statistical language models. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, pp. 523–530 (2005)

14. Sekine, S., Grishman, A.: A corpus-based probabilistic grammar with only two non-terminals. In: Proceedings of the 4th International Workshop on Parsing Technologies, pp. 216–223 (1995)
15. Shizuka, T.: The effects of stimulus presentation mode, question type, and reading speed incorporation on the reliability/validity of a computer-based sentence reading test. JACET Bulletin 29, 155–172 (1998)
16. Smith, E.A., Kincaid, P.: Derivation and validation of the automated readability index for use with technical materials. Human Factors 12, 457–464 (1970)
17. Someya, Y.: Word Level Checker: Vocabulary Profiling Program by AWK, Ver. 1.5 (2000) (consulted November 6, 2006), http://www1.kamakuranet.ne.jp/someya/wlc/wlc_manual.html
18. The Chauncery Group International, Ltd., TOEIC Technical Manual, The Chauncery Group International, Ltd., Princeton, NJ (1998)
19. Yoshimi, T., et al.: A method of measuring reading time for assessing EFL-learners' reading ability. Transactions of Japanese Society for Information and Systems in Education 22(1), 24–29 (2005)

Author Index