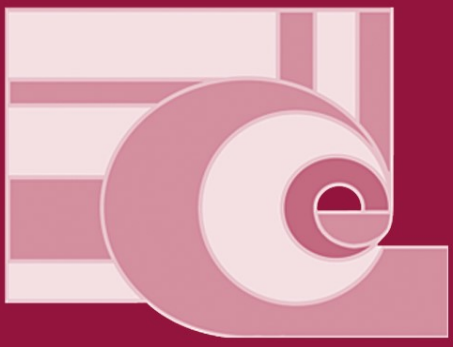
Alexander Gelbukh (Ed.)

Computational Linguistics and Intelligent Text Processing

LNCS 5449

10th International Conference, CICLing 2009
Mexico City, Mexico, March 2009
Proceedings



Springer

Lecture Notes in Computer Science 5449

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Alexander Gelbukh (Ed.)

Computational Linguistics and Intelligent Text Processing

10th International Conference, CICLing 2009
Mexico City, Mexico, March 1-7, 2009
Proceedings

Springer

Volume Editor

Alexander Gelbukh
Center for Computing Research
National Polytechnic Institute
Mexico City, 07738, Mexico
E-mail: gelbukh@gelbukh.com

Library of Congress Control Number: 2009921294

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 3-642-00381-8 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-00381-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12625819 06/3180 5 4 3 2 1 0

Preface

CICLing 2009 marked the 10^{th} anniversary of the Annual Conference on Intelligent Text Processing and Computational Linguistics. The CICLing conferences provide a wide-scope forum for the discussion of the art and craft of natural language processing research as well as the best practices in its applications.

This volume contains five invited papers and the regular papers accepted for oral presentation at the conference. The papers accepted for poster presentation were published in a special issue of another journal (see the website for more information). Since 2001, the proceedings of CICLing conferences have been published in Springer's Lecture Notes in Computer Science series, as volumes 2004, 2276, 2588, 2945, 3406, 3878, 4394, and 4919.

This volume has been structured into 12 sections:

- Trends and Opportunities
- Linguistic Knowledge Representation Formalisms
- Corpus Analysis and Lexical Resources
- Extraction of Lexical Knowledge
- Morphology and Parsing
- Semantics
- Word Sense Disambiguation
- Machine Translation and Multilinguism
- Information Extraction and Text Mining
- Information Retrieval and Text Comparison
- Text Summarization
- Applications to the Humanities

A total of 167 papers by 392 authors from 40 countries were submitted for evaluation by the International Program Committee, see Tables 1 and 2. This volume contains revised versions of 44 papers, by 120 authors, selected for oral presentation; the acceptance rate was 26.3%. It also features invited papers by

- Jill Burstein, Educational Testing Service, USA
- Ken Church, Microsoft, USA
- Dekang Lin, Google, USA
- Bernardo Magnini, Fondazione Bruno Kessler, Italy

who presented excellent keynote lectures at the conference. Publication of extended full-text invited papers in the proceedings is a distinctive feature of the CICLing conferences. What is more, in addition to presenting their invited papers, the keynote speakers organized separate vivid informal events; this is also a distinctive feature of this conference series.

The 2009 event was accompanied by a five-day pre-conference Lexicom Americas Workshop 2009, organized by Lexicography MasterClass and led by Adam

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

Country or region	Authors	Papers[1]		Country or region	Authors	Papers[1]	
	Subm	Subm	Accp		Subm	Subm	Accp
Algeria	4	1.33	–	Italy	14	5.93	2.43
Argentina	3	1.33	1	Japan	12	4.5	1
Brazil	14	4.8	–	Jordan	1	1	–
Canada	6	2.58	1.25	Korea	12	5	–
China	17	7.25	–	Lithuania	2	2	–
Colombia	3	0.75	0.75	Macao	4	1	–
Croatia	5	1	1	Mexico	44	15.38	3.88
Czech Rep.	9	5	3	Myanmar	2	1	–
Egypt	2	1	1	Norway	2	1	1
Estonia	1	1	1	Portugal	4	2	–
Finland	1	1	1	Romania	8	4	–
France	11	4.37	–	Russia	2	2	–
Germany	9	6	2	Spain	60	19.95	6.87
Greece	3	1.25	1	Sweden	3	3	1
Hong Kong	7	2.5	–	Switzerland	13	5	1
Hungary	2	1	–	Tunisia	9	4	–
India	32	13	2	Turkey	10	4	1
Iran	5	2	–	UK	9	3.32	1.32
Ireland	3	1	1	USA	40	21.75	7.5
Israel	2	1	1	Vietnam	2	2	–
				Total:	*392*	*167*	*44*

[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.

Kilgarriff, Lexicography MasterClass, UK, and Jan Pomikálek, Masaryk University, Czech Republic. The main conference program also included a discussion panel on the future of corpus design, organized by Adam Kilgarriff.

The following papers received the Best Paper Awards and the Best Student Paper Award, correspondingly:

1st Place: *Cross-Language Frame Semantics Transfer in Bilingual Corpora*, by Roberto Basili, Diego De Cao, Danilo Croce, Bonaventura Coppola, and Alessandro Moschitti (page 332);

2nd Place: *Detecting Protein-Protein Interactions in Biomedical Texts Using a Parser and Linguistic Resources*, by Gerold Schneider, Kaarel Kaljurand, and Fabio Rinaldi (page 406);

3rd Place: *Learning to Learn Biological Relations from a Small Training Set*, by Laura Alonso i Alemany and Santiago Bruno (page 418);

Student: *Enriching Statistical Translation Models Using a Domain-Independent Multilingual Lexical Knowledge Base*, by Miguel García, Jesús Giménez, and Lluís Màrquez (page 306).

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

Accepted	Submitted	Topic
13	39	Information extraction
12	31	Text mining
11	31	Clustering and categorization
8	19	Syntax and chunking (linguistics)
8	20	Statistical methods (mathematics)
8	29	Lexical resources
7	22	Information retrieval
7	24	Semantics and discourse
7	29	Formalisms and knowledge representation
7	31	Other
6	18	Word sense disambiguation
5	29	Symbolic and linguistic methods
3	11	Natural language interfaces
2	3	Emotions and humor
2	7	Text generation
2	8	Machine translation
2	9	Morphology
2	10	Summarization
1	1	Textual entailment
1	2	Parsing algorithms (mathematics)
–	2	Spell checking
–	2	POS tagging
–	3	Speech processing
–	3	Anaphora resolution

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

The best student paper was selected from those papers of which the first author was a full-time student, excluding the papers that received a Best Paper Award. 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.

Besides their high scientific level, one of the success factors of CICLing conferences is their excellent cultural program. CICLing 2009 was held in Mexico, a wonderful country, rich in culture, history, and nature. The participants of the conference had a chance to see the legendary 2000-years-old Teotihuacan pyramids, a monarch butterfly wintering site where the old pines are covered with millions of butterflies as if they were leaves, a great cave with 85-meter halls and a river flowing out of it, Aztec warriors dancing in the street in their colorful plumages, and the largest anthropological museum in the world; 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 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, Rada Mihalcea, and Ted Pedersen for their invaluable support in the reviewing process.

I express my most cordial thanks to the members of the Local Organizing Committee for their considerable contribution to making this conference become a reality. I thank the Mexican Government for providing financial support, the Mexican Society of Artificial Intelligence for valuable collaboration, and the Center for Computing Research (CIC) of the National Polytechnic Institute (IPN), Mexico, for hosting the conference.

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 2009 Alexander Gelbukh

Organization

CICLing 2009 was organized by the Natural Language and Text Processing Laboratory (nlp.cic.ipn.mx) of the Center for Computing Research (CIC) of the National Polytechnic Institute (IPN), Mexico, in collaboration with the Mexican Society of Artificial Intelligence (SMIA, www.smia.org.mx).

Program Chair

Alexander Gelbukh

Program Committee

John Atkinson
Christian Boitet
Nicoletta Calzolari
John Carroll
Kenneth Church
Dan Cristea
Walter Daelemans
Mona Talat Diab
Oren Etzioni
Alexander Gelbukh
Gregory Grefenstette
Yasunari Harada
Eva Hajičová
Graeme Hirst
Eduard Hovy
Nancy Ide
Diana Inkpen
Frederick Jelinek
Alma Kharrat
Adam Kilgarriff
Alexander Koller
Henry Lieberman
Dekang Lin
Aurelio López
Igor Mel'čuk
Rada Mihalcea
Masaki Murata
Nicolas Nicolov
Kemal Oflazer
Constantin Orasan
Ted Pedersen
Viktor Pekar
Stelios Piperidis
Fuji Ren
Fabio Rinaldi
Horacio Rodríguez
Vasile Rus
Franco Salvetti
Serge Sharoff
Grigori Sidorov
Thamar Solorio
Maosong Sun
Karin Verspoor
Manuel Vilares Ferro

Award Committee

Alexander Gelbukh
Eduard Hovy
Rada Mihalcea
Ted Pedersen
Yorick Wiks

Additional Referees

Mohamed Abdel Fattah
Mikhail Alexandrov
Muath Alzghool
Chris Biemann
Maya Carrillo Ruiz
Victor Darriba
Georgiana Dinu
Iustin Dornescu
Milagros Fernández Gavilanes
Oana Frunza
René Arnulfo García Hernández
Byron Georgantopoulos
Chikara Hashimoto
Laura Hasler
Laritza Hernández Rojas
Kaarel Kaljurand
Fazel Keshtkar
Rob Koeling
Marco Kuhlmann
Yulia Ledeneva
Joji Maeno
Nobuaki Minematsu
Tomoko Ohkuma
Juan Otero Pombo
Ryo Otoguro
Feng Pan
Michael Piotrowski
Natalia Ponomareva
Prokopis Prokopidis
Jonathon Read
Francisco Jose Ribadas Pena
Gerold Schneider
Petr Sgall
Kiyoaki Shirai
Takeo Tatsumi
Lorenzo Thione
Martin Volk
Christopher Walker
Zdenek Zabokrtsky
Carlos Mario Zapata Jaramillo

Organizing Committee

Ignacio García Araoz, CIC
Rosalía García, SMIA
Alexander Gelbukh, CIC & SMIA
Olga Kolesnikova, CIC
Yulia Ledeneva, UAEM
Raquel López Alamilla, CIC
Oralia del Carmen Pérez Orozco, CIC
Carlos Alberto Reyes García, SMIA
Grigori Sidorov, CIC & SMIA
Sulema Torres Ramos, CIC & SMIA

Website and Contact

The website of the CICLing conferences is www.CICLing.org. It contains information on the past CICLing conferences and satellite events, abstracts of all published papers, photos from past CICLing conferences, video recordings of most keynote talks, as well as the information on the forthcoming CICLing conferences. Contact options can be found on the website.

Table of Contents

Trends and Opportunities

Linguistic Knowledge Representation Formalisms

Corpus Analysis and Lexical Resources

Extraction of Lexical Knowledge

Morphology and Parsing

Semantics

Word Sense Disambiguation

Machine Translation and Multilinguism

Information Extraction and Text Mining

Information Retrieval and Text Comparison

Text Summarization

Applications to the Humanities

Has Computational Linguistics Become More Applied?

Kenneth Church

One Microsoft Way
Redmond WA 98052 USA
church@microsoft.com

Abstract. Where the field has been and where it is going? It is relatively easy to know where we have been, but harder (and more valuable) to know where we are going. The title of this paper, borrowed from Hull, Jurafsky and Martin (2008), suggests that applications have become more important, and that industrial laboratories will become increasingly prestigious.

1 Rise of Statistical Methods

Some trends are pretty well established. The Association for Computational Linguistics (ACL) is clearly accepting more statistical papers than it used to. Both Bob Moore (personal communication) and Fred Jelinek (personal communication) performed independent surveys and found a dramatic increase in statistical papers over the last couple decades [7].

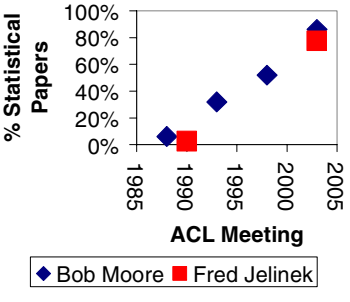


Fig. 1. Rise of Statistical Methods

Hull, Jurafsky and Martin [9], henceforth HJM, came to a similar conclusion by applying sophisticated modern methods such as LDA (Latent Dirichlet Allocation [4]) to a corpus of 14,000 documents from the ACL Anthology [3]. HJM suggest that 1988 was a particularly important year. I will refer to their suggestion as the "big bang." HJM called out two papers from 1988. It is probably not an accident that both of these papers came from well-funded industrial laboratories.

1. IBM: Brown *et al*, 1988 [5], a seminal paper on Statistical Machine Translation
2. AT&T Bell Labs: Church, 1988 [6], Part of Speech Tagging

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 1–5, 2009.

Figures 1 and 2 of HJM report that a number of statistical topics have become more popular over the last 25 years:

1. Classification
2. Probabilistic Models
3. Statistical Parsing
4. Statistical Machine Translation
5. Lexical Semantics

while other topics have declined:

1. Computational Semantics
2. Conceptual Semantics
3. Plan-Based Dialogue and Discourse

What happened in 1988? And was it really all that unprecedented? At the time, a few of us were excited about a revival of empirical traditions that pre-date the ACL:

> "The 1990s have witnessed a resurgence of interest in 1950s-style empiricism and statistical methods of language analysis. Empiricism was at its peak in the 1950s, dominating a broad set of fields ranging from psychology (behaviorism) to electrical engineering (information theory). At that time, it was common practice in linguistics to classify words not only on the basis of their meanings but also on the basis of their co-occurrence with other words. Firth, a leading figure in British linguistics during the 1950s, summarized the approach with the memorable line: 'You shall know a word by the company that it keeps' (Firth 1957)." [8, p. 1]

What makes for a successful paradigm shift? In *The Structure of Scientific Revolutions,* Kuhn [12] identified two requirements:

1. Novelty: Sufficiently unprecedented to attract an enduring group of adherents from competing modes of scientific activity
2. Opportunity: Simultaneously, sufficiently open-ended to leave all sorts of problems for the redefined group of practitioners to resolve

I like to think of Kuhn's requirements as simply good marketing. If you want to sell a new paradigm, young people are an obvious demographic to target because they aren't already committed to an established alternative. A pitch that combines novelty with opportunity will be effective with this target demographic. For students still looking for a thesis topic, it is exciting to hear about recent successes with novel methods, especially if there are opportunities for them to join in on the fun and make a contribution.

The revival of empiricism in the 1990s satisfied both of these requirements. Even though it sounds like a conflict in terms to call a revival unprecedented, the revival felt unprecedented, or at least, revolutionary (or counter-revolutionary), especially in academia, given just how much empiricism had fallen out of favor. It was very unusual in the 1980s to see a natural language paper with an evaluation section, perhaps just as unusual as it is to see a paper these days (like this one) without such a section.

But what really made empiricism take off was Kuhn's second requirement: opportunity. People think about what they can afford to think about. Opportunity is everything. When I was a student in the 1970s, empirical methods were beyond the means of most researchers, except for a relatively small minority working in well-funded industrial laboratories. But thanks to Moore's Law and data collection efforts like the Linguistics Data Consortium (LDC) http://www.ldc.upenn.edu/, by the 1990s, everyone could afford to play, including even starving students in underfunded universities. Young people found it exciting to be suddenly swamped with unprecedented opportunities.

It seemed like everything we touched was hitting pay dirt, especially at first. Progress was easy. Very simple methods were working amazing well, even on classic hard problems that had seemed intractable for decades. Lots of people were reporting encouraging numbers on lots of tasks including supposedly intractable tasks like word sense disambiguation. Recall that Bar-Hillel had given up on Machine Translation in 1960 because he couldn't see how to make progress on word sense disambiguation (see Appendix III of http://www.mt-archive.info/Bar-Hillel-1960.pdf). If word sense disambiguation was no longer intractable, then maybe Machine Translation would be back in play. Each success was opening up new avenues of work for the next generation. It was nice to see lots of encouraging numbers, but more importantly in terms of Kuhn's requirements, each accomplishment was creating more and more opportunities for the next generation of students.

2 Looking Forward: Consolidation

It is relatively easy to say where we have been, but where are we going? In [7], I suggested three possible futures:

1. We're making consistent progress. Performance numbers are going up and up and up (like Moore's Law or perhaps because of Moore's Law).
2. We're running around in circles, oscillating between rationalism and empiricism and back again every 40 years (and therefore we should expect a revival of rationalism just about now).
3. We're running off a cliff. What goes up must come down.

The first possibility looks the most promising right now. The great thing about Moore's Law is that it just keeps creating new opportunities. It is obvious how Moore's Law is making computing resources more and more affordable. But I believe Moore's Law is also driving increases in the availability of linguistic data. As the world buys more and more disks, more and more linguistic content is being created, and more and more of that content is finding its way into the hands of researchers. Disk prices (bytes per dollar) have been dropping about 1000x per decade (http://www.littletechshoppe.com/ns1625/winchest.html). I would expect availability of linguistic data to increase at about the same rate (1000x per decade).

The rising tide of data is lifting all boats. Banko and Brill [2] found that Mercer was right in 1985 when he said that "there is no data like more data" [10]. Increasing the size of the training set produced large improvements in performance. They also looked at differences across learners, but those effectives were relatively small and

Table 1. Growth of the size of available corpora

Data Source	Size	Date
Brown	1M words	1967
Cobuild	20M words	1980
Associated Press Newswire	1M words/week	
British National Corpus	100M words	1994
Google/Yahoo!/Live	20+B pages	

less reliable than "more data." We should expect improvements in performance to be indexed to increases in linguistic data. In the end, everything is indexed to Moore's Law.

Despite weaknesses in the economy, the research community is demonstrating increasing confidence in the current direction. HJM report that ACL, COLING and EMNLP are converging into a single "latent" conference. There was a time when different topics would appear in different places. EMNLP used to be smaller and more focused on empirical topics. COLING reached out for broad participation, encouraging project reports on work in progress in addition to completed results. Such differences are disappearing as the field consolidates.

HJM found increasing interest in applications. Applications used to be reserved for a separate conference, ACL Conference on Applied Natural Language Processing (popularly known as ANLP or even "Applied ACL"). Sergei Nirenburg, Program Chair for ANLP-00, provides a brief summary of the history of this conference in http://www.aclweb.org/anthology-new/docs/anlp.html. Referring to the first ANLP conference, he noted that "Of the eight organizers of the conference, only one was at the time affiliated with a university." Industrial laboratories are becoming increasingly attractive. Commercial applications such as web search are being taken more seriously than ever before.

It is extremely unlikely that rationalism will come back any time soon. That said, it would be a shame if rationalism was forgotten. I worry that we've been too successful. Have we created a Frankenstein-like monster? It sounds like a bad science fiction story, but could empirical methods take over the world and rewrite history? The methods in HJM are so powerful that they could rewrite history based on the documents in the ACL Anthology, excluding much that came before. Such methods could come to the mistaken conclusion that what happened in 1988 was unprecedented (as opposed to a revival). Similar trends are happening on the web, but at a much larger scale. The web is creating remarkable opportunities, but at the risk of forgetting much that came before the web.

Empirical methods are always subject to gaps in the training data, and such gaps are inevitable. The ACL Anthology is a remarkable resource. Few research communities are blessed with such a comprehensive collection of documents spanning such a wide range of time. Much of what is recorded in the Anthology will not be forgotten. But even so, the Anthology has limitations. In addition to leaving out much of what came before the ACL, the Anthology also doesn't cover much of what is happening elsewhere around the world. While university enrollments in America have been declining, there has been remarkable expansion elsewhere, especially in China. Studies based on the Anthology will inevitably tell us about what is in the Anthology, at

the risk of missing significant trends elsewhere. For example, http://belobog.si.umich.edu/clair/anthology/rank.cgi?type=Author&stat=H-Index& limit=30 produces a very nice list of top people in the field. But the list is stronger as a retrospective view of what used to be important than for forecasting what will be important. We need to be careful when using such methods to predict important future trends such as up and coming rising stars, as well as growth opportunities in Asia and elsewhere.

3 Concluding Remarks

The research community is demonstrating increasing confidence in the current direction. There is more agreement than there has been in a long time. Conferences are converging into a single "latent" conference. Differences between academia and industry are disappearing. The field is becoming more applied.

References

1. Bar-Hillel, Y.: Advances in Computers, vol. 1, pp. 91–163 (1960), http://www.mt-archive.info/Bar-Hillel-1960.pdf
2. Banko, M., Brill, E.: Scaling to Very Very Large Corpora for Natural Language Disambiguation. In: ACL (2001)
3. Bird, S.: Association of Computational Linguists Anthology (2008), http://www.aclweb.org/anthologyindex/
4. Blei, D., Ng, A., Jordan, M., Ng, A.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
5. Brown, P., et al.: A Statistical Approach to Machine Translation. In: COLING (1988)
6. Church, K.: A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In: ANLP (1988)
7. Church, K.: Joint talk to EMNLP 2004 and Senseval 2004 (2004)
8. Church, K., Mercer, R.: Introduction to the special issue on computational linguistics using large corpora. Computational Linguistics 19(1), 1–24 (1993)
9. Hall, D., Jurafsky, D., Manning, C.: Studying the History of Ideas Using Topic Models. In: EMNLP, pp. 363–371 (2008)
10. Jelinek (2004), http://www.lrec-conf.org/lrec2004/doc/jelinek.pdf
11. Kucera, H., Francis, N.: Computational Analysis of Present-Day American English (1967)
12. Kuhn, T.S.: The Structure of Scientific Revolutions. University of Chicago Press, Chicago (1962)

Opportunities for Natural Language Processing Research in Education

Jill Burstein

Educational Testing Service
Rosedale Road, MS 12R
Princeton, New Jersey 08540,
USA
jburstein@ets.org

Abstract. This paper discusses emerging opportunities for natural language processing (NLP) researchers in the development of educational applications for writing, reading and content knowledge acquisition. A brief historical perspective is provided, and existing and emerging technologies are described in the context of research related to content, syntax, and discourse analyses. Two systems, e-rater® and *Text Adaptor*, are discussed as illustrations of NLP-driven technology. The development of each system is described, as well as how continued development provides significant opportunities for NLP research.

Keywords: Natural language processing, automated essay scoring and evaluation, text adaptation, English language learning, educational technology.

1 Introduction

Theoretically, opportunities for natural language processing (NLP) research have existed in education in the area of reading research since the 1940's; writing research since the 1960's, and in the teaching of content knowledge since the early 1970's. While opportunities have existed for several decades, the general lack of computer-based technology proved to be an obstacle early on. In later years, when computers became increasingly more available, the lack of well-instantiated technological infrastructure (especially in schools), where educational applications could be broadly distributed and used, presented yet another obstacle – however, we can still see where the opportunities existed and how they grew over time.

Research in the area of *readability*, or *text quality* investigates the linguistic aspects of text that make a text relatively easier or more difficult to comprehend or follow. Early research examined the effect of morphological and syntactic aspects that contributed to readability, and [1] reported that features such as syllable counts (of words in a text), and sentence length were predictors of readability (text difficulty). Research in the area of readability has continued [2],[3], and has included increasingly more NLP-based investigation toward predicting grade-level fo texts [4], [5], [6],[7], or text quality in terms of discourse coherence [8],[9],[10]. In the context of text quality research, this paper will discuss a relatively new research prototype, *Text Adaptor*, that employs NLP-based methods and systems to support the creation of linguistically-appropriate reading materials for English language learners.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 6–27, 2009.

The first area of writing research related to NLP-based research was automated essay scoring with the development of Project Essay Grade [11]. While this early approach to automated essay scoring proposed in [11] was largely related to the number of words in an essay, newer methods examined linguistic aspects of text with the introduction of e-rater® [12], and Intelligent Essay Assessor [13]. E-rater examined lexical, syntactic and discourse-related text features and Intelligent Essay Asessor analyzed content through vocabulary usage with latent semantic analysis. Writer's Workbench was a pioneer in the development of automated editing and a proofreading tool [14]. The tool offered feedback primarily related to grammar and mechanics.

Intelligent tutoring systems are associated with support and evaluation of *content knowledge acquisition* [15],[16],[17],[18],[19]. The goal of intelligent tutoring is to help students work through problem sets in various domains (e.g, physics). This is another area that has involved increasingly more NLP for the purpose of evaluating students' responses as they work through problem sets in a subject area [20]. Intelligent tutoring applications make use of systems that use propositional information in responses to identify correct knowledge, given a particular problem. C-rater™ is an NLP-based system that was developed at ETS. The system is designed to evaluate the correct content of an open-ended response to a subject-area test question [21],[22]. The system generates tuple structures from sentences to examine word use, given a syntactic structure. C-rater identifies paraphrase, so that responses from different students that use different, but synonymous vocabulary, in alternate syntactic structures, can be identified as having simliar, correct meaning.

As both the availability and access to computer-based technology have become well-established, opportunities to build and distribute educational applications have vastly increased in a number of areas, including readability (text quality) research, evaluation of student writing, and assessment of student content knowledge. While the former two areas are strictly text-based, the latter, assessment of content knowledge, can be text- or speech-based [23],[24]. Text-based applications will be the focus in this paper. This paper will illustrate growth and opportunity for NLP with e-rater and *Text Adaptor* – two NLP-based, educational applications, developed to support writing and reading, respectively [12],[25].

2 E-rater®

2.1 Motivation

E-rater is an automated essay scoring system that was developed at ETS. The first version of e-rater was operationally deployed in February 1999 [12], and was used to provide one score for each of the two essays on the writing section of the Graduate Management Admissions Test (GMAT).[1] The second score for each essay was provided by an expert, human rater. Prior to e-rater deployment, GMAT used two human raters to score each essay. Since the first version of e-rater proved to be highly

[1] The GMAT is a high-stakes exam taken by individuals planning to apply to graduate business programs. E-rater no longer scores the writing section of the GMAT exam.

reliable, and e-rater was shown to agree with an expert rater as often as two expert raters agreed with each other, it made sense, at least from a cost perspective, to use e-rater for one of the two scores. So, essentially, the motivation for the first use of e-rater was a highly practical one.

While the initial motivation may have been practical, research and development around e-rater has *always* carefully attended to ensuring that the development of e-rater features reflect the *writing construct*, given a specific task.. Features that reflect the writing construct include aspects of essays that can be measured (evaluated), given a writing task. For instance, in expository writing tasks on assessments, readers might typically focus on features in writing that contribute to a high quality essay, including the writer's organization and development of ideas, the variety of syntactic constructions, the use of appropriate vocabulary, and the technical correctness of the writing in terms of its grammar, usage, and mechanics. All of these features are aspects of the writing construct that need to be evaluated for a reader to assign an appropriate rating to an essay. The linguistic nature of these features clearly illustrates that NLP methods offer natural approaches for the detection and evaluation of these features.

E-rater was designed specifically to assign a *holistic* rating. In a holistic essay scoring approach, readers take into account all aspects of writing as specified in the scoring guide, and assign a score based on their overall impression of an essay. For an automated system to simulate this approach, all feature values extracted from an essay text are combined to produce a single score that represents the overall quality of an essay. Holistic scoring often uses a six-point scale, where a score of "6" indicates the best quality essay, and a score of "1" indicates an essay of the lowest quality. E-rater scoring was modeled along a six-point scale; however, the system can be trained to score essays on a range of scoring scales.

As you will see in the remainder of this section, while the initial motivation for automated essay scoring systems was practical, research and development has always focused on capturing aspects of an essay that reflect the writing construct *and* that can be used for applications that provide diagnostic feedback and essay scoring.

2.2 E-rater v.1

For the first release of e-rater (e-rater v.1), a number of existing NLP capabilities were used that were readily available, and some new ones were developed [12]. Capabilities were used to identify and extract from essays, linguistic information related to content, syntax and discourse. In e-rater v.1 each of the *e-rater* modules identified features that corresponded to scoring guide criteria used in human scoring. These included features related to *content, syntactic variety, and organization and development (discourse structure).* These features were then used to build e-rater models for predicting essay score.

2.2.1 Content

2.2.1.1 Topical Analysis. To capture use of vocabulary, *e-rater* used content vector analyses based on the vector-space model [26]. Training essays were converted into vectors of word frequencies, and the frequencies were then transformed into word

weights, where the weight of a word was directly proportional to its frequency in the essay but inversely related to number of essays in which it appears. To calculate the topical analysis of a test essay, the essay was converted into a vector of word weights, and a search was conducted to find the training vectors most similar to it. Similarity was measured by the cosine of the angle between two vectors. For one feature, called *topical analysis by essay*, the test vector consists of all the words in the essay. The value of the feature is the mean of the scores of the most similar training vectors. The other feature, *topical analysis by argument*, evaluates vocabulary usage at the argument level. The discourse analysis (see Section 2.2.3) was used to partition the essay into its main discussion points, and a vector was created for each. These argument vectors were individually compared to the training set so that a topical analysis score could be assigned to each argument. The value for this feature was a mean of the argument scores [27].

2.2.1.2 Analysis of Lexical Complexity. While the topical analysis features compared the specific words of the test essay to the words in the scored training set, the lexical complexity features treated words more abstractly [28]. Each essay was described in terms of the number of unique words it contains, average word length, the number of words with 5 or more characters, with 6 or more characters, etc. These numerical values reflected the range, frequency, and morphological complexity of the essay's vocabulary. For example, longer words are less common than shorter ones, and words beyond 6 characters are more likely to be morphologically derived through affixation.

2.2.2 Syntactic Analysis

In order to evaluate this aspect of an essay, a shallow syntactic parser developed for *e-rater* identified several syntactic structures, such as subjunctive auxiliary verbs (e.g., would, should, might), and complex clausal structures, such as complement, infinitive, and subordinate clauses. The parsed sentences also provided the input for discourse analysis.

2.2.3 Discourse Analysis

E-rater contained a lexicon based on the conceptual framework of conjunctive relations [29] in which cue terms, such as "In summary," are classified as conjuncts used for summarizing ideas in the essay. These classifiers indicated whether or not the item was a discourse development term (e.g., "for example" and "because"), or whether it initiated a new discourse segment (e.g., "First," "Second," or "Third"). *E-rater* contained heuristics that denoted the syntactic structures in which these terms must appear to be considered discourse markers. For example, for the word "first" to be considered a discourse marker, it must not be a nominal modifier, as in the sentence, "*The first time that I read a very long essay, I thought that it was well-written*," in which "first" modifies the noun "time." Instead, "first" must occur as an adverbial conjunct, as in the sentence, "*First, it has often been noted that length is highly correlated with essay score." E-rater* uses a lexicon of cue terms and associated heuristics to automatically annotate a high-level discourse structure of each essay. The system used these annotations to partition essays into separate arguments, used these as input to the *topical analysis by argument* component (see Section 2.2.1.1).

2.2.4 Model Building and Score Prediction

In order to predict a score, *e-rater* measured more than 50 features in a training sample of approximately 270 human-scored essays. A stepwise linear regression was run to select the features that made significant contributions to the prediction of essay score. For each essay question, the result of training was a regression equation that was applied to the features of a new test essay to produce a predicted value. This value was rounded to the nearest whole number to yield the predicted score. Agreement between a human rater and e-rater, and two human raters was comparable – about 90% exact-plus-adjacent agreement. Agreement between human raters is typically measured in these terms for scoring standardized writing assessments. Only when two human raters disagreed by more than a single point was a third human rater introduced to adjudicate the score.

2.3 New NLP for *Criterion*^{SM} -- Diagnostic Feedback

In late 1999 and early 2000, there were issues with e-rater's ability to handle anomalous essays, such as off-topic essays [30]. We also began to talk to K-12 schools and community colleges about the use of e-rater in classroom settings. As an outcome of these discussions, and after a considerable development effort, the *Criterion*^{SM} online essay evaluation service was released in 2001.[2] It was intended for use in classroom settings. In the very first version of *Criterion*, teachers could select an essay topic and select a writing assignment for their students. Students could write an essay in *Criterion*, and receive an *essay score* in seconds (from e-rater v.1.), and, if they applied, *anomalous essay advisories* (described in the following section). *Criterion* embodies the *process writing* approach. This approach supports the idea that students should be able to write several drafts of a piece of writing. Consistent with this, *Criterion* allowed students to submit multiple revisions of essays, and receive a new score for each revision.

As we continued to interact and collaborate with teachers and school administrators for *Criterion* development, we consistently received feedback from teachers and school administrators who explained that in order to make the e-rater score more meaningful to students, it had to be accompanied by diagnostic feedback that more closely resembled the kind of feedback that teachers provide when grading students' writing assignments. This included information about grammar, usage, and mechanics errors, style advice, and essay-based organization and development. This led to a considerable amount of new development in the areas of detailed feedback related to students' essays. The feedback fed into a new incarnation of e-rater -- e-rater v.2, the foundation of the present version of e-rater. Both the feedback and e-rater v.2 are used in the current version of *Criterion* [31]. See Figure 1 for a screenshot of *Criterion* feedback.

2.3.1 Content-Related: Anomalous Essay Advisories

Capabilities needed to be developed to detect if an essay was anomalous. Specifically, such capabilities had to be able to determine if an essay was either *off-topic,* or the essay content was *overly repetitious* to the point where it was likely that the writer had copied-and-pasted either the text of the essay question, or sections of the essay,

[2] *Criterion* was developed and originally deployed by ETS Technologies, Inc. which was a wholly-owned subsidiary of ETS.

over-and-over again. Essentially, these advisories needed to be designed to catch e-rater user attempts to fool e-rater. It is interesting to note that the need for this arose more so due to newspaper journalists trying to fool the system than students. Most of these anomalous essays came from submissions to an e-rater demo that was released for public use. Over time, three methods were developed to detect *off-topic* and *overly repetitious* essays.

The *first method* deployed in *Criterion*, uses the follow approach to detect off-topic and overly repetitious essays.

For each essay, *z-scores* are calculated for two variables:

1. Relationship to words in a set of training essays written to an essay question
2. Relationship to words in the text of the essay question

The z-score value indicates a novel essay's relationship to the mean and standard deviation values of a particular variable based on a training corpus of human-scored essay data. The score range is usually 1 through 6, where 1 indicates a poorly written essay, and 6 indicates a well-written essay. To calculate a z-score, which ranges from 0 to 1, the mean value and the corresponding standard deviation (SD) for *maximum cosine* or *prompt cosine* are computed based on the human-scored training essays for a particular test question. The z-score indicates how many standard deviations from the mean our essay is on the selected dimension. The formula for calculating the z-score for an new novel essay is the following.

$$z\text{-}score = \frac{value - mean}{SD}$$

Z-scores are computed for the following.

1. The *maximum cosine*, which is the highest cosine value among all cosines between an unseen essay and all human-scored training essays, and
2. The *prompt cosine*, which is the cosine value between an essay and the text of essay question.

When a z-score exceeds a set threshold, it suggests that the essay is anomalous, since the threshold value indicates an acceptable distance from the mean.

A *second*, newer prompt-specific method was developed more recently and complements performance of the *maximum cosine* method. This new method helps to flag additional off-topic essays. It is based on calculating two rates for each word used in essays:

1. Proportion of word occurrences across many essay questions (generic, or essay question-independent, rate)
2. Proportion of word occurrences within a topic (essay question -specific rate).

The generic rate of occurrence for each word (G_i) across the large sample is calculated one time only from a large sample of essays across different essay questions from within one program, or within similar grade-levels. It is interpreted as the base-rate level of popularity of each word. The prompt-specific rate (S_i) is computed from

a training sample of essays that were written to the specific prompt for which an individual essay is to be compared. These two rates are used to compute an overall index for each individual essay.

$$\frac{1}{N}\sum_{i=1}^{n}\sqrt{S_i(1-G_i)}$$

Equivalently, in order to compute this index, we carry out the following steps.

1. Identify S_i and G_i values for all words in an essay based on pre-determined values from training sets.
2. For each word, compute $S_i(1-G_i)$ and take the square root. These square roots are summed over all words.
3. Multiply the sum of square roots by $1 \div N$, where N is the number of words in the essay, and the two rates are computed for all words in the essay.

A word in neither of the training samples will have a rate of 0, so a totally new word, not in either the generic or the specific sample, will also have a weight of zero ($0 \times (1 - 0) = 0$). This index can be interpreted as an average of word weights, where the weights are larger for words that appear more frequently in the prompt-specific essays, but at the same time are not frequent in other prompts. These are the words that would most contribute to the discrimination between on-topic essays and off-topic essays. The range of word weights is from 0 (when a word never appears in the specific sample and/or always appears in the generic sample of essays) to 1 (when it appears in every specific essay but never appears in the generic sample). The use of the square-root transformation in the weighting of words is designed to emphasize heavily-weighted words over low-weighted words. The classification of new essays as off- or on-topic is determined by setting a cutoff on the index values. This cutoff is based on the distribution of index values in the prompt-specific training sample.

In a *third method* for detecting off-topic essays, a training corpus *is not* required. The need for this method arose when *Criterion* began to allow teachers to introduce their own essay questions. We had no training data from teacher-created essay questions. Our topic-independent model for off-topic essay detection uses content vector analysis, and also relies on similarity scores computed between new essays and the text of the prompt on which the essay is supposed to have been written. Unlike the first and second method, this method does not rely on a pre-specified similarity score cutoff to determine whether an essay is on- or off-topic. Because this method is not dependent on a similarity cutoff, it also does not require any prompt-specific essay data for training in order to set the value of this parameter. Instead of using a similarity cutoff, our newer method uses a set of reference essay prompts, to which a new essay is compared. The similarity scores from all of the essay-prompt comparisons, including the similarity score that is generated by comparing the essay to the target prompt, are calculated and sorted. If the target prompt is ranked amongst the top few vis-a-vis its similarity score, then the essay is considered on topic. Otherwise, it is identified as off topic. This new method utilizes information that is available within *Criterion*, and does not require any additional data collection of student essays or test questions. Details for all three methods can be found in [30].

2.3.2 Diagnostic Feedback

Diagnostic feedback was developed to support teachers' requests that feedback was needed to support e-rater scores, to give the scores more meaning for students. Feedback is based on the detection of numerous errors in grammar, usage, and mechanics (*syntax*), highlights undesirable style (*content*), and identification of segments of essay-based discourse elements (*discourse*) for the student. This feedback was subsequently used to build a new version of e-rater (e-rater v.2).

2.3.2.1 Syntax

2.3.2.1.1 Grammar, Usage and Mechanics. The feedback capabilities identify five main types of errors – agreement errors, verb formation errors, wrong word use, missing punctuation, and typographical/proofreading errors. The approach to detecting violations of general English grammar is corpus-based and statistical. The system is trained on a large corpus of edited text, from which it extracts and counts sequences of adjacent word and part-of-speech pairs called *bigrams*. The system then searches student essays for bigrams that occur much less often than is expected based on the corpus frequencies.

The expected frequencies come from a model of English that is based on 30-million words of newspaper text. Every word in the corpus is tagged with its part of speech using a version of the MXPOST [32] part-of-speech tagger that has been trained on student essays. For example, the singular indefinite determiner *a* is labeled with the part-of-speech symbol AT, the adjective *good* is tagged JJ, the singular common noun *job* gets the label NN. After the corpus is tagged, frequencies are collected for each tag and for each function word (determiners, prepositions, etc.), and also for each adjacent pair of tags and function words. The individual tags and words are called unigrams, and the adjacent pairs are the bigrams. To illustrate, the word sequence, "*a good job*" contributes to the counts of three bigrams: *a*-JJ, AT-JJ, JJ-NN, which represent, respectively, the fact that the function word *a* was followed by an adjective, an indefinite singular determiner was followed by a noun, and an adjective was followed by a noun.

To detect violations of general rules of English, the system compares observed and expected frequencies in the general corpus. The statistical methods that the system uses are commonly used by researchers to detect combinations of words that occur more frequently than would be expected based on the assumption that the words are independent. These methods are usually used to find technical terms or collocations. *Criterion* uses the measures for the opposite purpose – to find combinations that occur *less often* than expected, and therefore might be evidence of a grammatical error [33]. For example, the bigram for *this desks,* and similar sequences that show number disagreement, occur much less often than expected in the newspaper corpus based on the frequencies of singular determiners and plural nouns.

The system uses two complementary methods to measure association: pointwise mutual information and the log likelihood ratio. Pointwise mutual information gives the direction of association (whether a bigram occurs more often or less often than expected, based on the frequencies of its parts), but this measure is unreliable with sparse data. The log-likelihood ratio performs better with sparse data. For this application, it gives the likelihood that the elements in a sequence are independent (we are looking for non-independent, dis-associated words), but it does not tell whether the

sequence occurs more often or less often than expected. By using both measures, we get the direction and the strength of association, and performance is better than it would otherwise be when data are limited.

Of course, no simple model based on adjacency of elements is adequate to capture English grammar, so filters are used to handle special conditions. Filters allow for low probability, but grammatical, sequences. With bigrams that detect subject-verb agreement, for example, filters check that the first element of the bigram is not part of a prepositional phrase or relative clause (e.g., *My friends in college assume.*) where the bigram *college assume* is not an error because the subject of *assume* is *friends*.

While the bigram method is used to handle a number of grammar, usage and mechanics error types, a number of error types in these categories are also implemented using a rule-based approach.

2.3.2.1.2 Confusable Words. Some of the most common errors in writing are due to the confusion of homophones, words that sound alike. In *Criterion*, we detect errors among *their/there/they're*, *its/it's*, *affect/effect* and hundreds of other such sets. For the most common of these, the system uses 10,000 training examples of correct usage from newspaper text and builds a representation of the local context in which each word occurs. The context consists of the two words and part-of-speech tags that appear to the left, and the two that appear to the right, of the confusable word. For example, a context for *effect* might be "*a typical effect is found*", consisting of a determiner and adjective to the left, and a form of the verb "BE" and a past participle to the right. For *affect*, a local context might be "*it can affect the outcome*", where a pronoun and modal verb are on the left, and a determiner and noun are on the right.

Some confusable words, such as *populace/populous,* are so rare that a large training set cannot easily be assembled from published text. In this case, generic representations are used. The generic local context for nouns consists of all the part-of-speech tags found in the two positions to the left of each noun and in the two positions to the right of each noun in a large corpus of text. In a similar manner, generic local contexts are created for verbs, adjectives, adverbs, etc. These serve the same role as the word-specific representations built for more common homophones. Thus, *populace* would be represented as a generic noun and *populous* as a generic adjective. The frequencies found in training are then used to estimate the probabilities that particular words and parts of speech will be found at each position in the local context. When a confusable word is encountered in an essay, a Bayesian classifier [34] is used to select the more probable member of its homophone set, given the local context in which it occurs. If this is not the word that the student typed, then the system highlights it as an error and suggests the more probable homophone.

2.3.2.2 Content

2.3.2.2.1 Undesirable Style. The feedback tool highlights aspects of style that the writer may wish to revise, such as the use of passive sentences, as well as very long or very short sentences within the essay. A feature of potentially undesirable style that the system detects is the presence of overly repetitious words, a property of the essay that might affect its rating of overall quality [35]. This is a feature that teachers consistently requested. The detection of overly repetitious words fits most appropriately

in the domain of content, i.e., vocabulary use. This feature is unique to Criterion, and does not exist in competitor systems.

Criterion uses a machine learning approach to finding excessive repetition. It was trained on a corpus of 300 essays in which two judges had labeled the occurrences of overly repetitious words. A word is considered as being overused if it interferes with a smooth reading of the essay. Seven features were found to reliably predict which word(s) should be labeled as being repetitious. They consist of the word's total number of occurrences in the essay, its relative frequency in the essay, its average relative frequency in a paragraph, its highest relative frequency in a paragraph, its length in characters, whether it is a pronoun, and the average distance between its successive occurrences. Using these features, a decision-based machine learning algorithm, C5.0, was used to model repetitious word use, based on the human judges' annotations. Some function words, such as prepositions and the articles *the* and *a*, were excluded from the model building. They are also excluded as candidates for words that can be assigned a repetition label.

2.3.2.3 Discourse

2.3.2.3.1 Essay-Based Discourse Elements. A well-written essay generally should contain discourse elements, which include introductory material, a thesis statement, main ideas, supporting ideas, and a conclusion. For example, when grading students' essays, teachers provide comments on these aspects of the discourse structure. The system makes decisions that simulate how teachers perform this task. Teachers may make explicit that there is no thesis statement, or that there is only a single main idea with insufficient support. This kind of feedback helps students to develop the discourse structure of their writing. While the original version of e-rater took into account the discourse cue words and terms in a text, it did not handle discourse segments of an essay. Therefore, to enhance *Criterion's* ability to handle organization and development in an essay, a system was built that identified essay based discourse elements in text. No competitor system has this text analysis feature.

For a system to learn how to identify discourse elements, humans annotated a sample of about 1400 student essays with essay-based discourse elements, based on a written annotation protocol. The annotation schema reflected the discourse structure of essay writing genres, such as *persuasive* writing where a highly-structured discourse strategy is employed. The discourse analysis component uses a decision-based voting algorithm that takes into account the discourse labeling decisions of three independent discourse analysis systems. Two of the three systems use probabilistic methods, and the third uses a decision-based approach to classify a sentence in an essay as a particular discourse element. The system labels sentences according to the discourse element to which they belong: *Introductory Material*, *Thesis Statement, Main Point, Support, Conclusion* and *Other*. The category, *Other*, typically is used for opening and closing salutations in essays that are written in a letter format. Full system details can be found in [36].

Criterion offers additional feedback that indicates if critical discourse elements are missing (e.g., *Thesis Statement*), or if more elements are desirable (e.g., the essay contains only 1 *Main Point*, and 3 *Main Points* would be desirable.)

2.4 E-rater v.2

As has been mentioned throughout the section on e-rater, a critical goal in e-rater development has been to continue to enrich the system with new features that better reflect the writing construct. When e-rater v.1 was developed, ETS researchers had access to a small number of *freeware* tools, such as part-of-speech taggers and syntactic parsers that could be used to develop e-rater features. The first version, therefore, was to some extent limited to existing tools, and new tools that could be built given time constraints and available resources. With the development of the diagnostic feedback described in the sections above, e-rater could now be enhanced with features that were considered more central to the writing construct, including errors in grammar, usage and mechanics, style features, and essay-based discourse analysis. To this end, e-rater had a complete makeover, and now includes a standard set of 8-10 features that are believed to be more representative of features associated with the writing construct for expository and persuasive essay writing.

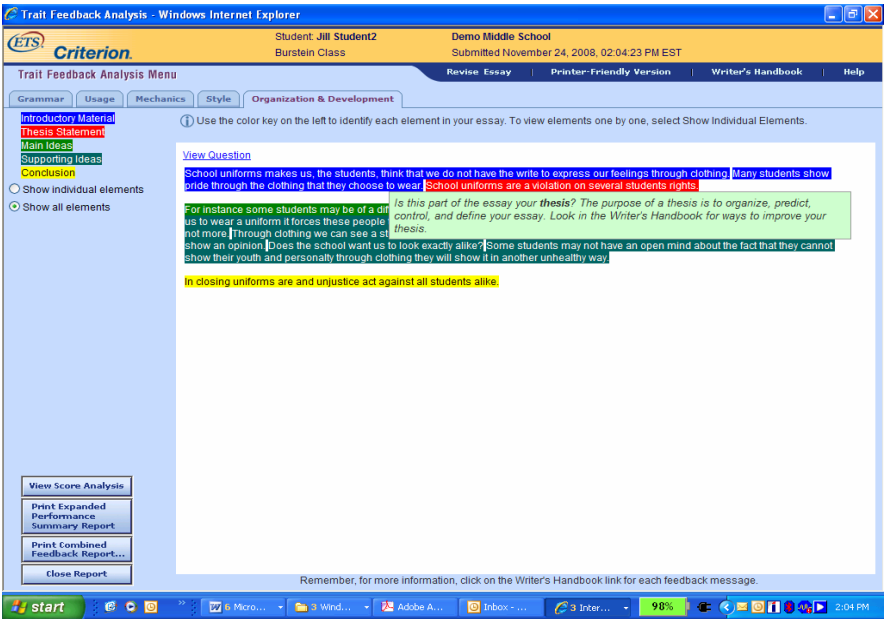


Fig. 1. *Criterion* Organization and Development Screen

E-rater v.2 forms the basis for all e-rater upgrades and uses the following information to create the standard feature set: (1) grammatical errors, (2) usage errors, (3) mechanics errors, (4) presence of discourse elements, (5) development of discourse elements, (6) style information, (7) a content vector analysis feature comparing an essay to the set of training essays (8) a content vector analysis feature comparing an essay to the set of training essays that received a score of 6, (9) average word length, and (10) a word frequency-based feature. Features (1) – (6) are derived from the

diagnostic feedback, so e-rater scores are aligned with *Criterion* feedback. The full feature set is run in a multiple regression to obtain feature weights for the final scoring model.

Two kinds of e-rater models are currently built. *Topic-specific models* are built using a set of human-scored essays on a given topic, and all ten features are typically used for these models. Topic-specific models can be built only when there are sufficient human-scored data for a topic. *Grade-level models* are built using a set of human-scored essay data written by students in a particular grade, across a number of essay topics. All features, except for the content-specific features created for (7) and (8), are used to build these models. These models can be applied to essay responses for any topic written by students at the specified grade level. No new training is required for new topics. [37] should be consulted for a full description of the e-rater v.2.

3 Text Adaptor[3]

3.1 Motivation

Subject-area academic vocabulary and general knowledge of English language skills can interfere with English language learners (ELLs) reading comprehension in K-12, content-area classrooms. Especially for ELLs beyond elementary school, large achievement gaps have been noted when the emphasis switches from *learning to read* to *reading to learn* [38]. At that point, ELLs must be able to understand grade-level, academic subject-area texts far beyond their English reading level [38],[39]. For example, social studies teachers use content materials from history, political science, sociology, geography, and economics, and each contains *specialized jargon* rooted in American culture [40]. ELLs must learn the specialized, academic vocabulary which often includes low-frequency, more difficult words. Therefore, the responsibility for educating ELLs rests not only with English language specialists or bilingual educators, but with *all* teachers. However, the number of teachers trained in effective instructional strategies to meet the needs of ELLs has not increased at the same pace as the increases in the population [41],[42].

Text adaptation [43],[44],[45] and *linguistically-targeted instruction* [38],[46] [47], [48],[49] are recommended instructional approaches that address the need to provide ELLs with improved access to academic content in text-based curriculum. *Text adaptation* involves the actual modification of a text, using techniques including, linguistic simplification (e.g., reducing complex sentence structure), elaboration of text (e.g., inserting an easier synonym adjacent to a difficult word), text summarization, and supplemental native language support. In developing *linguistically-targeted instruction,* teachers might emphasize specific linguistic features in the text, such as polysemous and morphologically-complex words, or complex syntactic structures, but do not necessarily modify the text. Using this approach, teachers might highlight a targeted linguistic feature in a text, and provide supplementary instruction about that feature via a related lesson. Both *text adaptation* and *linguistically-targeted instruction* require a strong linguistic awareness, specifically related to features that would interfere with ELL students' ability to comprehend text content.

[3] *Text Adaptor* core research and development, and the *Text Adaptor* vision presented in this paper were created in full collaboration with ETS researchers, Jane Shore and John Sabatini.

Teachers often lack training in identifying linguistic features of English that may be barriers for ELLs [50],[51],[52]. In addition, it takes considerable time to modify classroom texts, and to develop linguistically-targeted instruction for students with varying needs, even for teachers who are skilled in these practices. Consistent with the theme of this paper, effective strategies for text adaptation and linguistically-targeted instruction tie in directly to several NLP capabilities. Therefore, from *Text Adaptor's* inception, there has been considerable opportunity for NLP research. *Text Adaptor* currently uses a number of NLP capabilities to support text modification activities.

3.2 Text Adaptor and NLP

The motivation to develop *Text Adaptor* can be summarized by these three interrelated factors: (a) there is a large ELL population in K-12 classrooms in the U.S., (b) there is a lack of academic reading material where key content is accessible to ELLs, and (c) K-12 content-area teachers are not necessarily trained in methods to effectively communicate difficult academic content to ELLs.

Text Adaptor is a web-based system that was designed to provide linguistic guidance to teachers to help build their knowledge and skill, and facilitate actual development of *text adaptations* [25]. *Text Adaptor* feedback is designed to build teacher knowledge and foster the creation of more content-accessible reading materials for ELLs. The central idea is that *Text Adaptor* feedback offers linguistic insight related to linguistic features in a text that might be difficult for ELLs. Linguistic complexity can interfere with ELL students' content comprehension. The idea, then, is that teachers can use *Text Adaptor* feedback to make appropriate changes to a text, rendering the text more comprehensible to the student.

Text Adaptor's interface design and embedded functionality were inspired by academic research in the area of text adaptation, and through our early collaboration with teachers who participated in school-based pilot studies over the past three years [25]. See the screenshot of *Text Adaptor* Figure 2, below. *Text Adaptor* incorporates several NLP capabilities which were selected because (a) they are aligned with pedagogy in text adaptation practice and linguistically-targeted instructional approaches, and (b) these capabilities were immediately available and could be incorporated in the system given available resources. In light of the fact that all NLP capabilities incorporated into *Text Adaptor* were *off-the-shelf*, much opportunity still remains to build additional NLP capabilities that would enhance the current system. Our ideas for new capabilities are discussed later in this section.

Current *Text Adaptor* features highlight different kind of linguistic features in a text that address *content* and *syntax*. Given the following set of *Text Adaptor* features, (a) - (d) address text *content* issues, and (e) addresses *syntax*: (a) automated synonym detection to replace or supplement difficult words with synonyms [21], (b) antonym detection (using WordNet®) as a vocabulary lesson supplement, (c) automated text summarization [53], (d) English-to-Spanish machine translation[4], and (e) shallow parsing to identify complex sentence structures [12]. *Text Adaptor* also contains a

[4] Language Weaver's English-to-Spanish machine translation system is used: http://www.languageweaver.com.

Fig. 2. *Text Adaptor* main screen with pull-down menu of synonym options for the word *document*

hand-built, English-Spanish cognate dictionary, so that Spanish cognates can be used to supplement the English text. This is another feature that addresses text content, and currently supports the large population of native Spanish speakers in U.S. classrooms.

Users (teachers) can opt to accept *Text Adaptor* feedback to include in an adaptation. In addition, the system allows them to freely edit adaptations. They can incorporate their own ideas, and introduce new text, formatting, and highlighting into *text adaptations*. The system also retains images from the original text in the adapted version of the text. These can also be edited at the teacher's discretion. The system has a backend database where user-system interactions are stored. These data are used for research purposes, to gauge teacher performance, and to examine feature use, with system development in mind.

3.2.1 Current Text Adaptor Use

Text Adaptor is currently being piloted with two online teacher professional development programs for ELL teachers in the United States: one at a large, private university on the west coast, and another at a large, private university on the east coast. Approximately 120 teachers are participating in the pilot.

The goal of *Text Adaptor* use in teacher professional development settings is to expose teachers to linguistic complexity in text, and make them aware of how to identify and modify linguistic complexity so that text content is more accessible to an ELL

reader. Our hypothesis is that through continued exposure to linguistic complexity, teachers will develop a heightened awareness to linguistic complexity in texts which will better prepare them to develop reading materials for their ELLs. Linguistic sensitivity is one important factor that we believe will contribute to improved teacher quality.

In the scope of the pilot, teachers have completed the following activities in the following sequence: (a) a survey that elicits information about background, (b) manual adaptations based on example ELL student profiles (e.g., 7th grade native-Spanish speaker with intermediate English proficiency), (c) background readings about text adaptation, (d) *Text Adaptor* training, (e) two *Text Adaptor*-created adaptations, and (f) a perception survey to elicit teacher feedback about adaptation practice and use of *Text Adaptor*. There are control and treatment cohorts in the pilot. Control groups complete all adaptations manually, and do not receive information about the tool. Data collection and analyses are underway to evaluate if adaptation quality improves with *Text Adaptor* use. Teachers responses to the perception survey will inform future *Text Adaptor* development.

Teacher adaptations, created in the context of the pilot, will be rated by trained experts, according to a scoring protocol developed for the purpose of assigning numerical ratings to the adaptations for overall quality and for individual traits believed to contribute to overall text difficulty. *From an NLP perspective*, the relationships between the expert ratings and the linguistic features that can be automatically captured in texts (e.g., syntactic complexity, word choice, synonym use, idiom use) will be the first step toward building a model that defines "effective text adaptation" for English language learners. Continuing to build an increasingly larger corpus of rated adaptation data, created for different profiles of English language learners, could provide an extraordinary resource for NLP research related to text quality.

3.2.2 Thinking about NLP Capabilities in *Text Adaptor*

As *Text Adaptor* is currently a prototype, we see that there is opportunity to develop *Text Adaptor* even further to draw attention to potential obstacles in text in the areas of *content*, *syntax* and *discourse*. Our interactions with teachers in school settings (prior to the 2008 pilot) have informed our vision about what current opportunities related to NLP-based development may be.

3.2.2.1 Content. The presence of *polysemous words* in a text can contribute to overall text difficulty [54], [55], [56]. A polysemous word feature would help teachers become more sensitive to another aspect of vocabulary that can render a text more difficult. This is especially important with regard to academic vocabulary, since it is central to content learning. For instance, Social Studies and Science have their own sublanguages. To illustrate, let's take the senses of the word "plant." The sense, "crop," is more likely to be associated with the word "plant" in biology, and the sense "factory" is more likely to be associated with the word "plant" in Social Studies. Teachers can use this information to create an appropriate adaptation.

Several studies reviewed by [57] show a relationship between reading comprehension and knowledge of derivational morphology [43],[58],[59]. In derivational morphology, adding a suffix will change a word's part of speech (e.g., *information*

(noun) → *informational* (adjective)). In [57], the authors studied how ELLs' ability to break down words into meaningful units (popularity = "popular" + "ity") in 4th and 5th grade related to their vocabulary knowledge and reading comprehension. They reported that students who showed a greater understanding of morphology had high reading comprehension scores when holding constant their word reading fluency. They found a significant effect in the 4th grade, and a stronger effect in the 5th grade. Their findings suggest that teaching morphology might improve students' reading comprehension and language outcomes. The ability to identify and highlight *morphologically complex words* would further address vocabulary.

Non-literal expressions, such as idioms and fixed expressions can introduce difficulty into a text for ELLs. Teaching *multi-word expressions* (MWEs) is a critical part of teachers' reading comprehension curriculum [60]. MWEs can be divided into two categories: compositional and non-compositional. Let's use the expression, *red tape*, as an example. The compositional meaning of red tape is, "tape that is red," while the non-compositional meaning is "bureaucratic procedure." Highlighting MWEs in texts would point these terms out to teachers, and further draw their attention to linguistic aspects of a text that could pose difficulty.

3.2.2.2 Syntax. Sentence complexity contributes to text difficulty [1][2],[5],[6]. Further, the types of complex sentence structures that appear in texts can vary by subject domain [61]. *Text Adaptor* currently uses a shallow syntactic parser to capture the number of clauses that are in sentences. Using the parses, the system highlights passive sentences, and sentences with 1, 2 and 3 or more dependent clauses. Additional features related to syntactic complexity can be easily captured, and highlighted in texts, such as complex verb formation (e.g., past and progressive forms), complex noun phrases, and prepositions. While prepositions are not necessarily complex, they are abstract and may require further explanation, and teachers could provide this.

3.2.2.3 Discourse. Text difficulty exists when texts lack coherence [6],[62],[63], [64],[65],[66]. Specifically, if a text jumps from topic-to-topic, and/or the ideas in the text do not logically follow, this could confuse the reader. In terms of developing capabilities that predict text coherence, [13], and [67] have developed systems that examine coherence in student writing. Their systems measure lexical relatedness between text segments by using vector-based similarity between adjacent sentences. This approach to similarity scoring is in line with TextTiling [68],[69], an NLP approach used to identify the subtopic structure of a text. The issue of establishing the coherence of student essays, using the Rough Shift element (abruptness of topic shifts) of Centering Theory [70] has been addressed by [71].

In more recent work, developed a new approach for identifying text coherence using an entity-based representation that measures text coherence through the density of occurrence of vocabulary throughout a text [8],[9]. The approach is different from previous coherence algorithms as it takes into account the sentence position of vocabulary (i.e., subject, object, other). [9] applied their algorithm to relevant tasks in which the algorithm was able to predict text summary coherence and text readability (based on coherence factors). In this research, we will examine how well this new coherence algorithm predicts coherence in content-area classroom texts. Given the success of experiments in [9], useful measures might be derived to predict the relative

cohesiveness of classroom texts, and that these measures could be incorporated into *Text Adaptor* as useful indicators of the overall coherence of original and subsequent adaptations, created by teachers.

In addition, *transition words and phrases* can also contribute to text coherence. ETS has a capability that identifies transitional words and phrases in text. This capability is currently deployed in *Criterion.* In addition, it is critical from a coherence perspective that pronouns have appropriate and clear referents. *Pronoun* tools could be used to evaluate the appropriateness of co-reference in a text. Identification of unclear reference between nouns and pronouns and their locations in a text can help teachers resolve any unclear pronoun issues.

4 Discussion and Conclusions

To illustrate continued opportunities in the field of NLP and education, this paper has used system examples: e-rater®, a commercially-deployed application, and *Text Adaptor*, a research prototype. While both systems incorporate a significant number of NLP methods and tools, both systems have room to grow from an NLP perspective.

E-rater development began with a modest set of NLP-based tools that extracted content, syntactic, and discourse information from student essays to capture the kinds of text characteristics that reflect a range of writing quality. Since its initial development, new features have been developed and added to the system, always keeping in mind the goal of broadening and enriching the system's representation of the writing construct. New features in the past several years have included information about grammar, usage, and mechanics errors, style information, and essay-based discourse analysis. These features are closer to the kinds of information that teachers provide when grading papers, and the kinds of information that human raters consider when scoring writing assessments. All of this new information has been used to build and develop e-rater-produced feedback and scoring in *Criterion*, ETS' online essay evaluation service that has been used by over a million K-12 students in 3,200 schools in the United States. E-rater is also being used in other low-stakes practice settings, and in high-stakes assessments, including the Graduate Record Exam (GRE). Moving forward with e-rater development, considerable work is being done in the area of determiner and preposition error detection, especially to accommodate non-native English speakers who are more likely to make these kinds of errors [72], [73]. Both error types are implemented in *Criterion*. To further support English language learners, research is being done to identify collocation errors in student writing for inclusion in *Criterion* [74]. Additional research is also being pursued using entity-based approaches to develop discourse coherence measures that could be used to provide feedback, and to enhance e-rater scoring [8],[9].

With regard to *Text Adaptor* NLP-based development, continued collection of teacher adaptations created with *Text Adaptor* and then scored by experts, opens a couple of NLP research doors. First, increased numbers of human-rated data would give NLP researchers information about text quality that can be used to derive specific text quality measures which can then be associated with the appropriateness of a text for students at different levels of English proficiency. Second, the text quality features derived from the expert-rated adaptation data can be used to build an array of

models that reflect a range of ELL proficiency levels. While in our current research, we require *expert human raters* to evaluate teacher adaptations, over time, such models could be used to automatically rate the quality of teacher adaptations, in much the same way that students' essays are evaluated in *Criterion*. Certainly, this would be useful in teacher professional development settings for teacher instruction and assessment. It is also possible that a *Text Adaptor*-inspired test question might be used on a teacher certification exam, where teachers have to create adaptations in the context of a certification exam. These adaptations could then be scored automatically. Thinking about more pure NLP research, *Text Adaptor* can essentially be considered as an annotation tool. In our pilot studies, the multiple teacher-created adaptations of the same text that teachers create using *Text Adaptor* can be used for paraphrase research, since the texts are re-written (*paraphrased*) by different teachers. For a given text, teacher-created adaptations yield information about synonymous word use, sentence rewrites, and text summaries.

As indicated in the introduction to this paper, *only* two NLP-based educational applications were discussed in detail. Beyond these applications, however, there are a number of other NLP-based or –supported applications for education, across application domains, including, but not limited to, text- and dialogue-based intelligent tutoring systems, speech scoring systems, and readability or text quality applications. Data collected in the context of these applications can be used both for educational purposes, and for basic NLP research (e.g., paraphrase research).

Continued NLP research to develop educational applications could produce NLP-driven applications that make significant contributions to education, and to the populations (students or teachers) that they serve.

References

1. Flesch, R.: A new readability yardstick. Journal of Applied Psychology 32, 221–233 (1948)
2. MacGinitie, W.H., Tretiak, R.: Measures of sentence complexity as predictors of the difficulty of reading materials. In: Proceedings of the 77th Annual Convention of the APA (1969)
3. Chall, J.S., Dale, E.: Readability revisited: The new Dale-Chall readability formula, p. 159. Brookline Books, Cambridge (1995)
4. Collins-Thompson, K., Callan, J.: A language modeling approach to predicting reading difficulty. In: Proceedings of the HLT/NAACL (2004)
5. Schwarm, S., Ostendorf, M.: Reading level assessment using support vector machines and statistical language models. In: Proceedings of the ACL, Ann Arbor, MI, pp. 523–530 (2005)
6. Deane, P., Sheehan, K.M., Sabatini, J., Futagi, Y., Kostin, I.: Differences in text structure and its implications for assessment of struggling readers. Scientific Studies of Reading 10(3), 257–275 (2006)
7. Elhadad, N., Sutaria, K.: Mining a lexicon of technical terms and lay equivalents. In: Biological, translational, and clinical language processing, ACL, Prague, Czech Republic, pp. 49–56 (2007)
8. Barzilay, R., Lapata, M.: Modeling local coherence: An entity-based approach. In: Proceedings of the 43rd Annual Meeting of the ACL, Ann Arbor, MI, pp. 141–148 (2005)

9. Barzilay, R., Lapata, M.: Modeling local coherence: An entity-based approach. Computational Linguistics 34(1), 1–34 (2008)
10. Pitler, E., Nenkova, A.: Revisiting Readability: A Unified Framework for Predicting Text Quality. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (2008)
11. Page, E.B.: The imminence of grading essays by computer. Phi Delta Kappan 48, 238–243 (1966)
12. Burstein, J., Kukich, K., Wolff, S., Lu, C., Chodorow, M., Braden-Harder, L., Harris, M.D.: Automated scoring using a hybrid feature identification technique. In: Proceedings of the Annual Meeting of the ACL, Montreal, Canada (1998)
13. Foltz, P.W., Kintsch, W., Landauer, T.K.: Analysis of Text Coherence Using Latent Semantic Analysis. Discourse Processes 25(2-3), 285–307 (1998)
14. Macdonald, N.H., Frase, L.T., Gingrich, P.S., Keenan, S.A.: Writer's Workbench: Computer Aid for Text Analysis. IEEE Transactions on Communications, Special Issue on Communication in the Automated Office 30(1), 105–110 (1982)
15. Carbonell, J.R.: AI in CAI: An artificial intelligence approach to computer-assisted instruction. IEEE Transactions on Man-Machine Systems 11(4), 190–202 (1970)
16. Brown, J.S., Burton, R.R., Bell, A.G.: SOPHIE: A sophisticated instruction environment for teaching electronic troubleshooting (An example of AI in CAI). BBN Technical Report 2790. Bolt, Beranek, and Newman, Inc., Cambridge (1974)
17. Stevens, A.L., Collins, A.: The goal structure of a Socratic tutor. BBN Technical Report 351. Bolt, Beranek, and Newman, Inc., Cambridge (1977)
18. Burton, R.R., Brown, J.S.: An investigation of computer coaching for informal Activities. In: Sleeman, D.H., Brown, J.S. (eds.) Intelligent Tutoring Systems. Academic Press, New York (1982)
19. Clancy, W.J.: Knowledge-Based Tutoring: The GUIDON Program. MIT Press, Cambridge (1987)
20. VanLehn, K., Graesser, A.C., Jackson, G.T., Jordan, P., Olney, A., Rose, C.P.: When are tutorial dialogues more effective than reading? Cognitive Science 31(1), 3–52 (2007)
21. Leacock, C., Chodorow, M.: c-rater: Scoring of short-answer questions. Computers and the Humanities 37(4), 389–405 (2003)
22. Sukkarieh, J., Bolge, E.: Leveraging C-rater's automated scoring capability for providing instructional feedback for short constructed responses. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 779–783. Springer, Heidelberg (2008)
23. Bernstein, J.: PhonePass testing: Structure and construct, Ordinate Corporation, Menlo Park, CA (May 1999)
24. Zechner, K., Higgins, D., Xi, X.: SpeechRater^{TM}: A Construct-Driven Approach to Scoring Spontaneous Non-Native Speech. In: SLaTE 2007 (2007)
25. Burstein, J., Shore, J., Sabatini, J., Lee, Y., Ventura, M.: Developing a reading support tool for English language learners. In: Demo Proceedings of NAACL-HLT 2007, Rochester, NY (2007)
26. Salton, G.: Automatic text processing: the transformation, analysis, and retrieval of information by computer. Addison-Wesley, Reading (1989)
27. Burstein, J., Kukich, K., Wolff, S., Lu, C., Chodorow, M.: Enriching automated scoring using discourse marking. In: Proceedings of the workshop on discourse relations & discourse marking in conjunction with the ACL, Montreal, Canada (1998)
28. Larkey, L.: Automatic Essay Grading Using Text Categorization Techniques. In: Proceedings of the 21st ACM-SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, pp. 90–95 (1998)

29. Quirk, R., Sydney, G., Leech, G., Svartik, J.: A Comprehensive Grammar of the English Language, Longman, New York (1985)
30. Higgins, D., Burstein, J., Attali, Y.: Identifying off-topic student essays without topic-specific training data. Natural Language Engineering 12(2), 145–159 (2006)
31. Burstein, J., Chodorow, M., Leacock, C.: Automated essay evaluation: The Criterion Online Writing Evaluation service. AI Magazine 25(3), 27–36 (2004)
32. Ratnaparkhi, A.: A Maximum Entropy Part-of- Speech Tagger. In: Proceedings of EMNLP, University of Pennsylvania (1996)
33. Chodorow, M., Leacock, C.: An Unsupervised Method for Detecting Grammatical Errors. In: Proceedings of NAACL, pp. 140–147 (2000)
34. Golding, A.: A Bayesian Hybrid for Context-Sensitive Spelling Correction. In: Proceedings of the 3rd Workshop on Very Large Corpora, Cambridge, MA, pp. 39–53 (1995)
35. Burstein, J., Wolska, M.: Toward evaluation of writing style: Finding overly repetitive word use in student essays. In: Proceedings of the 11th Conference of the EACL, Budapest, Hungary (2003)
36. Burstein, J., Marcu, D., Knight, K.: Finding the WRITE stuff: Automatic identification of discourse structure in student essays. In: Harabagiu, S., Ciravegna, F. (eds.) IEEE Intelligent Systems: Special Issue on Advances in Natural Language Processing, vol. 18(1), pp. 32–39 (2003)
37. Attali, Y., Burstein, J.: Automated essay scoring with e-rater v.2. Journal of Technology, Learning, and Assessment 4(3) (2006)
38. Francis, D., Rivera, M., Lesaux, N., Keiffer, M., Rivera, H.: Practical guidelines for the education of English language learners: Research based recommendations for instruction and academic interventions. Center on Instruction, Portsmouth (2006) (retrieved April 25, 2008), http://www.centeroninstruction.org/files/ELL1-Interventions.pdf
39. Calderón, M.: Curricula and methodologies used to teach Spanish-speaking limited English proficient students to read English. In: Slavin, R.E., Calderón, M. (eds.) Effective programs for Latino students, pp. 251–305. Lawrence Erlbaum, Mahwah (2001)
40. National Council for the Social Studies. Expectations of excellence: Curriculum standards for social studies, Washington, DC (1994)
41. Calderón, M., Minaya-Rowe, L.: Teaching reading, oral language and content to English language learners – How ELLs keep pace with mainstream students. Corwin Press, Thousand Oaks (2007)
42. Gándara, P., Maxwell-Jolly, J., Driscoll, A.: Listening to teachers of English language learners: A survey of California teachers' challenges, experiences, and professional development needs. The Regents of the University of California, Sacramento (2005) (retrieved September 14, 2007), http://www.tyg.jp/tgu/school_guidance/bulletin/K14/images/nishimura.pdf
43. Nagy, W.E., Berninger, V.W., Abbott, R.D.: Contributions of morphology beyond phonology to literacy outcomes of upper elementary and middle-school students. Journal of Educational Psychology 98(1), 134–146 (2006)
44. August, D.: Supporting the development of English literacy in English language learners: Key issues and promising practices (Report No. 61). The John Hopkins University, Center for Research on the Education of Students Placed at Risk, Baltimore (2003), http://www.cde.state.co.us/cdesped/download/pdf/ELL_SupportDevelopEngLangLit.pdf

45. Echevarria, J., Vogt, M., Short, D.: Making content comprehensible for English language learners: The SIOP model. Pearson Education, Inc., New York (2004)
46. Scarcella, R.: Academic English: A Conceptual Framework. University of California, Linguistic Minority Research Group, University of California, Irvine, Technical Report 2003-1 (2003)
47. Calderón, M.: Teaching Reading to English Language Learners, Grades 6-12: A Framework for Improving Achievement in the Content Areas. Corwin Press, Thousand Oaks (2007)
48. Schleppegrell, M.: The Linguistic Challenges of Mathematics Teaching and Learning: A Research Review. Reading and Writing Quarterly 23, 139–159 (2007)
49. Kieffer, M.J., Lesaux, N.K.: Breaking down words to build meaning: Morphology, vocabulary, and reading comprehension in the urban classroom. The Reading Teacher 61, 134–144 (2007)
50. Adger, C.T., Snow, C., Christian, D.: What teachers need to know about language. Center for Applied Linguistics, Washington (2002)
51. Calderón, M., August, D., Slavin, R., Cheung, A., Durán, D., Madden, N.: Bringing words to life in classrooms with English language learners. In: Hiebert, A., Kamil, M. (eds.) Research and development on vocabulary. Lawrence Erlbaum, Mahwah (2005)
52. Hiebert, A., Kamil, M. (eds.): Research and development on vocabulary. Lawrence Erlbaum, Mahwah (2005)
53. Marcu, D.: The Theory and Practice of Discourse Parsing and Summarization. MIT Press, Cambridge (2000)
54. Gernsbacher, M.A., Faust, M.: The mechanism of suppression: A component of general comprehension skill. Journal of Experimental Psychology: Learning, Memory, & Cognition 17, 245–262 (1991)
55. Kang, H.-W.: How can a mess be fine? Polysemy and reading in a foreign language. Mid-Atlantic Journal of Foreign Language Pedagogy 1, 35–49 (1993)
56. McNamara, D.S., McDaniel, M.: Suppressing irrelevant information: Knowledge activation or inhibition? Journal of Experimental Psychology: Learning, Memory, & Cognition 30, 465–482 (2004)
57. Kieffer, M.J., Lesaux, N.K.: Breaking down words to build meaning: Morphology, vocabulary, and reading comprehension in the urban classroom. The Reading Teacher 61, 134–144 (2007)
58. Carlisle, J.F.: Awareness of the structure and meaning of morphologically complex words: Impact on reading. Reading and Writing: An Interdisciplinary Journal 12, 169–190 (2000)
59. Freyd, P., Baron, J.: Individual differences in acquisition of derivational morphology. Journal of Verbal Learning and Verbal Behavior 21, 282–295 (1982)
60. Palmer, B.C., Brooks, M.A.: Reading until the cows come home: Figurative language and reading comprehension. Journal of Adolescent and Adult Literacy 47(5), 370–379 (2004)
61. Schleppegrell, M., de Oliveira, L.C.: An integrated language and content approach for history teachers. Journal of English for Academic Purposes 5(4), 254–268 (2006)
62. Kintsch, W.: The use of knowledge in discourse processing: A construction-integration model. Psychological Review 95, 163–182 (1988)
63. Kintsch, W.: Comprehension: A paradigm for cognition. Cambridge University Press, Cambridge (1998)
64. McNamara, D.S.: Reading both high and low coherence texts: Effects of text sequence and prior knowledge. Canadian Journal of Experimental Psychology 55, 51–62 (2001)
65. McNamara, D.S., Kintsch, W.: Learning from text: Effects of prior knowledge and text coherence. Discourse Processes 22, 247–287 (1996)

66. Van Dijk, T.A., Kintsch, W.: Strategies of discourse comprehension. Academic Press, New York (1983)
67. Weimer-Hastings, P., Graesser, A.: Select-a-Kibbutzer: A computer tool that gives meaningful feedback on student compositions. Interactive Learning Environments 8(2), 149–169 (2000)
68. Hearst, M.A., Plaunt, C.: Subtopic structuring for full-length document access. In: Proceedings of the Association of Computational Linguistics, Special Interest Group on Information Retrieval, pp. 59–68 (1993)
69. Hearst, M.A.: Text Tiling: Segmenting text into multi-paragraph subtopic passages. Computational Linguistics 23(1), 33–64 (1997)
70. Grosz, B., Joshi, A., Weinstein, S.: Centering: A framework for modeling the local coherence of discourse. Computational Linguistics 21(2), 203–226 (1995)
71. Miltsakaki, E., Kukich, K.: Automated evaluation of coherence in student essays. In: Proceedings of the 2nd International Conference on Language Resources & Evaluation, Athens, Greece (2000)
72. Han, N.-R., Chodorow, M., Leacock, C.: Detecting errors in English article usage by non-native speakers. Natural Language Engineering 12(2), 115–129 (2006)
73. Tetreault, J., Chodorow, M.: The Ups and Downs of Preposition Error Detection in ESL Writing. In: COLING, Manchester, UK (2008)
74. Futagi, Y., Deane, P., Chodorow, M., Tetreault, J.: A computational approach to detecting collocation errors in the writing of non-native speakers of English Computer Assisted Language Learning 21(4), 353–367 (2008)

Information Structure in a Formal Framework
Unification-Based Combinatory Categorial Grammar

Maarika Traat

University of Tartu, Estonia
Institute of Computer Science
maarika.traat@ut.ee

Abstract. This paper presents Unification-based Combinatory Categorial Grammar (UCCG): a grammar formalism that combines insights from Combinatory Categorial Grammar with feature structure unification. Various aspects of information structure are incorporated in the compositional semantics. Information structure in the semantic representation is worked out in enough detail to allow for the determination of accurate placement of pitch accents, making the representation a suitable starting point for speech generation with context appropriate intonation. UCCG can be used for parsing and generating prosodically annotated text, and uses a semantic representation that is compatible with the currently available 'off-the-shelf' automatic inference tools. As such the framework has the potential to advance spoken dialogue systems.

1 Introduction

Information structure plays a crucial role in ensuring the coherence of a text or discourse. As such, its incorporation could improve the performance of a variety of natural language applications. Yet actual computational systems ignore it almost entirely. There are two main obstacles for its inclusion. Research into information structure tends to concentrate on specific linguistic phenomena, while its overall effect on compositional semantics applicable for a range of sentences is rarely worked out in enough detail to be useful for computational implementation. The second problem is that the formalizations that describe the semantic impact of information structure tend to use higher-order logic [1,2,3], which limits the use of inference in practice [4]. There is a tight connection between information and intonation. Thus, the inclusion of information structure in a formal grammar framework paves the way for improvements in the quality of intonation in systems where output is generated from semantic representations.

This paper presents Unification-based Combinatory Categorial Grammar (UCCG), which integrates aspects of Combinatory Categorial Grammar [3], Unification Categorial Grammar [5,6], and Discourse Representation Theory [7]. It offers a compositional analysis of information structure and a semantics compatible with first-order logic. The use of feature structure unification to combine grammatical categories makes UCCG easy to implement computationally, and

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 28–40, 2009.

allows for the integration of prosodic information in the semantics in a transparent and systematic way. UCCG provides a link between prosodically annotated text and semantics that includes information structure. As such, it has the potential to improve speech generation with context appropriate intonation in spoken dialogue systems. An early, less complete version of the work presented here was published in [8]. An in-depth treatment of the subject is available in [9].

2 Setting the Ground

2.1 Categorial Grammars

Categorial Grammars (CG) [10] are lexicalized theories of grammar: each lexical entry has a functional type associated with it. The functional type or *category* determines the ability of the lexical item to combine with other linguistic structures. A set of rules specifies the syntactico-semantic operations that can be performed on categories.

Combinatory Categorial Grammar (CCG) is a generalization of CG [3]. While 'pure' CG only used functional application as a means of combining categories, CCG has a variety of additional operations specified: composition, substitution, type-raising and coordination. The rich machinery of combinatory rules allows CCG to cover a wide range of linguistic phenomena. Differently from the original version of CG, CCG uses directional slash notation. For building semantic representation CCG pervasively uses lambda calculus, although unification has been proposed as well. Furthermore, CCG has a built-in theory of intonation and information structure [3], that will be used as the basis for the computational treatment of the semantic contribution of information structure in this paper.

Unification Categorial Grammar (UCG) represents categories as feature structures called *signs* [5,6]. Only application rules are used to combine categories. A special ordering feature marks the directionality of the arguments of the functor category. For semantic representation, UCG uses Indexed Language, which is related to Discourse Representation Theory (DRT). The use of feature structures makes the defining of the syntax-semantics interface straightforward.

2.2 Information Structure

By *information structure* we mean the way information is 'packaged' in an utterance. In this paper the terms *theme* and *rheme* are used. Theme relates an utterance to the previous discourse, while rheme advances the discourse by either providing entirely new information or modifying the information that was previously established.

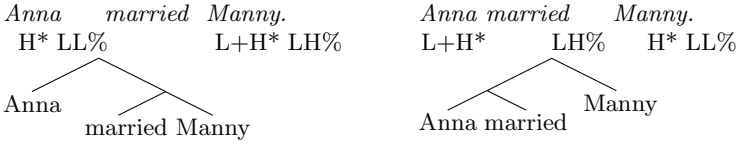
In many languages, including English, prosody is the main means of conveying information structure. In other languages additional or alternative means may be available, such as word order, or even morphology. Example (1) illustrates the connection between information structure and prosody in English. The lexical items in capital letters carry the main rhematic accent of the sentence. As

illustrated by this example, the placement of this accent determines whether the answer given to the question is appropriate or not.

What did Mark eat?
[Mark ate]$_\theta$ [a baked POTATO.]$_\rho$ (1)
*[MARK]$_\rho$ [ate a baked potato.]$_\theta$

In CCG, information structure has been implemented as part of the formalism. Steedman [3] argues that there are specific non-overlapping sets of theme and rheme pitch accents. The most common theme pitch accent is *L+H**, and the most common rheme pitch accent is *H**.[1] Each intonational phrase is delimited by a boundary tone. The most frequently occurring boundary tones are a low boundary *LL%*, and a rising boundary *LH%*.

According to the prosodical phrasing, CCG provides different parses for the same string of words, giving rise to different interpretation with respect to information structure:



(2)

In CCG categories of lexical items can either be theme-marked by a theme accent, rheme-marked by a rheme accent, or unmarked. Theme- and rheme-marked categories can freely combine with adjacent categories with the same marking or with no marking. If a theme- or a rheme-marked category combines with an unmarked category, the result category inherits the themeness or rhemeness from the marked category that participated in the combination process.

While pitch accents are seen as properties of words that carry them, boundary tones are seen as individual lexical entries which have their own category. When the boundary tone category combines with a category to its left, the result of the combination is marked as a complete intonational phrase. Phrase-marked categories can only combine with other phrase-marked categories: boundary tones function like 'stoppers' of theme and rheme categories, preventing theme and rheme to be spread over intonational phrase boundaries.

In [12] Steedman introduces boundary tone semantics, and adds a further dimension to the meaning of pitch accents. By choosing a particular boundary tone the speaker shows whether she or the hearer is responsible for, or committed to, the corresponding information unit. The further dimension attributed to pitch accents is called $\pm AGREED$. This feature reflects whether the speaker expects the hearer to share her opinion about what she says regarding the focused item.

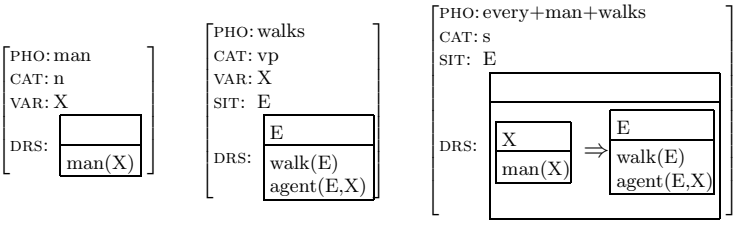
[1] The intonational notation used is due to Pierrehumbert [11]. According to her, intonational phrases are made up of pitch accent(s), a phrasal tone and a boundary tone. In Steedman's [3] representation the last two have been joined together under the name *boundary tone*. *L* stands for low pitch, and *H* for high pitch.

3 Unification-Based Combinatory Categorial Grammar

Unification-based Combinatory Categorial Grammar (UCCG) builds upon CCG and UCG. From CCG it inherits the directional slash notation, the rich set of combinatory rules, and the analysis of intonation. Similarly to UCG, linguistic data is represented as feature structures called *signs*. However, in UCCG signs there is no recursive embedding. The uniform vertical layout of features improves the readability of UCCG signs over those of UCG. The feature structures of UCCG have been enhanced with several novel features. Unlike UCG, UCCG uses standard DRT to represent semantics. In both UCG and UCCG, syntax and semantics are built up simultaneously by means of unification.

3.1 Signs

There are two kinds of signs in UCCG: basic signs and complex signs. A UCCG basic sign is a list of features that describe the syntactic and the semantic characteristics of a linguistic expression. There are three types of basic signs in UCCG, which correspond to the UCCG basic syntactic categories: noun (n), verb phrase (vp) and sentence (s). The basic signs are illustrated in Ex. 3.



(3)

Depending on the syntactic category of a linguistic expression, different features need to be specified.[2] There are three obligatory features that need to be specified regardless of the syntactic category: the phonological form (PHO), the syntactic category (CAT) and the semantic representation (DRS). Depending on the needs of a specific application and the properties of a particular language, many more features can be introduced in each sign: it is always possible to add detail to obtain a more precise description of a linguistic expression.

UCCG complex signs are formed from basic signs and CCG style slashes. Example 4 shows the complex sign corresponding to the noun phrase *every man*. The process of forming complex signs is recursive: complex signs can be combined to form even more complex signs (see e.g. Ex. 5). Box 1 provides the formal definition of the set of UCCG signs. The following terminology is used to refer to sub-parts of complex signs (illustrated in Ex. 5): in a sign of shape X/Y or X\Y, X is the *result* and Y is the *active part*.

[2] In this paper only the features that need to be specified for a given syntactic category are shown in the sign. However, in a computational implementation it may well be more convenient to use the same number of features in all signs.

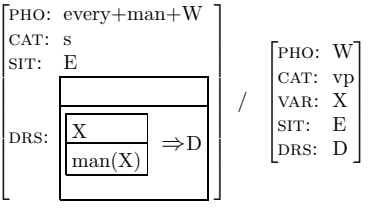
- *If A and B are signs then A/B is a complex sign.*
- *If A and B are signs then A\B is a complex sign.*
- *All basic and complex signs are UCCG signs.*

1

3.2 Features

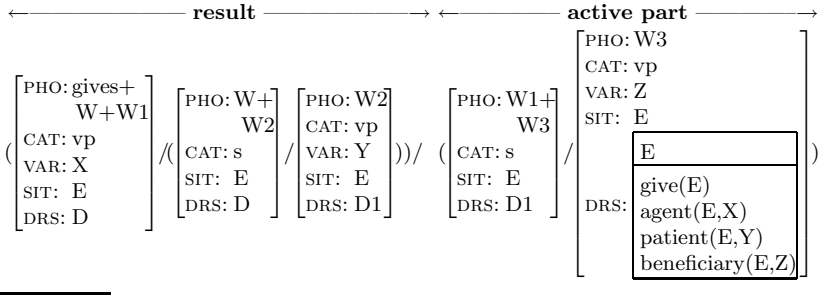
This section describes the main features used in UCCG signs.[3] The value of almost all features can be either a constant or a variable. The exceptions are the (VAR) and the (SIT) features: their value is always a variable.

The (PHO) feature name refers to the phonological form, but for now it holds the orthographic form of the linguistic expression that the given sign characterizes. The value of this feature can be of the form *word* or of the form *... + W1 + W2 + ...* where some or all the variables have been replaced by words. In basic signs the feature value contains only words and no variables. In complex signs the leftmost basic sub-sign is the part where the final string representation is being built up.



(4)

Each sign has a syntactic category. In basic signs the CAT feature represents the syntactic category of the corresponding linguistic expression, in complex signs the category is made up of the CAT features of all the sub-signs and the slashes and brackets between the sub-signs (e.g. the category of the sign in Ex. 5 is *(vp/(s/vp))/(s/vp)*). UCCG categories differ slightly from those of CCG. There is no noun phrase category *np* in UCCG. The 'type-raised' version of CCG noun phrase was needed in order to be able to account for the quantificational scope of determiners in the semantic component. Since *np* no longer existed as an autonomous category, it was possible to use the shorthand *vp* for categories of the form *s\np* (CCG's intransitive verb). Hence, the basic categories of UCCG are *n*, *vp* and *s*, and the category of a noun phrase is *s/vp* (see Ex. 4).



(5)

[3] The linguistic agreement features have been omitted for space considerations.

As pointed out previously, the value of the VAR and the SIT features is always a variable. They both provide a link between syntax and semantics. The variable in VAR is also used as a discourse referent in the semantic representation. The variable in the SIT feature refers to an event or a situation.

Finally, the DRS feature holds the DRS that corresponds to the semantics of the linguistic expression described by the sign. UCCG uses neo-davidsonian style event semantics. In neo-davidsonian semantics verbs are one-place predicates over a special *event argument*, whilst the participants in the event are presented by *θ-roles*. The value of the DRS feature can also be a variable. Likewise, any sub-DRS can be represented by a variable.

3.3 Combinatory Rules

Besides basic and complex signs, UCCG signs can also be divided into lexical and combined signs. Lexical signs originate from the lexicon: they correspond to words. Combined signs come into existence as the result of combining signs by means of combinatory rules. To date seven CCG combinatory rules have been introduced into UCCG: forward application, backward application, forward composition, backward composition, type-raising (two rules) and coordination. The remaining CCG rules can be introduced into UCCG with equal ease.

2

Forward application	$X/Y\ Y' \rightarrow X'$	————>
Backward application	$Y\ X\backslash Y' \rightarrow X'$	<————
Forward composition	$X/Y\ Y'/Z \rightarrow_{\mathbf{C}} X'/Z'$	————*C*>
Backward composition	$Y\backslash Z\ X\backslash Y' \rightarrow_{\mathbf{C}} X'\backslash Z'$	<*C*————
Type-raising A	$X \rightarrow_{\mathbf{T}} T/(T\backslash X)$	————*T*>
Type-raising B	$X \rightarrow_{\mathbf{T}} T\backslash(T/X)$	<*T*————
Coordination	$X \text{ and } X' \rightarrow_{\&} X''$	———< & >

Box 2 presents the formal definitions of the UCCG combinatory rules. The first column contains the names of the rules, the second the rules themselves, and the third the corresponding notation that will be used in derivations. The variables X, Y, Z, X', Y', Z' and T stand for signs, which can be either basic or complex. The sign X' is similar to the sign X: it has the same syntactic category as X does (unless the value of the CAT feature of one of the signs is a variable). With the exception of the type-raising rules, all UCCG rules involve the operation of unification. For example, the forward application rule says that if a sign X/Y has a sign Y' to its right, and the feature structures Y and Y' can be successfully unified, then the result is X', which is similar to X, except that its features have been updated with the values resulting from the unification of Y and Y'. Actual examples of combinations will be presented in Sect. 4.

4 Information Structure in UCCG

When introducing information structure into UCCG, we follow the theory of the meaning that prosody adds to the semantics of utterances proposed in [3,12]. There are some technical differences in the implementation as compared to CCG.

In CCG pitch accents and boundary tones are treated differently. Pitch accents are viewed as properties of words and their contributions are introduced as features on the corresponding category in the lexicon. This means that there are multiple lexical entries for every single word to cover all the pitch accents that can occur on them, as well as the case when the word is unaccented. Boundary tones, on the other hand, are viewed as independent lexical units. In UCCG, pitch accents and boundary tones are treated in a unified manner: both are autonomous lexical units. This way we avoid having to unnecessarily expand the lexicon. However, a special constraint needs to be introduced in the framework that pitch accent signs are to be combined with the signs of the words on which they occur, before any other combinations take place. Otherwise, it would be impossible to tell at a later stage of syntactic analysis, which lexical items carried the pitch accents, i.e. were focused.

To accommodate information structure, three new features are introduced into UCCG signs: INF, FOC and BND. The first two have to do with the semantic contribution of pitch accents and the last with that of boundary tones.

4.1 Pitch Accents

According to [3,12] pitch accent type determines the themeness/rhemeness of an intonational phrase, as well as the $\pm AGREED$ value. Since both of these values come from the same source (pitch accent), the new feature INF in UCCG signs takes care of both of them. The values are presented in the form $I{:}A$, where I stands for the themeness/rhemeness and A for the $\pm AGREED$ value. Variable I can assume constant values θ (theme) and ρ (rheme). Variable A can take on values $+$ and $-$. In a lexical sign, the INF feature initially contains variables: constant values are acquired through feature unification when the sign combines with a pitch accent, or another sign which already has these values determined.

Pitch accents mark the word that they occur on as being focused. Focus is the property of the particular word that the pitch accent occurs on, and may not be spread to any other words. The new feature FOC was introduced in order to handle focus marking. FOC can assume values $+$ and $-$. The value $+$ of this feature can only ever be encountered in the active part of the pitch accent sign, in other signs this value can either be a variable or have the value $-$.

The main goal is to have information structure marking in semantics. For this purpose information structure flags are used on DRS conditions.[4] In lexical signs these flags are initially set to the same variables as used in INF and FOC. When the variables in the INF and FOC features assume constant values through

[4] Information structure flags are only used on conditions that have an actual lexical exponent (not e.g. on conditions expressing semantic roles).

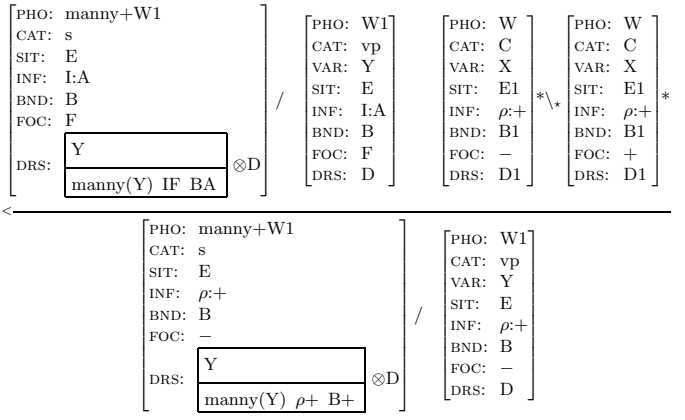


Fig. 1. Combining the word *Manny* with the pitch accent *H**

unification, whilst combining with another sign, the corresponding variables in the DRS flags are simultaneously replaced by the respective constant values.

In Fig. 1,[5] the lexical sign for the proper name *Manny* can be seen before and after combining with the rheme pitch accent *H**. Initially, its INF and FOC features hold variables. The same variables are used as flags on the DRS-condition *manny(X)*: the first and the second flag (*I* and *F*) represent the themeness/rhemeness and the focus value correspondingly, the last flag (*A*) stands for the $\pm AGREED$ value. After the combination the INF feature reflects the fact that *Manny* bears a rheme pitch accent which is of the kind $+AGREED$ (ρ:+). The first and the last flag on the DRS-conditions are accordingly set to ρ and to +. The second flag on the DRS-condition reflects the fact that the lexical item is focused. The value of the FOC feature, however, remains − as focus is not to be spread to other signs through subsequent combinations.

A UCCG pitch accent sign has the general form $S^*\backslash S'^*$, where S and S' stand for a generic basic sign. The sign for the rheme pitch accent *H** can be seen on the upper right of Fig. 1. Sign S' is similar to sign S, except for the focus value. Sign S' is the active part of the pitch accent sign. Since prosodic signs are combined with other signs via backward application, it is their active part that is unified with the lexical sign during a combination. Note that all occurrences of a variable, regardless of their location in the sign, are replaced when the variable assumes a new value in the unification process. In S, the result part of the pitch accent, the focus value is −. It is a constant value not linked to the FOC value in the active part: as such is in no way influenced by the unification of the active part with a lexical sign during a combination. Therefore, the focus flags on the DRS-conditions of the lexical sign assume the value + during the combination with a pitch accent sign due to variable unification, while the value of the FOC

[5] The sign ⊗ in the noun phrase sign in Fig. 1 stands for the merge operation, which joins two DRSs into a single DRS by combining their universes into a single universe, and their sets of DRS conditions into a single set of DRS conditions.

feature becomes −. As far as the values of the INF feature are concerned, they are the same in both S^* and S'^*.

The noun phrase sign and the pitch accent sign in Fig. 1 are combined via backward application. For the time being imagine that the pitch accent sign has the ability to adapt its shape to the lexical sign it needs to combine with and thus its category in Fig. 1 becomes (s/vp)\(s/vp). Once the unification has taken place between the active part of the pitch accent sign and the noun phrase sign, the active part of the pitch accent sign gets removed, and the changes caused by unification are stored in the result part of the pitch accent sign.

When the pitch accent sign combines with a basic sign, ordinary unification on the level of signs applies. When combining with a complex sign the operation of *recursive unification* is used. This operation can only be applied to signs of the form $S^*\backslash S'^*$: in UCCG only prosodic signs match this criterion. The meaning of the star is related, but not identical to that of Kleene star in regular expressions. In a nutshell, S^* is the shorthand for any of the signs S, $S|S'$, $S|S'|S''$, $S|S'|S''|S'''$, etc., where | stands for a slash: either a forward or a backward slash. The prime marking on the repetitions of S emphasizes the fact that the repetitions in S^* are not distinct occurrences of the same object: a new copy is made of S at each iteration. As backward application is the only type of combination that prosodic signs participate in, recursive unification is only needed in the context of this operation. Box 3 demonstrates the pseudocode algorithm of the rule of *Backward Application with Recursive Unification* (BARU).

3

```
1:    BARU:  X S*\S'* →ru X'
2:       if X is a basic sign
3:          use standard backward application X S\S' → X'
4:       else
5:          make a copy S1*\S1'*
6:          if X is of the form Y/Z
7:             apply BARU to Y:  Y S*\S'* →ru Y'
8:             apply BARU to Z:  Z S1*\S1'* →ru Z'
9:             return Y'/Z'
10:         else
11:            apply BARU to Y:  Y S*\S'* →ru Y'
12:            apply BARU to Z:  Z S1*\S1'* →ru Z'
13:            return Y'\Z'
```

As illustrated in Box 3, if a prosodic sign combines with a basic sign, standard backward application is used (lines 2,3).[6] If it is to combine with a complex sign, the prosodic sign needs to adjust its category to its argument. Therefore, a copy,

[6] The 'template' sub-signs (S and S') of a prosodic sign have all the features used in the framework. If the template combines with a sign that has fewer features, the superfluous features of the template are discarded. In a computational implementation where all signs have an equal number of features, no such problem arises.

S_1*\$S_1$'*, is made of the original prosodic sign (line 5).[7] The category of the argument X is split at its principal slash (the least embedded one), and BARU is applied to both sub-signs (lines 7,8 and 11,12). If the sign only contains one slash (A/B or $A\backslash B$), then the next step involves using standard backward application (lines 1,2,3) with both basic sub-signs, A (with the original S*\$S'$*) and B (with the copy S_1*\$S_1'$*), after which the resulting sign (A'/B' or $A'\backslash B'$) is stitched back together (lines 9,13). If the argument sign X is more complex and contains multiple slashes, further recursion is needed (lines 4-13).

Figure 1 illustrates the combination of the sign for *Manny* with the sign for the rheme accent H*. The actual process is as follows. First, standard backward application is attempted. When this does not work, a copy is made of the pitch accent sign, and the noun phrase sign is split into its s and vp components. The original pitch accent sign combines with the result part of the noun phrase sign, forming s'. The copy of the pitch accent sign is combined with the vp portion of the noun phrase sign, resulting in vp'. As the final step, s' and vp' are joined back together by the forward slash.

A final remark to be made on prosodic signs in this section concerns their use of a *multimodal slash*. In [13] Baldridge introduced multi-modal slashes into CCG that restricted the use of combinatorial rules according to slash type. The slash with the modality $\star$ only allows the use of the rule of functional application. Without this restriction, given the sign following the prosodic sign has an appropriate category, backward composition could take place between the two.[8]

4.2 Boundary Tones

Boundary tones mark a theme or a rheme as a complete intonational phrase, and add their own semantic contribution. As suggested in [12], the contribution of boundary tones to the meaning of an intonational phrase is to show whether the speaker is committed to what she is saying or not.

The boundary tone sign is similar to the pitch accent sign. The sign for the boundary *LH%* can be seen in the upper right of Fig. 2. The phrase marking is achieved by using the INF feature. In the active part of the boundary tone sign the value of INF is a variable, and can thus be unified with any value. In the result part of the sign, the INF value is a single constant ϕ. When a sign combines with a boundary tone sign, its INF feature will assume the phrase marking value ϕ. Combination with a boundary tone sign only replaces the previous INF value of the lexical sign with ϕ, but makes no changes in semantics, where theme- and rheme-marking is preserved. A phrase-marked sign can further combine only with similarly phrase-marked signs.

[7] The features in S and S' share several variables. The variables are renamed in the copy, i.e. they differ from the ones used in the original. However, if a feature value A, that is used both in S and S', is renamed B in the copy, then both S_1 and S_1' share the same variable. Constant values remain unchanged in the copy.

[8] This problem has not been addressed in the original implementation of the prosodic approach in CCG, where the boundary tone category faces this danger.

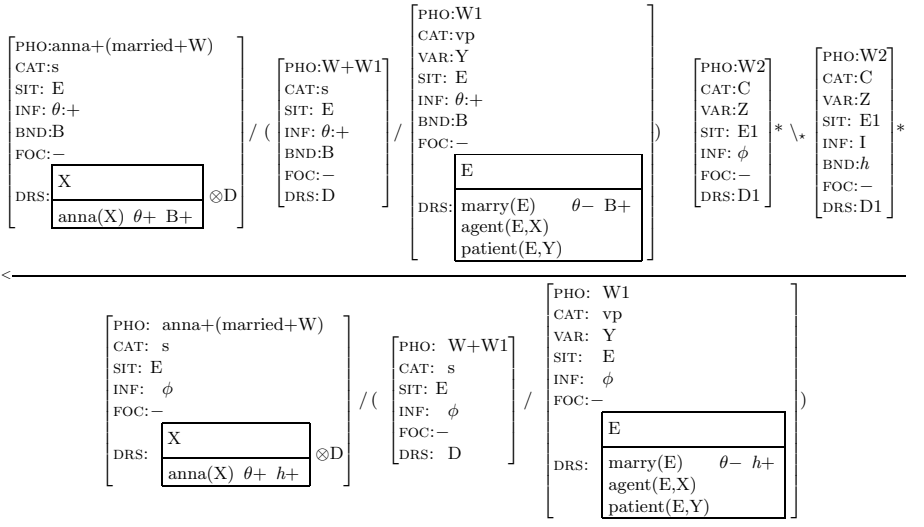


Fig. 2. Combining *Anna L+H* married* with the boundary tone *LH%*

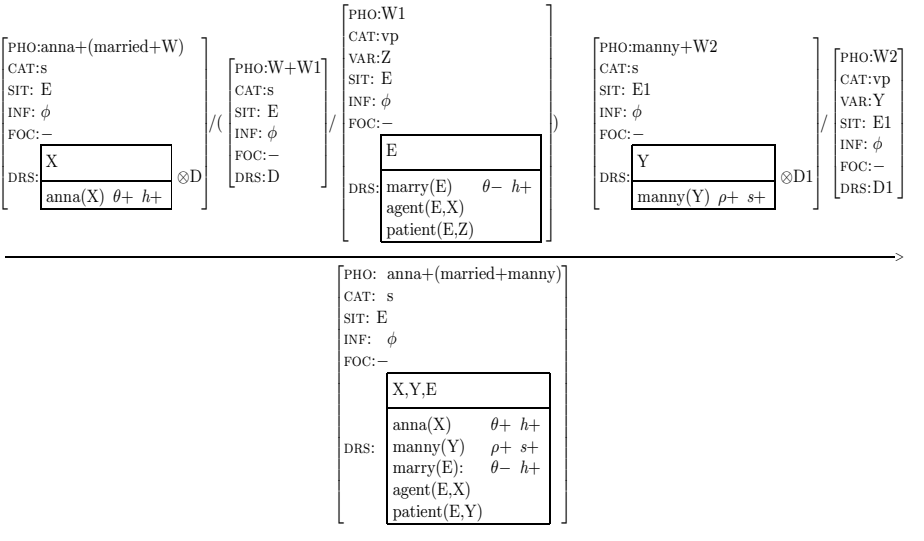


Fig. 3. Combining two complete intonational phrases: *Anna L+H* married LH%* and *Manny H* LL%*

In order to introduce boundary tone semantics in the DRS, the BND feature is incorporated in signs. This feature, and the corresponding DRS-flag, can have two constant values: *s* (speaker commitment) or *h* (hearer commitment). In boundary tone signs this feature is only present in the active part.

When a boundary tone sign combines with a complex sign, BARU is used. Figure 2 shows the combining of the theme *Anna L+H* married* with the boundary tone *LH%*. This boundary shows hearer commitment: its BND feature has the value *h*. During the combination the boundary tone flags on the DRS-conditions for the sign for *Anna L+H* married* obtain this value through unification. In the resulting sign, the BND feature is no longer present, and its INF value has become ϕ. For the sake of completeness, Fig. 3 shows the combining of the two complete intonational phrases: *Anna L+H* married LH%* and *Manny H* LL%*.

5 Conclusions and Future Work

This paper described Unification-based Combinatory Categorial Grammar, a formalism that can be used for parsing and generating prosodically annotated text. One of the key features of UCCG is the novel use of Discourse Representation Theory combined with a theory of information structure without compromising the semantic representation's compatibility with first order logic, and the state of the art automatic inference tools. We believe that UCCG has the potential to advance spoken dialogue systems, by significantly contributing to speech generation with context-appropriate intonation, and by providing support for fine-grained semantic analysis of speech that is based on intonation. Although current automatic speech recognizers do not output prosodic information, some of the state-of-the-art speech synthesizers can already handle prosodically annotated input.

A UCCG parser has been successfully implemented for a fragment of English, that takes prosodically annotated strings as input and generates DRSs marked with information structure. The next step would involve reversing the procedure and using UCCG to generate from such DRSs. It would also be interesting to try to adapt UCCG to languages other than English, which express information structure by morphological or syntactic means. Since the contribution of information structure can be specified directly in the lexicon, attributing different information structure values to different morphological forms or categories representing different word order configurations seems rather straightforward.

Acknowledgements

I would like to thank my PhD supervisors Mark Steedman and Johan Bos for their guidance during my studies, when the research presented here was conducted, and Mare Koit for her support while this paper was being written. This work was partly supported by targeted financing grant SF0180078s08, allocated by the Estonian Ministry of Education and Research.

References

1. Krifka, M.: Focus and presupposition in dynamic interpretation. Journal of Semantics 10(4), 269–300 (1993)
2. Kruijff-Korbayová, I.: The Dynamic Potential of Topic and Focus: A Praguian Approach to Discourse Representation Theory. PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague (1998)
3. Steedman, M.: The Syntactic Process. MIT Press, Cambridge (2000)
4. Blackburn, P., Bos, J.: Computational semantics. Theoria: revista de teoria, historia y fundamentos de la ciencia 18(46) (2003)
5. Zeevat, H.: Combining Categorial Grammar and unification. In: Reyle, U., Rohrer, C. (eds.) Natural Language Parsing and Linguistic Theories. D. Reidel Publ. Co. (1988)
6. Calder, J., Klein, E., Zeevat, H.: Unification Categorial Grammar: A concise, extendable grammar for natural language processing. In: Proceedings of the 12th International Conerence on Computational Linguistics, Budapest (August 1988)
7. Kamp, H., Reyle, U.: From Discourse to Logic. Kluwer Acad. Publ., London (1993)
8. Traat, M., Bos, J.: Unificational Combinatory Categorial Grammar: A formalism for parsing and generating prosodically annotated text. In: Proc. Coling 2004 (2004)
9. Traat, M.: Information Structure in Discourse Representation: Using CCG for Computational Analysis of Theme, Rheme and Focus. PhD thesis, The University of Edinburgh (2006), http://www.era.lib.ed.ac.uk/handle/1842/1260
10. Wood, M.M.: Categorial grammars. Routledge, London (1993)
11. Pierrehumbert, J.: The Phonetics and Phonology of English Intonation. PhD thesis, Massachusetts Institute of Technology, Bloomington, IN (1980)
12. Steedman, M.: Information-structural semantics of English intonation. In: LSA Summer Institute Workshop on Topic and Focus, Santa Barbara, July 2001 (2003)
13. Baldridge, J.: Lexically Specified Derivational Control. PhD thesis, The University of Edinburgh (2002)

A Karaka Based Annotation Scheme for English

Ashwini Vaidya, Samar Husain, Prashanth Mannem, and Dipti Misra Sharma

Language Technologies Research Centre, International Institute of Information Technology, Hyderabad, India
{ashwini_vaidya, samar, prashanth}@research.iiit.ac.in, dipti@iiit.ac.in

Abstract. The paper describes an annotation scheme for English based on Panini's concept of karakas. We describe how the scheme handles certain constructions in English. By extending the karaka scheme for a fixed word order language, we hope to bring out its advantages as a concept that incorporates some 'local semantics'. Our comparison with PTB-II and PropBank brings out its intermediary status between a morpho-syntactic and semantic level. Further work can show how this could benefit tasks like semantic role labeling and automatic conversion of existing English treebanks into this scheme.

1 Introduction

Beginning with the Penn treebank [14], treebank annotation has remained an important research area in CL and NLP. The PTB itself has become richer by incorporating various facets of language phenomenon over the basic phrase structure syntactic representation. Some of these include addition of grammatical relations (PTB-II, [15], [14]), predicate argument structure (PropBank [11]), and immediate discourse structure (PDTB [16]). Treebanks in other languages have continued to enrich this research initiative. For morphologically rich languages like Czech, one major effort has been the Prague Dependency Treebank [8], which has used a dependency based formalism. The Hyderabad dependency treebank- HyDT [1] for Hindi also follows the dependency based approach. In this paper we elaborate & extend the karaka based annotation scheme used in HyDT to English. We also compare some of our tags with similar tags in other well known schemes. As we will see from the examples discussed in the paper, karaka relations capture some level of 'local semantics'. As Rambow et al. [19] state, "local semantic labels are relevant to the verb meaning in question, while global semantic labels are relevant across different verbs and verb meanings". Previous work [18] has used an annotation scheme based on a dependency structure for English but our scheme differs considerably.

The paper is arranged as follows; in Section 2 we discuss the concept of karaka relations. Section 3 describes the data used for annotation. In Section 4 we explain the tagset used. We show how some English constructions are handled in the scheme in Section 5. Section 6 compares our work with a dependency version of Penn Treebank as well as with PropBank. We discuss some related issues in Section 7.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 41–52, 2009.

2 Karaka Relations

The annotation scheme carries out the analysis of each sentence taking into consideration the verb as the central, binding element of the sentence. Sanskrit grammarians like Panini and later Tesnière [22] have used this idea in their grammars. The concept of syntactic valency, where a verb plays the central role has been applied to English before [9].

In this scheme, the verb's requirements for its arguments are the starting point of the analysis. Both arguments and adjuncts are annotated, taking into consideration the verb meaning. Their relationship with the verb is described using relations that we call **karaka** relations. This is a term borrowed from Sanskrit grammar to describe the way in which *arguments participate in the action described by the verb*. We claim that the notion of karaka will incorporate the elements of the local semantics of a verb in a sentence, while also taking cues from the surface level morpho-syntactic information.

For example, *karta* or k1 is a relation that describes an argument that is *most central to the action described by the verb*. The discovery procedure for a karaka like k1 uses such a semantic definition as well as certain morpho-syntactic information. We will discuss the discovery procedure of k1 to give a sense of the type of analysis we have carried out. Some of these tests were created after a pilot annotation of some English sentences.

In a sentence with a finite, transitive verb like ***John gave the flowers to Mary,*** John is a clear candidate for k1, as John is the locus of the activity of the particular verb in the sentence. Moreover, the verb agrees with John and occupies a position to the left of the verb. Both these are also important clues. But, the position of the argument is not always useful. In a sentence like ***To Mary, he gave the flowers; to Susan he gave nothing*** (example from [13], the position of the constituents will not help us. In that case, we will use the other tests of agreement and semantic relationship.

To elaborate further, in the following sentences :

i. The boy opened the lock.
ii. The key opened the lock.
iii. The lock opened. Example from [4]

The boy, the key and the lock will be annotated as k1. In case of ii and iii, the key and the lock are not actually the agents, but in the 'local semantics' of the sentence, i.e. the portion of the action described by the verb, they are the central participants. Hence, these relations differ from the broader 'global' semantic relations of Agent, Patient, Goal etc. To some extent, the notion of k1 corresponds with that of Subject, but there are some important differences. We have discussed these in Section 6. The differences are also apparent in the way subjects for Passives and Expletive sentences are handled (see section 5).

Note that the discovery procedure for k1 in the case of a passive sentence will also take into account prepositional information such as the preposition 'by'. Similarly, we can take into consideration prepositions like 'with' for annotating the relation k3 – instrument essential for the action to take place. (For example, a sentence like *John*

*cut the fruit **with** a knife*). The prepositions are additional clues for the discovery of karaka relations along with the semantic information.

3 Data

The corpus used for annotation consisted of 500 POS tagged sentences from the Wall Street Journal section of the Penn Treebank. The corpus was first converted to the Shakti Standard Format (SSF) [3].

Each sentence was manually chunked and then annotated for dependency relations. While chunking, we assumed that a chunk was a minimal, non-recursive structure consisting of correlated groups of words [2][1].

Karaka relations were marked among chunk heads rather than among each word, as the emphasis was on showing the right modifier-modified relationship. In addition to these, we also annotated verbal nodes with feature structure information. For instance, in order to handle cases with expletive 'it' ('It is raining') we add <stype=expletive__it>[2].

As the task was a preliminary one, a total of two annotators worked on the data. The corpus was small and as the annotators worked on a separate set of sentences, no comment can be made about inter-annotator agreement at this stage.

4 Tagset

We will elaborate on the tagset used in this section. (Fig. 1) shows the hierarchical nature of the tagset. 'Advmod', 'nmod', 'vmod' and 'jjmod' correspond to the adverb modifier, noun modifier, verb modifier and adjective modifier respectively. Below the

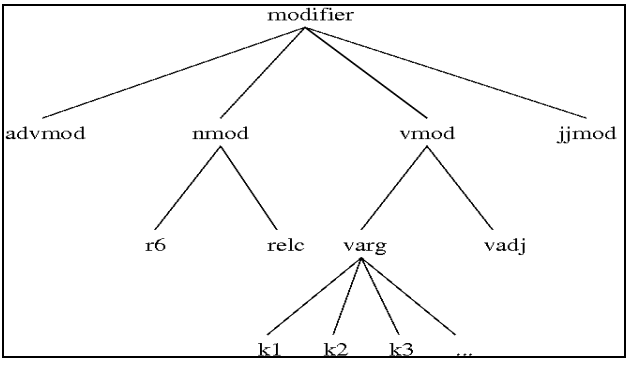


Fig. 1. Hierarchical tagset

[1] This particular chunk definition was used in order to facilitate an English-Hindi machine translation task.

[2] The double underscore '__' is read as 'of the type', and provides a more fine grained classification of the element to its immediate left. So, 'stype__expletive__it' would be read as, *sentence type 'of the type' expletive ' of the type' it.*

noun modifier, we have the noun dependencies of r6 (possession) and relc (relative clause). Similarly, below the verb modifier we have the verb arguments, which are the karaka labels, k1, k2 and so on. This can continue to be expanded into more fine grained labels based on the need. Hence, a relation like k1 can be further divided into different types.

Based on this hierarchy, we list the important noun and verb modifiers (nmod and vmod). (The complete list of tags may be found here[3]). In addition to the relations based on an expansion of the nmod and vmod nodes, we also have the tags 'fragof' and 'ccof'. These do not represent the kind of modification that is vmod or nmod, but show other kind of relations among chunks. For examples, see section 4.3.

The tagset is relatively small (currently 24 tags). Below, the SSF format shows the actual annotation format, with dependency relations and feature structures marked at the chunk level. The SSF representation shows four columns for node index, token (and chunk boundaries), tag and feature structure respectively.

```
<Sentence id="1">
0     ((      SSF
1     ((      VG      <drel=fragof:1>
1.1   Did     VBD
      ))
2     ((      NP      <drel=k1:1>
2.1   Rama    NNP
      ))
3     ((      VG      <name=1/stype=interrogative__yes-no>
3.1   eat     VB
      ))
4     ((      NP      <drel=k2:1>
4.1   the     DT
4.2   banana  NN
4.3   ?       ?
      ))
      ))
</Sentence>
```

The corresponding dependency tree can be seen in Section 4.1, (Fig. 6) The node indexed with 2 for instance shows the chunk boundary of NP followed by the chunk label and the feature structure containing the karaka label (k1) and its head, which is VG (node 3), marked as <name=1>. Using information from the edge label (karaka or others), dependency attachment, feature structure and the word order retained in the format above, the analysis of a sentence is carried out.

4.1 Verb Modifiers

The Sanskrit grammar system described by Panini assigns karakas to verbal arguments based on the relationship they have with the verb. In the annotation effort

[3] http://sites.google.com/site/deptagset/Home?previewAsViewer=1

described here, we have followed the way in which karakas have been defined in Paninian grammar. He classifies six karakas according to the way in which they participate in the action of the verb. These may be listed as follows, with the approximate translations from the sutras that mention them [21]:

k1: *karta*: central to the action of the verb
k2: *karma*: the one most desired by the karta
k3: *karana*: instrument which is essential for the action to take place
k4: *sampradaan*: recipient of the action
k5: *apaadaan*: movement away from a source
k7: *adhikarana*: location of the action

One of the peculiarities of the karaka system according to Panini shows that constructions like active and passive are the realizations of the same structure apart from certain morphological distinctions [12].

We follow the same principle to handle the case of passives in the annotation scheme (Fig. 3). While (Fig. 2) shows the analysis of an active sentence, the same dependency tree is drawn for the passive, only marking the verb's TAM (Tense, Aspect & Modality) as passive. The feature structure that marks the verb morphology as passive will indicate that the agreement and positional information in the tree is applicable to k2 and not k1 (see (Fig. 3), cf. Section 2).

The difference between the two constructions is lexical (and morphological) rather than syntactic in this framework. Such relocation of syntactic information into the lexicon is not unique; frameworks such as MTT [10] also make extensive use of the lexicon to account for various linguistic phenomena.

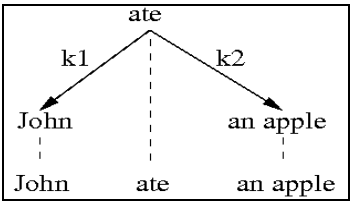


Fig. 2. An active sentence

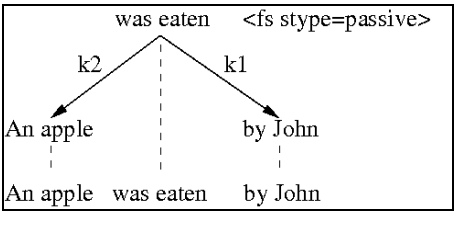


Fig. 3. A passive sentence

4.2 Noun Modifiers

Noun modifiers consist of noun-noun relations such as the possessive relation **r6.** It will hold between two noun chunks. For example, in 'The book of John' the head will be [The book] and [of John] will have a relation of possession (**r6**) with it. Those relative clauses that modify nouns will be marked as **nmod__relc.** (Fig. 4). The figure clearly shows the verb 'joined' as the head of the relative clause and the relative pronoun 'who' which is coreferenced with 'students'.

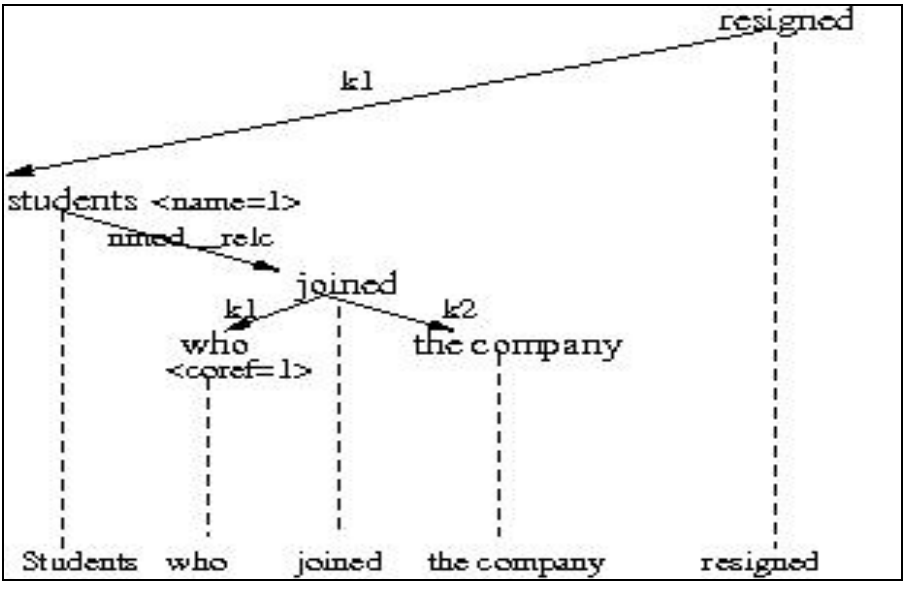


Fig. 4. Relative clause

4.3 Other Labels

In addition to the karaka labels of the hierarchy, we handle co-ordination using the label ccof 'conjunct of'. (Fig. 5) below gives an example for coordination:

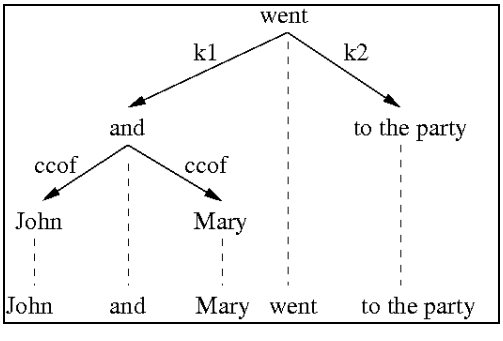


Fig. 5. Co-ordination

Note that in this case, 'and' which is a functional element acts as the head. This label can be used for co-ordination relations among constituents of any kind-noun, verb or adjective chunks. It should also be noted here that unlike the relations discussed earlier, 'ccof' is not a pure dependency label. The problem of representing coordination in the dependency framework is well known, different schemes follow different strategies. In this respect our handling of coordination is close to Prague dependency framework [8].

5 Some Constructions

In this section, we will elaborate on our annotation scheme with the help of some syntactic constructions in English.

5.1 Yes-No Questions

In English, interrogative sentences can be of yes-no type or use a wh-element. In both cases, we treat the displaced element without the use of traces. Instead, the moved constituent is analyzed *in situ*. In the case of yes-no questions, (Fig. 6) shows the dependency tree. We add the information that the sentence is a yes-no type of interrogative sentence. The moved TAM marker is given the label 'fragof'[4] to show that it belongs to the verb chunk that is its head. Finally, we mark the remaining arguments of the verb with karaka relations.

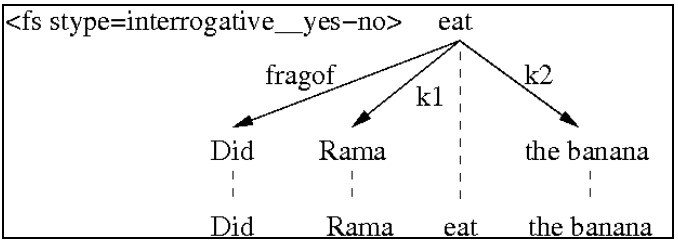


Fig. 6. Yes-no questions

5.2 Control Verbs

For English, the control verbs such as *promise* or *persuade* are not analyzed as cases with an empty PRO. Instead, the analysis shows a difference in the verb semantics of promise and persuade, which again amounts to making the lexicon richer. Traditionally, for an object-control verb like persuade, the object of persuade is coindexed with the missing subject of the subordinate clause.

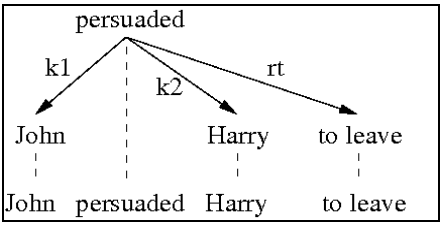


Fig. 7. Object control verb

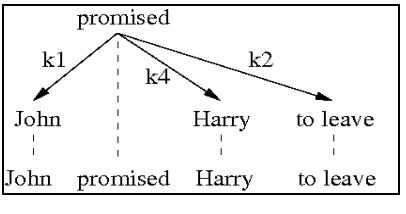


Fig. 8. Subject control verb

[4] *Fragment of.*

In (Fig. 7), the tree does not show a missing element but analyses the verb semantics of persuade differently from the semantics of a verb like promise (a subject-control verb) in (Fig. 8) Persuade is shown to take a karta (k1), a karma (k2) and tadarthya (rt or purpose) labels. Promise on the other hand takes karta(k1), karma(k2) and sampradaan (recipient of the action -k4) labels.

5.3 Expletive Subjects

In (Fig. 9), 'It' is a dummy element in the sentence to fill the empty subject position. We mark it with a special relation 'dummy', which reflects the fact that 'It' is semantically vacuous. We also add the information about the expletive construction to the feature structure of the head. The semantically vacuous 'It' will fail the test for k1 although it is in the subject position and agrees with the verb (Section 2 lists some of the criteria to test for k1).

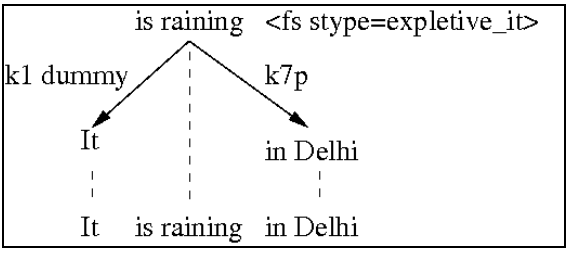


Fig. 9. Expletive sentence

5.4 Subordinate Clauses

Verbs such as *want* that take subordinate clauses can be represented where the subordinate clause is related with the relation k2 'karma'. This analysis is a direct consequence of the semantics of the verb and the way 'k2' is defined (Section 4.1). In (Fig. 10) for example, 'want' takes 'to leave' as its immediate child with a 'k2' relation and 'Harry' is shown attached to 'to leave 'with a relation 'k1'. It is easy to see that (Fig. 10) reflects the predicate argument of 'want' and 'leave'.

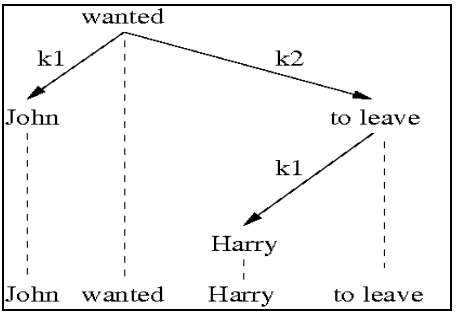


Fig. 10. Subordinate clause

6 Comparison with Other Schemes

As the data used of annotation has been taken from the WSJ[5] section of the Penn Treebank, it is possible to compare our labels with PropBank labels and labels taken from a converted PTB-II treebank. It has been automatically converted into dependency trees [23] and henceforth for convenience, we will refer to it as PennDep. In this section we sketch a preliminary outline about how the karaka labels relate to the syntactic labels in PennDep and the semantic labels of PropBank. We only show mapping between some important labels, an exhaustive mapping analysis is out of the scope of this paper.

Mapping the karaka labels with PennDep labels is straight forward: if the incoming edge of a node in the PennDep tree is marked with some label we find its correspondence in our dependency tree. While mapping NP-SUBJ[6] to k1, we found that 10% of the total NP-SUBJs do not map to k1. This 10% is distributed over **Expletives, Passive constructions, Coordinating conjunctions** and **Non-finite clauses**.

As mentioned earlier, the notion of k1 though syntactically grounded, also entails some semantics. In the case of expletives which are syntactic subjects but are semantically vacuous there is a mismatch (also see, Section 5.3). We see a mismatch for coordinating conjunctions too, the subtree representing conjunction is rooted at the conjunct (see Section 4.3), this is not how conjuncts are represented in PennDep. In PennDep the conjunct becomes the child of the right conjoined element. Unlike PennDep where the object of an active sentence appears as a subject in the passive counterpart, the karaka labels in active-passive sentence do not vary; as mentioned earlier, the scheme looks at a passive construction as a realization of an underlying structure with lexical/morphological variation rather than syntactic. Such asymmetries between k1 and NP-SUBJ mapping point to the fact that the notion of the k1 karaka relation entails a level of semantics which is absent in NP-SUBJ.

Comparison between karaka labels and Propbank labels can be done at two distinct, though related, levels. One is definitional, where we can compare the assumptions which come into play while using terms such as Arg0, Arg1 etc. on the one hand and k1, k2, etc. on the other. The second level would be practical i.e. the comparison of the actual annotation practice and deciding the mapping criterion.

Unlike the mapping between karaka labels and PennDep labels, where simple label comparison of a node sufficed, the mapping in this case would be between a span of text (PropBank annotation) and a subtree (of a dependency tree). Different strategies can be arrived at while deciding the configuration of this subtree. For the purpose of this paper we follow an approach similar to DepPropBank 1 as described in [6] and compare it with the actual PropBank annotation spans. Hence, when mapping, k1 with ARG0, we see if the text span annotated as ARG0 is contained in the subtree rooted at the node which has a k1 relation with the predicate. The map is said to be valid only if all the lexical elements in the dependency subtree appear in the argument span.

The PropBank annotation marks the predicate-argument onto the syntactically parsed, or treebanked, structures. It aims to provide consistent argument labels across

[5] Wall Street Journal.
[6] Noun Phrase Subject.

different syntactic realizations of the same verb. These arguments are labeled as Arg0, Arg1 etc. The annotation also marks modifiers of the verb such as manner (MNR), locative (LOC) etc.

Arg0 arguments are the arguments which cause the action denoted by the verb, either as agents or as experiencers, for instance the arguments of stative verbs such as *love*, *hate*, *fear* etc. Arg1 arguments, on the other hand, are those that change due to external causation, as well as other types of 'patient'-like arguments [5].

It turns out that the karaka labels are very similar to the PropBank labels in this respect. Going by the definition of ARG0, it maps to k1 for most cases. However, k1 also maps to ARG1 in the case of unaccusative verbs. The distribution of k1 across various labels in PropBank is as follows:

(a) k1 → ARG0: ~59%
(b) k1 → ARG1: ~19%
(c) No map: ~20%

We have already observed that k1 will map with ARG1 in the case of unaccusatives. The statistics show ~20% cases where k1 did not map to any PropBank label. Looking at such instances, we found that this occurs mainly due to our chunking guidelines and the mapping strategy used. Some of the chunking decisions are related to **prepositions**, **negation**, **punctuations coordinating conjuncts**, **participle constructions** etc.

According to the chunking guidelines, a preposition appears as part of the chunk, eg. *for* in [For Mr. Winston Smith]; in PropBank only "Mr. Winston Smith" appears as an argument, say ARG0, of a predicate. Mapping the PropBank text span with a subtree in our dependency tree will therefore need a more refined strategy. In fact, [6] mention other mapping versions in their paper. Along with handling the effect of our chunking guidelines, we will have to consider other strategies to reduce this asymmetry. Considering that this problem arises mainly due to the difference in the guideline decisions and can be resolved, we can see that the mapping from karaka labels to PropBank labels can be achieved systematically using a controlled strategy.

Preliminary statistics and observations from the mappings described in this section show that the proposed scheme lies between the syntactic level of the PennDep and the semantic level of the PropBank.

7 Discussion

The comparison effectively brings out some of the properties of the karaka relations that we are annotating on the corpus. Such a comparative study also becomes essential from an interoperability perspective. Mapping various relations in the two schemes helps in bringing out important trends and issues, which prove to be pertinent while automatically converting a treebank from one scheme to other.

We think that the karaka based tagset will give considerable leverage to tasks such as semantic role labeling. We saw in Section 6 that mapping karaka labels with PropBank labels and PennDep labels show consistent pattern.

It has been shown that syntactic parsing helps in identifying semantic relations [7]. Similarly, the karaka based dependency trees proposed in the scheme should help in getting to global semantic relations.

Another characteristic of the scheme is that it has a hierarchical tagset. This means that many relations are left under-specified at the present stage. These relations are typically those which do not concern the argument structure of the verb; for example, noun-noun, participle-verb, and adverb-verb relations. One needs under-specification mainly due to practical concerns. Parsing experiments with very fine-grained labeled treebanks have shown that learning such labels is not always easy [17]. To maximize the overall performance of the parser different levels of granularity can be tried out.

There are a number of issues which still need to be worked out, some obvious phenomena not described in Section 5 are VP ellipsis, wh-movement and raising verb construction like 'seem'. Also, discourse level information needs to be captured using a richer feature structure. In terms of comparison with other annotation schemes for English, we need to carry out experiments with a larger corpus to understand its performance with respect to NLP tasks like semantic role labeling.

References

1. Begum, R., Husain, S., Dhwaj, A., Sharma, D.M., Bai, L., Sangal, R.: Dependency annotation scheme for Indian languages. In: Proceedings of IJCNLP 2008 (2008)
2. Bharati, A., Sangal, R., Sharma, D.M., Bai, L.: AnnCorra: Annotating Corpora Guidelines For POS And Chunk Annotation For Indian Languages. Technical Report, Language Technologies Research Centre IIIT, Hyderabad (2006)
3. Bharati, A., Sangal, R., Sharma, D.M.: Shakti Analyser: SSF Representation (2005), http://shiva.iiit.ac.in/SPSAL2007/ssf-analysis-representation.pdf
4. Bharati, A., Chaitanya, V., Sangal, R.: Natural Language Processing: A Paninian Perspective. Prentice-Hall of India, New Delhi (1995), http://ltrc.iiit.ac.in/downloads/nlpbook/nlp-panini.pdf
5. Babko-Malaya, O.: PropBank Annotation Guidelindes (2005), http://verbs.colorado.edu/~mpalmer/projects/ace/PBguidelines.pdf
6. Ekeklint, S., Nivre, J.: A Dependency-Based Conversion of PropBank. In: Proceedings of FRAME 2007: Building Frame Semantics Resources for Scandinavian and Baltic Languages, pp. 19–25 (2007)
7. Gildea, D., Palmer, M.: The Necessity of Parsing for Predicate Argument Recognition. In: Proceedings of ACL 2002 (2002)
8. Hajicova, E.: Prague Dependency Treebank: From Analytic to Tectogrammatical Annotation. In: Proc. TSD 1998 (1998)
9. Herbst, T.: English Valency Structures - A first sketch. Technical report EESE 2/99 (1999)
10. Kahane, S.: The Meaning-Text Theory. In: Dependency and Valency. An International Handbook on Contemporary Research. De Gruyter, Berlin (2003)
11. Kingsbury, P., Palmer, M.: From Treebank to PropBank. In: Proceedings of the 3rd LREC, Las Palmas, Canary Islands, Spain (2002)
12. Kiparsky., P.: On the Architecture of Panini's grammar. In: Three lectures delivered at the Hyderabad Conference on the Architecture of Grammar (2002), http://www.stanford.edu/~kiparsky/Papers/hyderabad.pdf

13. Kroeger., P.: Analyzing Syntax: A lexical functional approach. Cambridge University Press, Cambridge (2004)
14. Marcus, M., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of English: The Penn Treebank. In: Computational Linguistics (1993)
15. Marcus, M., Kim, G., Marcinkiewicz, M., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., Schasberger, B.: The Penn treebank: Annotating predicate argument structure. In: Proceedings of the ARPA Human Language Technology Workshop (1994)
16. Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., Webber, B.: The Penn Discourse Treebank 2.0. In: Proceedings of the 6th LREC (2008)
17. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., Marsi, E.: MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering 13(2), 95–135 (2007)
18. Rambow, O., Creswell, C., Szekely, R., Taber, H., Walker, M.: A dependency treebank for English. In: Proceedings of the 3rd LREC, Las Palmas, Gran Canaria, Spain (2002)
19. Rambow, O., Dorr, B., Kucerova, I., Palmer, M.: Automatically Deriving Tectogrammatical Labels from other resources- A comparison of Semantic labels across frameworks. The Prague Bulletin of Mathematical Linguistics 79-80, 23–35 (2003)
20. Sgall, P., Hajicova, E., Panevova, J.: The meaning of the sentence and its semantic and pragmatic aspects. Reidel, Dordrecht (1986)
21. Subrahmanyam, P.S.: Pa: ninian Linguistics, Tokyo, Japan: Inst. for the Study of Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies (1999)
22. Tesnière, L.: Eléments de Syntaxe Structurale. Klincksiek, Paris (1959)
23. Yamada, H., Matsumoto, Y.: Statistical dependency analysis with support vector machines. In: Proceedings of IWPT (2003)

Substring Statistics

Kyoji Umemura[1] and Kenneth Church[2]

[1] Toyohashi University of Technology, Tempaku, Toyohashi, Aichi 441-8580, Japan
[2] Microsoft, One Microsoft Way, Redmond, WA 98052, USA

Abstract. The goal of this work is to make it practical to compute corpus-based statistics for all substrings (ngrams). Anything you can do with words, we ought to be able to do with substrings. This paper will show how to compute many statistics of interest for all substrings (ngrams) in a large corpus. The method not only computes standard corpus frequency, $freq$, and document frequency, df, but generalizes naturally to compute, $df_k(str)$, the number of documents that mention the substring str at least k times. df_k can be used to estimate the probability distribution of str across documents, as well as summary statistics of this distribution, e.g., mean, variance (and other moments), entropy and adaptation.

1 Introduction

Substring (ngram) statistics are fundamental to nearly everything we do: language modeling (for speech recognition, OCR and spelling correction), compression, information retrieval, word breaking and more. Many textbooks discuss applications of ngrams including [14] [16] [17] [13] [8] [5] [7] [12]. This paper describes an easy-to-follow procedure for computing many popular statistics for all substrings (ngrams) in a large corpus. **C** code is posted at [21].

[19] showed how to compute standard corpus frequency, $freq$, and document frequency, df, for all substrings in a large corpus. Document frequency is a commonly used statistic, especially in the Information Retrieval community [2] [5]. Document frequency is traditionally defined over words or terms, though we will apply it to substrings.

Definition 1. $df(str) \equiv$ *number of documents that mention str at least once.*

Generalized document frequency, $df_k(str)$, is the number of documents that mention str at least k times. For expository convenience, we use cdf_k for the cumulative document frequency: $cdf_k(str) \equiv \sum_{i=k}^{\infty} df_i(str)$. We can work directly with cdf_k as in [20], or alternatively, cdf_k can be used to compute more standard quantities such as frequency and df_k:

$$freq = cdf_1$$

$$df_k = cdf_k - cdf_{k+1}$$

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 53–71, 2009.

Here, df_1 is the same as standard document frequency (df). df_2 plays an important role in *adaptation*, a term borrowed from the literature on language modeling for speech recognition [13, chapter 14], where there is considerable interest in adapting the probabilities to the first few words of a document. If "Noriega" is mentioned early in the document, chances are that that word (and its friends) will be mentioned again[15]. Psychologists use the term priming [1] to reflect the fact that people react quicker and more accurately to "nurse" if it has been primed by a highly associated word like "doctor."

We define adaptation to be the chance that a term will be mentioned again, given that we have seen it before.

Definition 2. $adaptation \equiv Pr(k \geq 2 | k \geq 1) \approx df_2/df_1$

There are huge quantity discounts, especially for good keywords. The first mention costs $-log(df_1/D)$ bits, but subsequent mentions are cheaper: $-log(df_2/df_1)$ bits. For good keywords like "Noriega," the first mention is quite surprising (e.g., "Noriega" is mentioned in one document in a thousand Associcated Press (AP) stories, shortly after the US invasion of Panama), but the subsequent mention is less so (more than half of the documents that mention "Noriega" once, mention him a second time). There is considerably less adaptation for function words and meaningless random substrings, where the first mention and subsequent mentions are about equally likely (no quantity discounts).

This discounting view is reminiscent of the Given-New Distinction in Discourse Theory [3], which is commonly used in intonation studies such as [4]. The first mention ("new" information) is marked, whereas subsequent ("given") mentions are unmarked. In statistical terms, first mentions tend to be more surprising than subsequent mentions, at least for meaningful words. Random substrings behave more randomly, with less difference between first mentions and subsequent mentions.

In Japanese, we find that words adapt more than most substrings of Japanese characters, and therefore we believe adaptation could be useful in word-breaking applications for Asian languages.

2 Suffix Arrays

A suffix array [9] is a convenient data structure for computing the frequency and location of a substring (ngram) in a large corpus. The input text (corpus) is a sequence of N tokens. Tokens can be words, bytes, Asian characters, etc.

We will use different tokenization rules from time to time. The simplest tokenization rule is to split the text up into bytes, starting a suffix at each byte position. In this case, the number of suffixes, S, is the same as the number of bytes in the input corpus N. For Japanese and Chinese text, it is more appropriate to tokenize by characters (typically 2-bytes), rather than by bytes, so that $S \approx N/2$. For English text, it is often convenient to start suffixes at word boundaries so $S \approx N/5$. The code posted at [21] provides options for different tokenization rules. For expository convenience, we will assume, for the remainder of this discussion, the simplest (and worst case) where $S = N$.

The suffix array of the input is an array of integers, s, whose elements, $s[i]$, denote semi-infinite strings ($suffix[i]$), sorted in lexicographic order. Each semi-infinite string starts at position $s[i]$ in the text and extends to the end of the text. The semi-infinite string, $suffix[i]$, is represented in **C** with a single integer, consuming just $O(1)$ space, but it denotes a long substring, $substr(text, s[i])$, which would require $O(N)$ bytes (if we materialized it). Another way to express the materialized semi-infinite string is the **C** expression, $text + s[i]$, where $text$ is a string (char *) and i is an integer (int).

Definition 3. $suffix[i] \equiv substr(text, s[i]) = text + s[i]$

A simple procedure for constructing the suffix array, s, is shown in Fig. 1.

```
s = (int *)malloc(N * sizeof(int));
for(i=0; i<N; i++) s[i] = i; /* initialize */
qsort(s, N, sizeof(*s), sufcmp); /* sort lexicographically */

int sufcmp(int *a, int *b) { return strcmp(text + *a, text + *b); }
```

Fig. 1. A simple procedure creating a suffix array from an input corpus (*text*). The suffix array, $s[i]$, is initialized to the integers from 0 to $N-1$. Each integer denotes a suffix or semi-infinite string, starting at position $s[i]$ and extending to the end of the corpus. The integers in s are then sorted so the semi-infinite strings will be in alphabetical order.

Fig. 2 shows the suffix array before and after sorting.

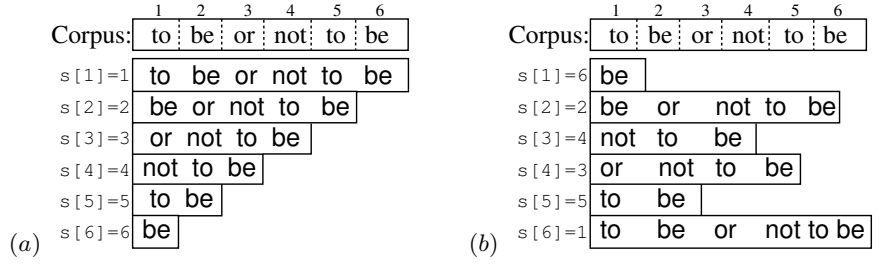


Fig. 2. (a) Illustration of a suffix array, s, after initialization but before sorting. Each element in the suffix array, $s[i]$, is an integer denoting a suffix or semi-infinite string, starting at position $s[i]$ in the corpus and extending to the end of the corpus. (b) The suffix array, s, after sorting. The integers in s are sorted so that the semi-infinite strings are now in alphabetical order.

See http://www.cs.dartmouth.edu/∼doug for an excellent tutorial with **C** code of a more sophisticated algorithm with better theoretical bounds [9]. If appropriate care is taken to remove duplicate documents, it has been our experience that the complications of the more sophisticated algorithm are often not needed in practice, and can actually degrade performance (slightly).

```
sufconc -l 10 -r 40 AP/AP8912 'Manuel Noriega' | sed 3q
17913368     5441: osed Gen.  ^ Manuel Noriega\nin Panama _ their wives
13789741     4193: apprehend  ^ Manuel Noriega\n   The situation in Pana
 3966027     1218: nian Gen.  ^ Manuel Noriega a\n$300,000 present, and
```

Fig. 3. A concordance computed from a suffix array. The `sufconc` program prints out a concordance line for each suffix in the interval $\langle i, j \rangle$. Newlines are translated to "\n." Similar conversions are performed for other characters that would cause trouble. A carrot ($\wedge$) is inserted just before the input pattern. The command line arguments specify how much context to print to the left and right of the carrot. The first two numbers of each line are the suffix, $s[i]$, and the associated document id, $doc(s[i])$.

2.1 LCP (Longest Common Prefix)

In addition to s, the suffix array literature also makes use of Longest Common Prefixes (LCP), as illustrated in Fig. 4. $LCP[i]$ contains the length of the common prefix between $s[i]$ and $s[i+1]$. We will refer to "prefixes of suffixes" as "substrings." LCP is a vector of N ints, like s.

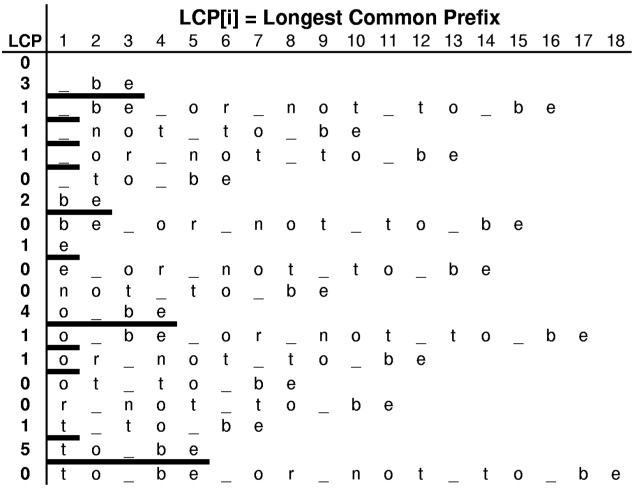


Fig. 4. LCP (Longest Common Prefix). The bars highlight the length of the common prefix shared between the pair of substrings just above and just below the bar.

The suffix array literature shows how to compute the LCPs efficiently, even in the worst case. The code posted at [21] provides a much more straightforward implementation, which is simpler to understand, and is often faster in practice, but can take $O(N^2)$ time in the worst case.

2.2 `sufconc`

As mentioned above, suffix arrays make it convenient to find the frequency and location of any substring in a large corpus. The example below, for example,

shows a couple of concordance lines for the substring "Manual Noriega" from a month of Associated Press news (December, 1989), as distributed by TIPSTER and the LDC [10]. The sufconc program takes an input substring pattern and performs two binary searches to find an interval on the suffix array: $\langle i, j \rangle$. $s[i]$ is the first suffix in the suffix array that starts with the input pattern ("Manuel Noriega"); s[j] is the last suffix that starts with "Manuel Noriega." Standard corpus frequency can be computed straightfowardly from the width of the interval $\langle i, j \rangle$. That is, $freq = j - i + 1$.

The `sufconc` program, as illustrated in Fig. 3, prints out a concordance line for each suffix in the interval $\langle i, j \rangle$.

The subroutine that materializes suffixes and prints them out is called *pname*, by analogy to the LISP function that converts a symbol pointer to a string. We view the elements of a suffix array as analogous to symbols in a symbol table. As with LISP, most of the processing can be done in terms of the pointers, and except for printing routings, there shouldn't be much need to dereference the pointers and materialize the strings. The *pname* function converts newlines to "\n." Similar conversions are performed for other characters that would cause trouble. A carrot ("^") is inserted just before the input pattern. The command line arguments specify how much context to print to the left and right of the carrot. The first two numbers of each line are $s[i]$ and the associated document id, $doc(s[i])$.

3 Classes

3.1 Distributional Equivalence

We have seen thus far how to compute the frequency and location for a particular substring. This section will show how to compute these statistics for all substrings, not just for a particular substring.

Although there are too many substrings ($N(N+1)/2$) to work with directly, they can be grouped into a manageable number of N equivalence classes [19]. The construction starts with an interval $\langle i, j \rangle$ on the suffix array, where $i \leq j$. From this interval, we construct an equivalence class, the (possibly empty) set of substrings that start every suffix within the interval, and no other suffixes. We say that the interval is *valid* iff the class is non-empty.

Classes form an equivalence relation, *distributional equivalence*, on substrings. For example, in the "to be or not to be" example, the substrings "to" and "to be," which are in the same class, have identical distributions; they both start exactly the same set of suffixes. Distributional equivalence is reflexive, symmetric and transitive. Classes partition the set of all substrings; every substring is a member of one and only one class.

Distributionally equivalent substrings have the same statistics, at least for many popular statistics including standard corpus frequency, document frequency, joint document frequency, and combinations of these quantities including moments, entropy, adaptation, etc. In particular, all the substrings in $Class(\langle i, j \rangle)$ have the same frequency $(j - i + 1)$. The set of substrings

in a class can be computed from the LCP vector, as illustrated in Fig. 5. $Class(\langle i,j\rangle) = \{$ substrings $str \mid str$ starts every suffix in $\langle i,j\rangle$ and no others $\}$ $= \{substr(text + s[i], 1, k)\}$ for $LBL < k \leq SIL$.

The LBL (Longest Bounding LCP) $= max(LCP[i-1], LCP[j])$. The SIL (Shortest Interior LCP) $= MIN_{i \leq k < j} LCP[k]$. In fact, we can replace $s[i]$ with $s[w]$ for any witness $i \leq w \leq j$ since all of the suffixes in the interval are the

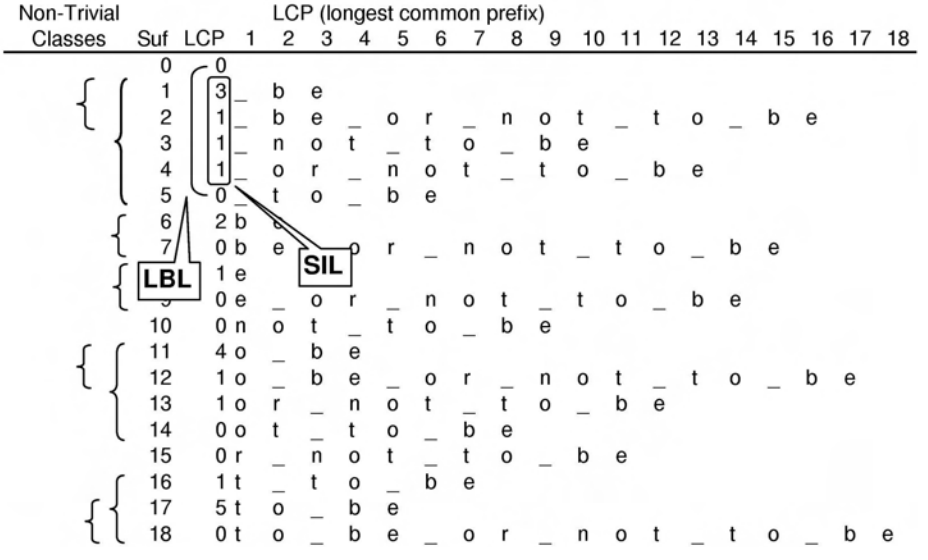


Fig. 5. $Class(\langle i,j\rangle) = \{$ substrings $str \mid str$ starts every suffix in $\langle i,j\rangle$ and no others $\}$ $= \{$ $substr(text + s[w], 1, k)$ $\}$ for $LBL < k \leq SIL$. The LBL (Longest Bounding LCP) $= max(LCP[i-1], LCP[j])$. The SIL (Shortest Interior LCP) $= MIN_{i \leq k < j} LCP[k]$. The interval $\langle 1, 5\rangle$, for example, has an LBL of 0 and a SIL of 1. Thus, the class contains just one substring, $substr(text + s[1], 1, 1) =$ "_".

Table 1. Although there are too many substrings to work with (N^2), they can be grouped into a manageable number of interesting (non-empty and non-trivial) classes

Objects	Description	Quantity
Text	the Corpus	N tokens
Suffix Array	s[i] = substr(text, i)	N suffixes or less
Substring	substr(text, i, j)	$N(N+1)/2$ or less
Interval on Suffix Array	$< i,j >= \{s[k] \mid i \leq k \leq j\}$	$N(N+1)/2$ or less
$Class(< i,j >)$	$\{$ substrings $str \mid$ str starts every suffix in $< i,j >$, and no others $\}$	$N(N+1)/2$ or less
Valid (Non-Empty) Class	$Class(< i,j >) \neq \emptyset$	$2N$ or less
Trivial Class	$< i,i > \Rightarrow freq(< i,i >) = 1$	N or less
Interesting Class (Non-Trivial and Non-Empty)	$Class(< i,j >) \neq \emptyset \wedge i \neq j$	N or less

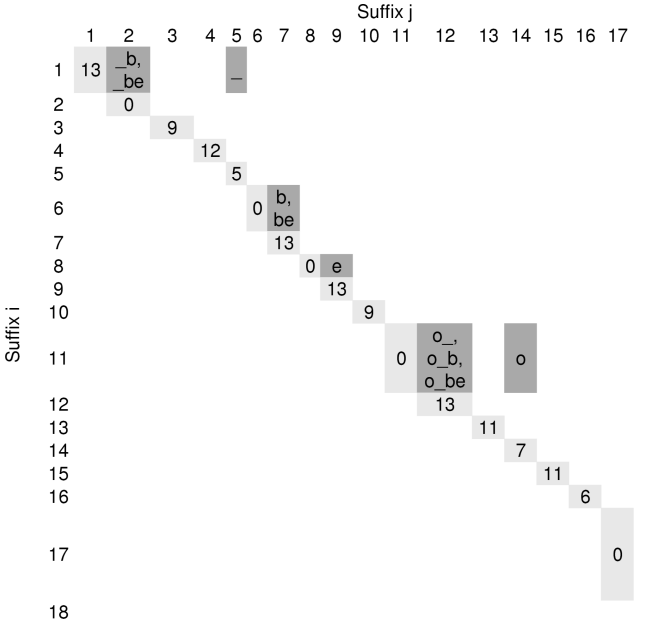


Fig. 6. The 171 substrings fall into just 8 interesting classes. Most of the classes are empty (invalid). Most of the rest are trivial (freq = 1). The trivial classes fall along the main diagonal. There are just 8 interesting classes (non-empty and non-trivial). 15 substrings appear in the 8 interesting classes; 135 substrings appear in trivial classes along the main diagonal.

same up to the SIL. We will refer to the longest member of $Class(\langle i, j\rangle) = substr(text + s[i], 1, SIL)$ as the canonical member of the equivalence class.

Table 1 shows that there are lots of substrings and lots of intervals (order N^2), but relatively few interesting classes (at most N). We say that an interval is invalid if the class is empty, and we say that the class is trivial if the frequency is 1. Most of the N^2 intervals are invalid, and most of the valid intervals are trivial. There are relatively few remaining classes (non-trivial and non-empty).

Fig. 6 illustrates graphically the massive reduction from N^2 down to N. The string "to be or not to be$" has N=19 characters, and $N * (N-1)/2 = 171$ possible substrings. For each substring, there is a possible interval $\langle i, j\rangle$ on the suffix array, but only 8 of the 171 possible intervals are interesting (non-empty and non-trivial).

The main motivation for grouping substrings into classes is the computational consideration: N is more manageable than N^2. Most statistics of interest can be computed over classes rather than substrings because distributionally equivalent substrings have the same statistics, at least for many popular statistics.

3.2 Enumerating Classes

Fig. 7 shows that interesting (non-empty and non-trivial) intervals form a tree. The procedure in Fig. 8 performs a depth-first traversal of this tree. The classes,

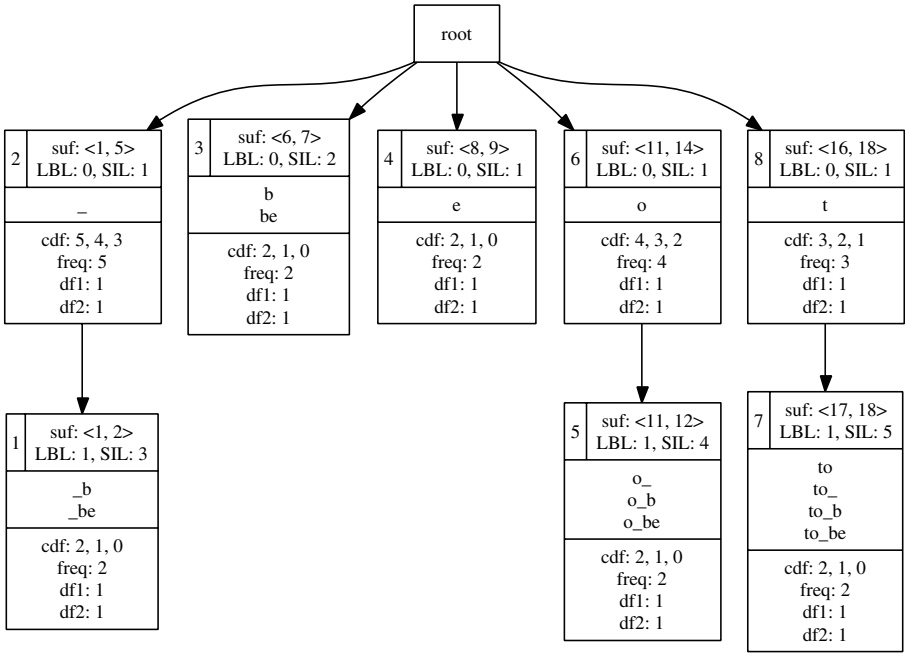


Fig. 7. Class Tree for "to be or not to be$." Interesting (non-empty and non-trivial) intervals form a tree. The procedure in Fig. 8 performs a depth-first traversal of this tree. The numbers in the upper left hand corner of each node indicate the class id, the position of the class in the depth-first traversal. The class id is followed by the interval $\langle i, j \rangle$, LBL and SIL. The middle of each node lists the substrings in $Class(\langle i, j \rangle)$. At the bottom of each node are various statistics that will be discussed later.

$\langle i, j \rangle$, are enumerated in sorted order, sorted first by j in increasing order and then by i in decreasing order.

The fact that the output is sorted is convenient for retrieval purposes. Table 2 shows the output from find_class. This program takes a string such as "Norieg" as input and retrieves the $Class(\langle i, j \rangle)$ as well as the LBL, SIL, a number of pre-computed statistics and the set of distributionally equivalent substrings. The program performs three binary searches. There are two binary searches into the suffix array to find $\langle i, j \rangle$, the first and last suffix starting with the input pattern. The third binary search is performed on the classes, as enumerated by the method described above. Some of the values in Table 2 could have been computed without classes (i, j, LBL, freq), but others benefit considerably from classes (SIL, df_k, distributionally equivalent substrings).

Table 3 shows snapshots of the stack as each of the 8 classes are output using the depth-first traversal introduced in Fig. 8. The stack pointer corresponds (roughly) to the depth of the class tree shown in Fig. 7, though not exactly because of head recursion. Although the class tree reaches a depth of 3 in 3 places in Fig. 7 (class ids 1, 5 and 7), the stack pointer reaches 3 in just one

Table 2. Find class: inputs a pattern (a substring such as "Norieg") and outputs $\langle i, j \rangle$, LBL, SIL, a number of pre-computed statistics and the set of distributionally equivalent substrings, $Class(\langle i, j \rangle)$. (The table below shows just the longest and shortest member of $Class(\langle i, j \rangle)$; the others have be replaced with "...")

Source	i	j	LBL	SIL	freq	df_1	df_2	$Class(< i, j >)$
AP8901	6693634	6693741	4	7	108	22	14	{ Norie,...,Noriega }
AP8902	5927492	5927574	4	7	83	16	12	{ Norie,..., Noriega }
AP8903	6373730	6373804	3	7	75	14	11	{ Nori,...,Noriega }
AP8904	6121594	6121819	4	7	226	34	30	{ Norie,...,Noriega }
AP8905	6804959	6806255	4	7	1297	188	158	{ Norie,...,Noriega }
AP8906	6470367	6470572	3	7	206	45	31	{ Nori,...,Noriega }
AP8907	6389275	6389438	4	7	164	35	26	{ Norie,...,Noriega }
AP8908	6150308	6150676	4	7	369	63	52	{ Norie,...,Noriega }
AP8909	6353292	6353519	4	7	228	38	28	{ Norie,...,Noriega }
AP8910	6818197	6819484	4	6	1288	180	149	{ Norie,...,Norieg }
AP8911	6320624	6320745	4	7	122	29	20	{ Norie,...,Noriega }
AP8912	5968758	5971461	4	6	2704	403	327	{ Norie,...,Norieg }

Table 3. Stack Trace. Snapshots of the stack are shown as each of the 8 classes are output using the depth-first traversal introduced in Fig. 8. The stack pointer corresponds (roughly) to the depth of the class tree shown in Fig. 7. The stack frames for the root node are omitted because they are not very interesting.

Stack Pointer	i	SIL	Canonical member of class($\langle i, j \rangle$)	Output class id	Output interval	Output SIL
2	1	3	_be	1	$\langle 1, 2 \rangle$	3
2	1	1	_	2	$\langle 1, 5 \rangle$	1
2	6	2	be	3	$\langle 6, 7 \rangle$	2
2	8	1	e	4	$\langle 8, 9 \rangle$	1
2	11	4	o_be	5	$\langle 11, 12 \rangle$	4
2	11	1	o	6	$\langle 11, 14 \rangle$	1
2	16	1	t			
3	17	5	to_be	7	$\langle 17, 18 \rangle$	5
2	16	1	t	8	$\langle 16, 18 \rangle$	1

place (class id 7). Head recursion avoids pushing the stack in the other two cases (class ids 1 and 5).

Head recursion is the dual of tail recursion [6]. Tail recursion replaces right branching recursion structures with iteration; head recursion does the same but for left branching structures. Class ids 1 and 5 are examples of left branching structures where the interval for the mother ($\langle i, j \rangle$) and the interval for the daughter ($\langle i, l \rangle$) share the same starting point (i). In the case of class id 1, both the mother ($\langle 1, 5 \rangle$) and the daughter ($\langle 1, 2 \rangle$) share the same starting point (1). Similarly, in the case of class id 5, both the mother ($\langle 11, 14 \rangle$) and the daughter ($\langle 11, 12 \rangle$) share the same starting point (11). Class id 7, on the other hand, is

```
struct stackframe { int i, j, SIL } *stack;
int sp = 0; /* stack pointer */
stack[sp].i = 0; stack[sp].SIL  = -1;

for(w=0; w<N; w++) {
    if(LCP[w] > stack[sp].SIL) {
        sp++; /* push */
        stack[sp].i = w;
        stack[sp].SIL = LCP[w]; }
    while(LCP[i] < stack[sp].SIL) {
       stack[sp].j = w;
       output(&stack[sp]);
       if(LCP[w] <= stack[sp-1].SIL)  sp--; /* pop */
       else stack[sp].SIL = LCP[w]; }} /* head recursion */
```

Fig. 8. A depth-first enumeration of classes

an example of a right branching structure since the mother ($\langle 16, 18\rangle$) and the daughter ($\langle 17, 18\rangle$) share the same endpoint (18).

The last line of Fig. 8 implements the head recursion for left branching structures. After outputting class id 1, the last line of Fig. 8 recycles the daughter's stackframe (class id 1) for its mother (class id 2). The same pattern applies after class id 5, where the daughter's stackframe is also recycled for its mother (class id 6).

With suffix arrays, we could compute the frequency and location for a particular substring (ngram). Classes make it feasible to do that and more (generalized document frequency) for all substrings.

4 Document Frequency

Generalized document frequency, $df_k(str)$, is the number of documents that contain str at least k times. The method will compute cumulative document frequency, $cdf_k = \sum_{i=k}^{\infty} df_i(str)$. As mentioned above, frequency and df_k can be recovered from cdf_k:

$$freq = cdf_1$$

$$df_k = cdf_k - cdf_{k+1}$$

A simple (but slow) method for computing cdf_k from an interval $\langle i, j\rangle$ is

$$cdf_1(\langle i, j\rangle) = \sum_{i \le w \le j} 1 = j - i + 1$$

$$cdf_2(\langle i, j\rangle) = \sum_{i \le w \le j} \begin{cases} 1 & if\ neighbor[w] \ge i \\ 0 & otherwise \end{cases}$$

$$cdf_k(\langle i, j\rangle) = \sum_{i \le w \le j} \begin{cases} 1 & if\ neighbor^{k-1}[w] \ge i \\ 0 & otherwise \end{cases}$$

where $Neighbors^k[s] \equiv Neighbors^{k-1}[Neighbors[s]]$, for $k \neq 1$. (The first two formulas above can be viewed as special cases of the third, where $neighbor^0$ is the identity function.)

Neighbors is a function from suffixes to suffixes. Neighbors$[s_2] = s_1$ if s_1 and s_2 are in the same document and s_1 and s_2 are adjacent. By adjacent, we mean there is no suffix s_3 in the same document that is between s_1 and s_2. Table 5 shows the neighbors for the sample corpus in Table 4.

Fig. 10 shows a simple (but slow) method to compute cdf_k, using a straightforward implementation of the discussion above. This method is slow because the class tree structure may become very deep. Fig. 11 is the same as Fig. 10 except that the top level loop has been folded into the depth-first traversal of the class tree.

We recommend the faster (nearly linear time) improvement in Fig. 12. The speed-up is achieved by propagating counts up the class tree. The mother receives

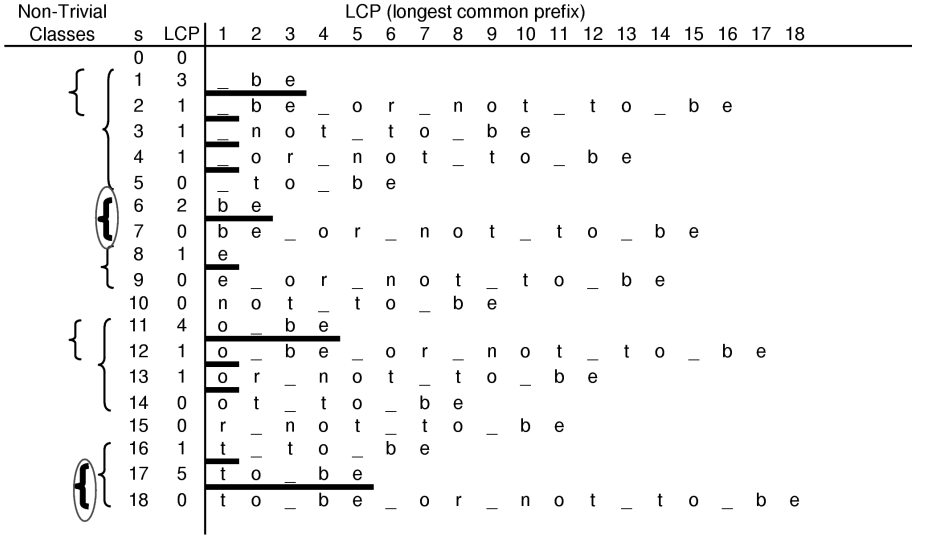


Fig. 9. The $N(N+1)/2$ substrings can be grouped into N or fewer equivalence classes. A class is defined in terms of intervals $\langle i, j\rangle$ on the suffix array. The class contains the set of substrings that start every suffix within the interval and no suffixes outside the interval. The circles highlight two examples. $Class(\langle 6, 7\rangle) =$ {b, be}. $Class(\langle 17, 18\rangle) =$ {to, to_, to_b, to_be}.

Table 4. A sample corpus with three documents

doc	body
0	Hi_Ho_Hi_Ho
1	Hi_Ho
2	Hi

Table 5. Neighbors Neighbors[s_2] = s_1 if s_1 and s_2 are in the same document and s_1 and s_2 are adjacent. By adjacent, we mean there is no suffix s_3 in the same document that is between s_1 and s_2. For example, for the 21-byte collection in Fig. 4, there are 21 suffixes as shown in Table 6. Each of the three documents in the collection are associated with a set of suffixes. The neighbors can be computed by "shifting" these sets. Thus, the first suffix in each document has no neighbor (neighbor[2] = neighbor[1] = neighbor[0] = "NA"). Subsequent suffixes are mapped to the previous suffix in the same document, as illustrated below: neighbors[4] = 2, neighbors[6] = 4, etc.

doc	s
0	2, 4, 6, 8, 9, 10, 11, 13, 15, 17, 19, 20
1	1, 5, 7, 12, 16, 18
2	0, 3, 14

doc 0		doc 1		doc 2	
s	neighbor	s	neighbor	s	neighbor
2	NA	1	NA	0	NA
4	2	5	1	3	0
6	4	7	5	14	3
8	6	12	7		
9	8	16	12		
10	9	18	16		
11	10				
13	11				
15	13				
17	15				
19	17				
20	19				

```
struct stackframe { int start, SIL, cdfk } *stack;

/* returns neighbor^k(suf) or -1 if NA */
int kth_neighbor(int suf, int k)
{ if(suf >= 0 || k > 1)
    return(kth_neighbor(neighbors[suf], k-1));
  else return suf; }

struct class c;
while(fread(&c, sizeof(c), 1, stdin)) {
  int cdfk = 0;
  for(w=c.start; w<=c.end; w++)
    if(kth_neighbor(w, K-1) >= c.start) cdfk++;
  putw(cdfk, out); } /* report */
```

Fig. 10. Simple (but slow) code for cdf_k, using a straightforward implementation of $cdf_k(< i, j >) = \sum_{i \le w \le j} \begin{cases} 1 & if\ neighbor^{k-1}[w] \ge i \\ 0 & otherwise \end{cases}$

```
for(w=0; w<N; w++) {
   if(LCP[w]> stack[sp].SIL) {
      sp++;
      stack[sp].start = w;
      stack[sp].SIL = LCP[w];
      stack[sp].cdfk = 0; }

   for(sp1=0; sp1<=sp; sp1++) {
      if(kth_neighbor(w, K-1) >= stack[sp1].start)
        stack[sp1].cdfk++; }

    while(LCP[w] < stack[sp].SIL) {
      putw(stack[sp].cdfk, out); /* report */
      if(LCP[w] <= stack[sp-1].SIL) sp--;
      else stack[sp].SIL = LCP[w]; }}
```

Fig. 11. Same as Fig. 10, but folded into the depth-first traversal of the class tree

```
/* return first stack frame not before suffix */
/* binary search works because stack is sorted */
int find(int suffix)     /* log(max(LCP)) time */
{ int low = 0;
  int high = sp;
  while(low + 1 < high) {
    int mid = (low + high) / 2;
    if(stack[mid].start <= suffix) low = mid;
    else high = mid;  }
  if(stack[high].start <= suffix) return high;
  if(stack[low].start <= suffix)  return low;
  fatal("can't get here"); }

for(w=0; w<N; w++) {  /* N time */
  if(LCP[w]> stack[sp].SIL) {
    sp++;
    stack[sp].start = w;
    stack[sp].SIL = LCP[w];
    stack[sp].cdfk = 0; }

    int prev = kth_neighbor(w, K-1);
    if(prev >= 0) stack[find(prev)].cdfk++;

    while(LCP[w] < stack[sp].SIL) {
      putw(stack[sp].cdfk, out); /* report */
      if(LCP[w] <= stack[sp-1].SIL) {
         /* propagate counts up class tree */
         stack[sp-1].cdfk += stack[sp].cdfk;
         sp--; }
      else stack[sp].SIL = LCP[w]; }}
```

Fig. 12. We recommend this this $O(N\ max(k,\ log\ max(LCP)))$ procedure for cdf_k

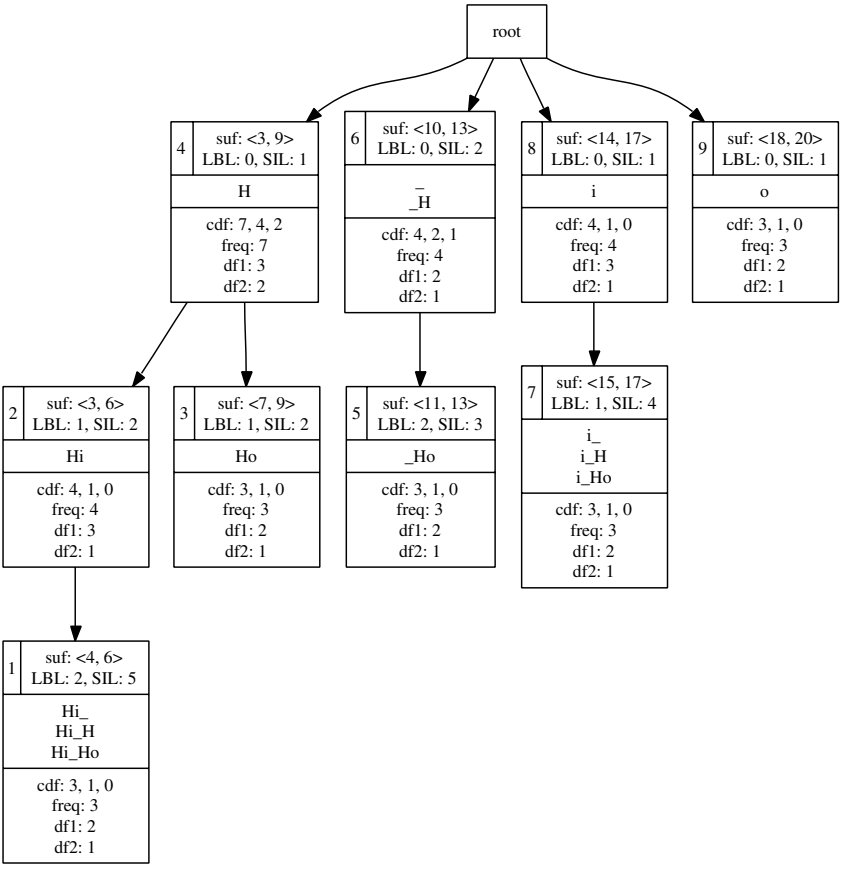


Fig. 13. Results (class tree and cdf_k) for input in Table 4. The numbers in the upper left hand corner of each node indicate the class id, the position of the class in the depth-first traversal of the class tree. The class id is followed by the interval $< i, j >$, LBL and SIL. The middle of each node lists the substrings in $Class(< i, j >)$. At the bottom of each node, cdf_1, cdf_2 and cdf_3 are presented, followed by $freq$, df_1 and df_2, where $freq = cdf_1$ $df_1 = cdf_1 - cdf_2$ $df_2 = cdf_2 - cdf_3$

all of the counts from her daughters plus whatever counts she receives on her own ($cdf_k[mother] \geq \sum_{d \in daughters} cdf_k[d]$). The results are shown in Fig. 13.

See [21] for **C** code that inputs a text file and outputs a number of indexes including the suffix array, LCP, classes and cdf_1, cdf_2 and cdf_3, using the recommendation in Fig. 12.

5 Some Practical Experience with AP Newswire

The code in [21] has been applied to 12 months of the 1989 AP news as distributed by TIPSTER and the LDC [10] to compute the suffix array, LCP,

Table 6. Suffix array for Table 4. p is an offset into the corpus; s is an offset into the suffix array.

doc	p	s	suffix
2	20	0	“”
1	17	1	“”
0	11	2	“”
2	18	3	“Hi”
0	6	4	“Hi_Ho”
1	12	5	“Hi_Ho”
0	0	6	“Hi_Ho_Hi_Ho”
1	15	7	“Ho”
0	9	8	“Ho”
0	3	9	“Ho_Hi_Ho”
0	5	10	“_Hi_Ho”
0	8	11	“_Ho”
1	14	12	“_Ho”
0	2	13	“_Ho_Hi_Ho”
2	19	14	“i”
0	7	15	“i_Ho”
1	13	16	“i_Ho”
0	1	17	“i_Ho_Hi_Ho”
1	16	18	“o”
0	10	19	“o”
0	4	20	“o_Hi_Ho”

classes and cdf_1, cdf_2 and cdf_3. Space and time are dominated by the suffix array computation. While there are a prohibitive number of possible substrings ($N(N-1)/2$), Table 7 shows that there aren't that many interesting (non-trivial non-empty) classes (C). In practice, C is quite a bit smaller than N ($C \approx N/2$), which is quite a bit better than the bound in Table 1, $C \leq N$, based on theoretical considerations.

For practical applications, the most serious concern is physical memory. If we wanted to scale up from a month of newswire (4M words) to the web (20B pages), we would need to generalize the methods to work in external memory, distributed across a cluster of machines. It is not too difficult to convert most of the **C** code to work in external memory, though we are not aware of a publicly available external memory implementation of suffix arrays.

The recommended speed ups make it practical to compare df_1 and df_2 at scale, as illustrated in Fig. 14. Both df_1 and df_2 have a Zipf-like distribution, but the df_1 curve is well above df_2 (especially at low frequencies). It is interesting that the two lines are not parallel. These two lines determine adaptation, $Pr(k \geq 2|k \geq 1) \approx df_2/df_1$.

The code is [21] also makes it easy to compute LCPs at scale, though that had been possible before the recommended speed ups. Fig. 15 shows LCPs for three months of AP news. The three distributions are nearly identical to one another

Table 7. Instead of computing statistics of interest over the $N(N-1)/2$ substrings, we compute them over C classes. In practice, $C \approx N/2$, which is quite a bit better than the bound in Table 1, $C \leq N$, based on theoretical considerations.

Month	N = Text Size (millions of bytes)	C = Number of Classes (millions)
Jan	23	13
Feb	21	11
March	22	12
April	21	12
May	24	13
June	23	12
July	22	12
Aug	21	12
Sept	22	12
Oct	24	13
Nov	22	12
Dec	21	11

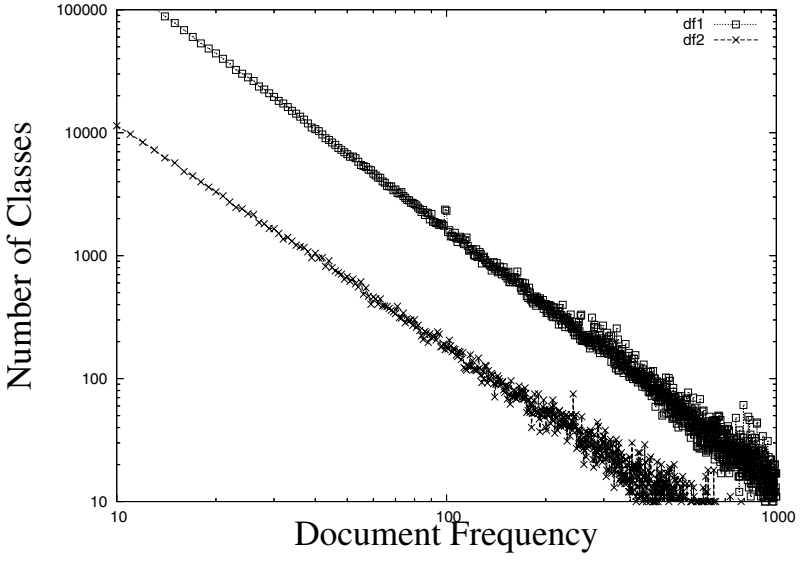


Fig. 14. df_1 and df_2 have a Zipf-like distribution (in January 1989 AP newswire). The df_1 curve is well above df_2 (especially at low frequencies).

(at least for reasonable LCPs up to tens of bytes). The very long LCPs are associated with artifacts in the AP data such as duplicated stories and boilerplate.

Artifacts raise serious issues for language modeling. Consider the difference between "in Monday" and "on Monday." The former sounds weird, but it is common in headers:

```
<HEAD>Eds: Also in Monday AMs report.</HEAD>.
```

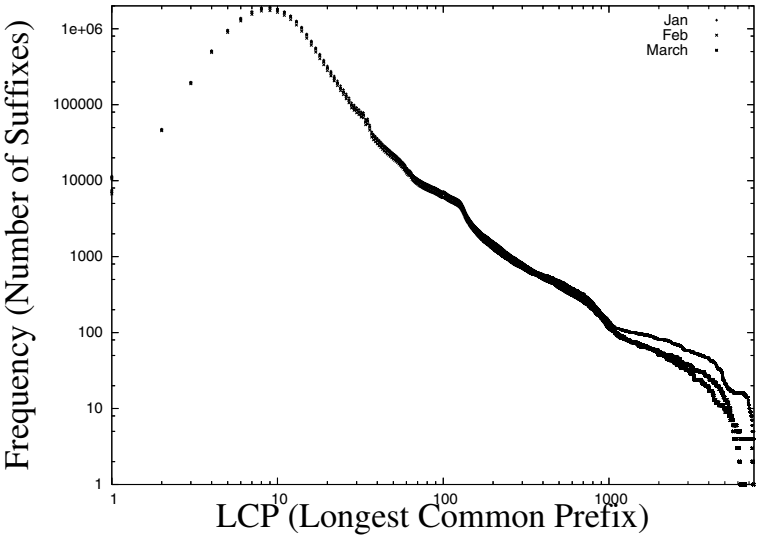


Fig. 15. The distribution of LCPs in three months of 1989 AP newswire. The distributions are remarkably similar from one month to the next. This peak is at 9 bytes, which corresponds roughly to bigrams. There are, of course, many common prefixes that extend to hundreds or even thousands of bytes. The very long LCPs tend to be associated with artifacts in the AP news such as duplicated documents.

The ability to compute df_1 and df_2 at scale can help identify artifacts like this. Both "in Monday" and "on Monday" are reasonably frequent in this corpus (57 and 394 documents, respectively). The difference is more saliient in terms of adaptation. "On Monday" is repeated fairly often (10% of the 394 documents that it appears in), in constrast to "in Monday" (0 of 57).

Duplicate documents are quite common. Many of these duplicate documents have long repeated substrings, as illustrated in Fig. 15. The code in [21] makes it easy to compute LCPs for all substrings, though that had been possible using previous methods such as [19].

6 Conclusion

Substring (ngram) statistics are fundamental to nearly everything we do. The goal of this work is to make it practical to compute corpus-based statistics for all substrings (ngrams). Anything you can do with words (and short ngrams), we ought to be able to do with million-grams. Previous work [19] showed how to compute standard frequency ($freq$) and document frequency (df) for all substrings. This paper has simplified that procedure, and generalized the result. We showed how to compute cumulative document frequency, cdf_k, which encodes $freq$ and df, as well as generalized document frequency (df_k):

$$freq = cdf_1$$

$$df_k = cdf_k - cdf_{k+1}$$

These values determine the probability distribution of substrings in documents ($Pr(k) \approx df_k - df_{k+1}$), as well as summary statistics of $Pr(k)$ such as moments, entropy, adaptation, etc. The leading terms of cdf_k are usually the largest and most important.

The computation of cdf_k started with suffix arrays. Suffix arrays make it easy to determine the frequency and location of a particular substring. To compute those statistics and more for all substrings, we grouped the N^2 substrings into N equivalance classes.

Classes are defined in terms of intervals on the suffix array: $\langle i, j \rangle$. The set of substrings in a class is: $Class(\langle i, j \rangle) = \{$ substrings $str | str$ starts every suffix in $< i, j >$ and no others $\} = substr(text, s[i], k)$ where $LBL < k \leq SIL$. Equivalent substrings, e.g., "to" and "to be" in the "to be or not to be" example, have identical distributions; wherever "to" appears in the corpus, "to be" is sure to follow.

Generalized document frequency can be computed directly from the classes

$$cdf_k(< i, j >) = \sum_{i \leq w \leq j} neighbor^{k-1}[w] \geq i$$

but it is more efficient to fold this calculation into a depth first traversal of the interval tree, propagating the counts from each daughter to its mother. The traversal outputs classes sorted first by i (in increasing order) and then by j (in decreasing order). The total order is convenient, making it possible to find classes quickly with a binary search.

References

1. Meyer, D., Schvaneveldt, R.: Facilitation in recognizing pairs of words: Evidence of a dependence between retrieval operations. Journal of Experimental Psychology 90, 227–234 (1971)
2. Jones, K.S.: A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation 28(1), 11–21 (1972)
3. Prince, E.: Toward a taxonomy of given-new information. In: Cole, P. (ed.), pp. 236–256. Academic Press, New York (1981)
4. Davis, J.R., Hirschberg, J.: Meeting of the Association for Computational Linguistics, 187–193 (1988)
5. Salton, G.: Automatic text processing. Addison-Wesley Longman Publishing Co., Inc., Amsterdam (1988)
6. Steele, G.: Debunking the "expensive procedure call" myth or, procedure call implementations considered harmful or, LAMBDA: The Ultimate GOTO. In: ACM Proceedings of the 1977 Annual Conference, pp. 187–193. ACM Press, New York (1988)
7. Bell, T., Cleary, J., Witten, I.: Text Compression. Prentice Hall, Englewood Cliffs (1990)
8. Charniak, E.: Statistical Language Learning. MIT Press, Cambridge (1993)

9. Manber, U., Myers, G.: Suffix arrays: a new method for on-line string searches. SIAM J. Comput. 22(5), 935–948 (1993)
10. Harman, D., Liberman, M.: TIPSTER, LDC, vol. 1 (1993), http://www.ldc.upenn.edu
11. Broder, A.Z., Glassman, S.C., Manasse, M.S., Zweig, G.: Syntactic clustering of the Web. Comput. Netw. ISDN Syst. 29(8-3), 1157–1166 (1997)
12. Witten, I., Moffat, A., Bell, T.: Managing gigabytes: compressing and indexing documents and images. Van Nostrand Reinhold, New York (1999)
13. Jelinek, F.: Statistical Methods for Speech Recognition. MIT Press, Cambridge (1999)
14. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
15. Church, K.W.: Empirical Estimates of Adaptation: The chance of Two Noriegas is closer to $p/2$ than p^2. In: Coling (2000)
16. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Prentice Hall, Upper Saddle River (2000)
17. Huang, X., Acero, A., Hon, H.-W.: Spoken Language Processing. Prentice Hall, Upper Saddle River (2001)
18. Baayen, R.H.: Word Frequency Distributions. Kluwer Academic Publishers, Dordrecht (2001)
19. Yamamoto, M., Church, K.: Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. Computational Linguistics 27(1), 1–30 (2001)
20. Xu, Y., Umemura, K.: Improvements of Katz K Mixture Model. Information and Media Technologies 1(1), 411–435 (2006)
21. Umemura, K.: www.cicling.org/2009/Umemura-Church/

Evaluation of the Syntactic Annotation in EPEC, the Reference Corpus for the Processing of Basque

Larraitz Uria, Ainara Estarrona, Izaskun Aldezabal, Maria Jesús Aranzabe, Arantza Díaz de Ilarraza, and Mikel Iruskieta

IXA Group (Natural Language Processing)
University of the Basque Country
Computer Science Faculty
Manuel Lardizabal Pasealekua 1
E-20018 Donostia
{larraitz.uria, ainara.estarrona, izaskun.aldezabal, maxux.aranzabe, a.diazdeilarraza, mikel.iruskieta}@ehu.es

Abstract. The aim of this work is to evaluate the dependency-based annotation of EPEC (the Reference Corpus for the Processing of Basque) by means of an experiment: two annotators have syntactically tagged a sample of the mentioned corpus in order to evaluate the agreement-rate between them and to identify those issues that have to be improved in the syntactic annotation process. In this article we present the quantitative and qualitative results of this evaluation.

Keywords: Basque corpus, dependency-based syntactic annotation, evaluation, annotators' agreement-rate, Kappa agreement index.

1 Introduction

This work has been carried out in the framework of the Ixa research group[1], where resources such as data-bases and corpora annotated at different linguistic levels are being developed.

The EPEC corpus [1], considered in the Ixa group a reference corpus for the processing of Basque, is so far annotated at syntactic level, with dependencies' relations; and a part of the semantic annotation (the nominal part) is also finished.

Every annotation process has to be evaluated in order to warranty its quality. In this paper, we present the qualitative and quantitative evaluation of the dependency-based annotation of a sample of EPEC. The aim of this evaluation is twofold: to measure the agreement-rate between the annotators and to identify those issues that have to be improved in the syntactic annotation process.

The paper is organized as follows: in section 2 we explain some features of the EPEC corpus. Section 3 deals with the model adopted for the syntactic analysis and annotation. In section 4 we present the evaluation carried out: first, the quantitative

[1] http://ixa.si.ehu.es/Ixa

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 72–85, 2009.

evaluation, indicating the data obtained from the Kappa agreement index based on [2] is explained, and secondly, the qualitative evaluation, which is illustrated with some representative examples. Finally, some conclusions and future works are outlined in section 5.

2 The EPEC Corpus

The EPEC Corpus is a 300,000 words collection of written standard Basque. It is aimed to be a "reference" corpus for the development and improvement of several NLP (Natural Language Processing) tools we are developing for Basque [3].

The corpus has been linguistically annotated at different levels: it was first morphologically analyzed by means of MORFEUS [4] and then manually disambiguated [5]. In the manual tagging, each word-form of the whole corpus was assigned its corresponding analysis at the segmentation level: part-of-speech, number, definiteness and declension case. After the morphological disambiguation, other modules within the chunker IXATI [6], [7] such as complex postpositions, name-entities, multiword lexical units and morphosyntax were applied. The manual dependency-based syntactic annotation started precisely at this stage. This way, we have nowadays a Treebank for Basque of 300,000 words completely and correctly analyzed at dependency level [8], [9]. The semantic annotation has been so far carried out at the nominal part [10], based on Euskal Wordnet [11].

Although this is the process followed when annotating manually the dependency relations, we have also developed grammars and tools for automatic disambiguation [12], including the disambiguation of syntactic functions. For this purpose, we have made use of the Constraint Grammar (CG) formalism [13], [14], and stochastic methods have been also applied [15]. At present, the analyzers and disambiguation tools for the dependency-based syntactic annotation are being developed [16], [9]. In all cases, the correct data (the manually disambiguated data) is used both to validate the grammars and disambiguation tools as well as to apply methods of machine learning [15].

3 Syntactic Annotation

Syntactic annotation means adding syntactic information to a text using special markers which provide information about the syntactic structures of sentences; e.g. labelled bracketings or symbols indicating dependency relations between words.

Annotation schemes usually differ in the labels used and in some cases the nodes composing the trees have different functions. However, most schemes provide a similar constituency-based representation of relations among the syntactic components (see [17]). In contrast, dependency schemes (e.g., [18]; [19]; [20]) do not provide a constituent-based analysis but rather specify explicitly the grammatical relations among the components of a sentence.

The debate whether a constituency-based or a dependency-based formalism should be used when developing a Treebank is still open. In fact, some researchers have

adopted a middle-ground position, as in [21], where they use the dependency-based approach only to combine the basic components of the sentence (noun phrases, prepositional phrases and the verb).

The above described formalisms may be suitable in general. However, the success and influence they may exert on applications highly depend on the language under consideration. After considering a number of trials presented in [22], [23], and [24], we have decided to follow the dependency-based procedure to deal with the free word-order structure displayed by the Basque syntax. The dependency-based formalism describes the relations between components (i.e. word-forms). This way, for each sentence in the corpus we explicitly determine the syntactic dependencies between the head and its dependants. This is the formalism used in the Prague Dependency TreeBank (PDT) [25], which is considered as the first consistently annotated Treebank based on dependencies.

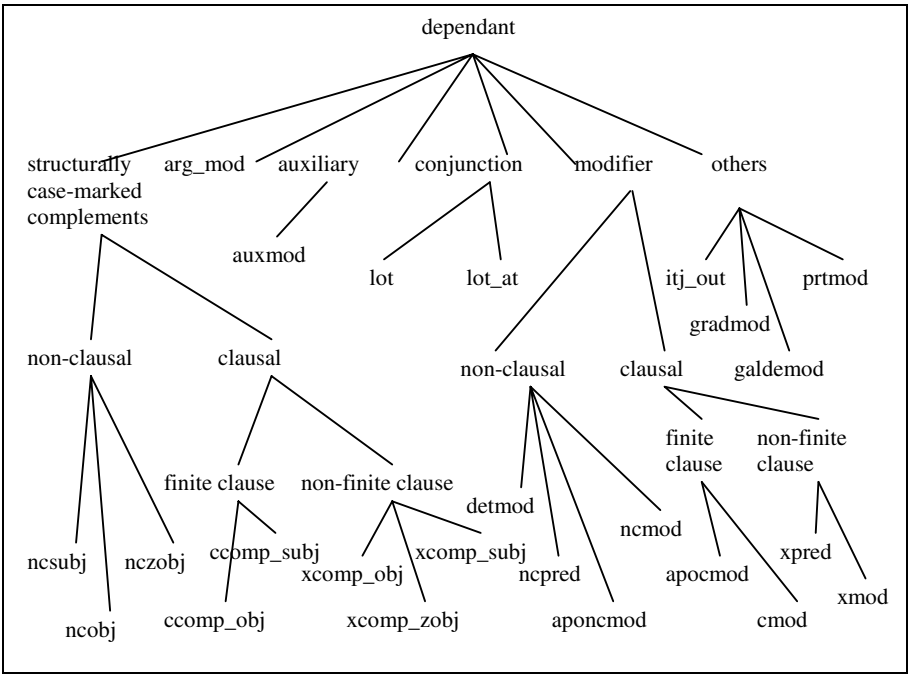


Fig. 1. Hierarchy of grammatical relations

The dependency scheme we have adopted is based on [26] and we defined the hierarchy relations shown in figure 1, following that scheme. This hierarchy consists of several general levels, which are further specified in subsequent levels. Structurally case-marked complements, thematic roles (arg_mod)[2], modifiers, auxiliaries and conjunctions belong to the general level. In addition, structurally case-marked

[2] Although this field is previewed, it is not filled yet. We have planned to complete this task in future steps, when treating semantics.

complements are divided into noun phrases and clauses. Each continuous gradation achieves further specification by taking into account their grammatical function (e.g. ncsubj, ncobj, and nczobj). Below we present the representation of the grammatical relations regarding the mentioned dependency-tags, which are structurally case-marked non-clausal (nc) complements:

ncsubj (Case, Head, Head of NC, Case-marked element within NC, subj)
ncobj (Case, Head, Head of NC, Case-marked element within NC, obj)
nczobj[3] (Case, Head, Head of NC, Case-marked element within NC, ind.obj)

For example, the sentence "Aitak haurrari sagarra eman dio" ('Father has given an apple to the child') is annotated using the three mentioned tags for the *aitak* ('father'), *haurrari* ('to the child') and *sagarra* ('an apple') dependants, typed in italic:

ncsubj (erg, eman, ***aitak***, aitak, subj)
nczobj (dat, eman, ***haurrari***, haurrari, ind.obj)
ncobj (abs, eman, ***sagarra***, sagarra, obj)

Dependency relations can also be represented by a tree structure, as in Figure 2. Head is shown as node at the upper end of branches and dependants are shown at the lower end of branches. Thus, "eman" is analyzed as the head of "aitak", "haurrari", "sagarra" and "dio".

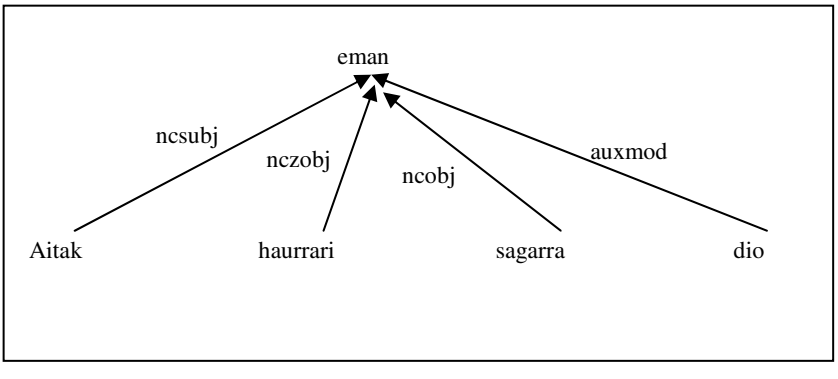


Fig. 2. The tree structure of the sentence "Aitak haurrari sagarra eman dio"

The main features of the syntactic annotation are the following ones:

- Only explicit elements are annotated; that is, neither dropped elements (such as pro, PRO or other types of ellipsis), control structures nor co-references are marked.
- The order in the annotation is not relevant. The dependency-based formalism has been chosen actually to allow free word-order representation, which is

[3] *nczobj* would be equivalent to the English nciobj (non-clausal indirect object).

appropriate for free order languages such as Basque as well as to represent discontinuous multiword expressions[4].
- The adopted formalism does not belong to any concrete theory; it is thought to be a neutral formalism.

4 Evaluation

In this section we explain the methodology used for the evaluation as well as the two types of evaluation we have carried out.

4.1 Methodology

In order to evaluate the dependency-based annotation, 50 sentences including the most common verb have been selected at random and annotated by two taggers. This verb is "izan" ('to be'), which usually appears next to other verbs which are also analyzed.

The purpose of the evaluation has been twofold. On the one hand, we wanted quantitative results: some statistics concerning the agreement-rate between the two annotators. On the other hand, we wanted to analyze in which cases the annotators disagreed in the tagging process to identify both the phenomena to be improved for future annotations and the cases which are intrinsic to the language's ambiguity and complexity.

For the quantitative evaluation, we have applied the Kappa agreement index based on [2]. In addition, we present the tags used by the annotators with their absolute number (see table 2). Taking into account that in the annotation process some sentences have been excluded for several reasons (section 4.2), we have distinguished two disagreement types: disagreements in the annotation and disagreements when excluding sentences. The Kappa index is applied in the annotated data.

For the qualitative evaluation, we have taken into account the linguistic phenomena of each example in which the annotators disagreed as well as the possible reasons which caused those disagreements.

4.2 Quantitative Evaluation

The annotators can exclude sentences which are syntactically incorrect or extremely long. Consequently, it is necessary to distinguish two kinds of disagreements: disagreement when annotating the same sentence and disagreement when one annotator annotates a sentences and the other one excludes it. The Kappa index is then applied only in the first case: the comparison is made in sentences that both annotators have annotated. For this reason, the statistics we present in this paper will be divided into these two cases.

[4] It does not mean that the order of the words in the sentence is not relevant from other point of view such as semantics, but from a pure phrasal and functional point of view (and taking into account that in Basque the functions are not changed depending on the order of the phrases) the dependency-based formalism does not require annotators to maintain the exact order, and then they can tag the word in the preferred order.

The general percentages of the two disagreement types are presented in Table 1.

Table 1. General percentages of the two disagreement types

	No. of disagr.	%
Disagreement when annotating	30	88,23
Disagreement when excluding sentences	4	11,76

4.2.1 Disagreements When Tagging a Sentence

Before explaining the Kappa results, in table 2 we show the results of the annotation, specifying the number of labels used by each annotator and the match between them.

Table 2. Matching between the annotators (excluded sentences are not included)

	Labels	Annot.1	Annot.2	Agr.	Disagr.	%
1	ncmod	125	118	100	43	27,92
2	ncsubj	43	45	36	16	10,38
3	lot	40	41	27	27	17,53
4	ncpred	35	34	23	23	14,93
5	detmod	20	21	19	3	1,94
6	auxmod	18	19	18	1	0,64
7	entios	9	15	8	8	5,19
8	ncobj	10	10	8	4	2,59
9	cmod	9	9	7	4	2,59
10	lotat	6	6	5	2	1,29
11	postos	3	2	2	1	0,64
12	xcomp_obj	2	3	2	1	0,64
13	xmod	4	3	2	3	1,94
14	aponcmod	6	5	1	9	5,84
15	ccomp_obj	2	3	1	3	1,94
16	gradmod	1	1	1	0	0
17	menos	1	1	1	0	0
18	nczobj	1	1	1	0	0
19	haos	1	4	0	5	3,24
20	ccomp_subj	1	0	0	1	0,64
	TOTAL	337	341	262	154	99,99

In total there have been 30 discrepancies between annotators, that is, different options when annotating the same word (see section 4.3 Qualitative evaluation, for further explanations), and they have caused 154 disagreements. Annotator 1 has used 75 labels; annotator 2 has used 79. Sometimes the disagreement is not in the label but inside the label (see example 4 in section 4.3.1). One option can often imply disagreements in more than one label (example 3 in section 4.3.1). That is way 30 discrepancies of the annotators carry out 154 disagreements in labels.

We have based on Landis & Koch [2] to get the Kappa measures (which is a more robust method than the simple percentage of agreements), and Cohen [27] for the coefficients of the agreement-rate (table 3).

Table 3. Coefficients for the agreement-rate based on [27]

Kappa Statistic	Strength of agreement
<0.00	Poor
0.0-0.20	Slight
0.21-0.40	Fair
0.41-0.60	Moderate
0.61-0.80	Substantial
0.81-1.00	Almost perfect

The results obtained from the Kappa inter-annotator agreement-rate are shown in table 4. Two Kappa measures are provided: one belongs to annotator 1, taking annotator 2 as gold standard, and vice versa.

Table 4. Kappa results

	Used tags	Agreement	Percentage of the agreement Pr(a)	Kappa	Possible agr. at random Pr(e)
Annot. 1	337	262	0.7774	0.73	0.1825
Annot. 2	341	262	0.7683	0.72	0.1782

The agreement-rate between the two annotators (0.7683, 0.7774) is in fact considerable; there is almost one point difference between them. This is because annotator 1 has used 337 tags while annotator 2 has used 341. Annotators' agreement is substantially bigger than possible agreement at random (0.1782).

Although the results are acceptable, we think they could be improved if we chose other verb instead of "izan" ('to be'). In fact, ambiguity in the tags ncsubj and ncpred is probably bigger in this verb than in others. We have observed it in the qualitative evaluation (see example 13 in section 4.3.1).

4.2.2 Disagreements When Excluding Sentences

The agreement-rate between the two annotators when tagging and excluding sentences is 92 %. Therefore, in 8 % of the cases there was disagreement (table 5).

Table 5. Agreement-rate between the two annotators when tagging or excluding a sentence

	Annot. 1	Annot. 2	Agr.	Disagr.	TOTAL
Analysed clauses	43	43	41	4	86
Excluded clauses	7	7	5	4	14

4.3 Qualitative Evaluation

After showing the results of the quantitative evaluation, in this section we present in which cases and why annotators did not agree. We first focus on the linguistic phenomena in which the annotators disagreed, and then on the possible reasons for disagreements.

The linguistic phenomena in which the annotators disagreed are:

1- **Different head:** the annotators selected a different head.
2- **Different label:** the annotators selected a different dependency-tag.
3- **Different case:** the annotators selected a different case in the dependency-tag.
4- **Incomplete tree:** the annotator forgot a tag and the tree is not complete.
5- **Excluded sentence:** the annotator excluded the sentence.

As concerns the reasons for disagreements, we have distinguished three types: ambiguity, erroneous annotation and gaps; the second and third types consist of some subtypes.

1- Ambiguity: The annotators analyze the sentence in different ways but both analyses are possible and correct.

2- Erroneous annotation:

- The annotator does not realize s/he has made an error: the annotator does not doubt the analysis of a word when it is in fact wrong and not even mentioned in the manual.
- The annotator does not follow the manual.
- The annotator does not use auxiliary labels when needed. The correct treatment of multiword expressions require some auxiliary tags to join the elements that form the expression when the provided syntactic analysis has not treated multiword expressions as such. The annotator may not use those auxiliary tags to join the multiword expressions.

3- Gaps:

- Gaps in the manual: the manual does not cover all the possible existing linguistic phenomena.
- Gaps in the previous modules, such as the analysis of complex postpositions, multiword lexical units, name-entities or morphosyntax.

In the following tables, we show the data concerning the disagreements between the two annotators when tagging a sentence (table 6) and when excluding a sentence (table 7).

The most frequent disagreement is due to annotators' errors and the most frequent disagreement subtype lies in the erroneous annotation group (56.66 %). Gaps in general represent the 3.3 %. Most of the gaps are caused because the annotation manual is still being built. 10 % of the cases occurred because of language's ambiguity. Fewer disagreements come from specific errors.

Table 6. Disagreements when annotating sentences

		Errors			Gaps		
	Anbig.	**Annotation**	**Following manual**	**Using Auxiliary labels**	**Manual**	**Other modules**	**Total**
Head	3	1	1	0	4	0	9
Label	0	2	2	6	4	0	14
Case	0	1	2	1	1	1	6
Tree	0	1	0	0	0	0	1
Total	3	5	5	7	9	1	30
Total	3	17			10		30
%	10 %	56.66 %			33.3 %		100 %

Table 7. Disagreements when excluding sentences

	Annotator's error	**Manual gap**
Sentence limit	1	0
Length	0	1
Grammatical error	1	0
Total	2	1
Total	3	
%	9 %	

4.3.1 Some Representative Examples

In this section we show some examples which represent the most common types of disagreements and the main reasons for them. However, in [8], more of them are deeply explained.

--- **Different head**

▪ Neither all the sentence connectors nor conjunctions are listed in the manual, but only general considerations are remarked and illustrated with some examples. Therefore, one of the annotators has sometimes considered an element a conjunction and the other one a sentence connector or an adverb. Consequently, the head also results different. That is the case of the elements "besteak beste" ('between others') and "ondorioz" ('in consequence') in the examples bellow.

> [1] Granadan, Jaime Mayor Oreja Barne ministroa, Juan Cotino Espainiako Poliziako zuzendaria, PPko Teofila Martinez eta Jose Moratalla Granadako alkatea izan ziren, ***besteak beste***.
>
> In Granada, there were the Home Secretary Jaime Mayor Oreja, the head of the Spanish Police Juan Cotino, Teofila Martinez from PP and the mayor of Granada Jose Moratalla, **among others**.

In example [1], one annotator interpreted “besteak beste” as a coordinating conjunction and the other one as an explicative sentence connector.

> [2] Madrilen arabera, laguntza horiek "orokorrak" dira, eta **ondorioz**, ez dira legez kanpoko Estatu laguntzak.
>
> According to Madrid, those subsidies are “general” ones, and **as a consequence**, they are not illegal state subsidies.

In example [2], one annotator interpreted “ondorioz” as an adverb and the other one annotated it as a conjunction.

--- Different label

▪ Multiword expressions constitute another controversial issue in the tagging task. There is a wide range of multiwords in Basque. Besides, there are new expressions continuously being created. As a consequence, it is quite common to find units in the corpus that in previous syntactic analysis have not been treated as such, that have not been correctly detected by the automatic tool. Because the range of possibilities is high, annotators do not agree when jointing the multiword expressions. In example [3], for instance, they do not agree when delimiting the multiword entity: one annotator considers “*Justizia eta Lan sailburua*” (‘Justice and Job member’) a whole entity and the other one joins the two words “*Justizia eta Lan*” (‘Justice and Job’) with the coordinating conjunction “eta” (‘and’).

> [3] Eusko Jaurlaritzaren izenean Sabin Intxaurraga **Justizia eta Lan sailburua** izan zen hiletetan.
>
> On behalf of the Basque Government, the **Justice and Job member** Sabin Intxaurraga was in the funeral.

--- Different case

▪ Sometimes different case might be assigned because of a gap in the manual. When explaining how to treat the punctuation marks that work as conjunctions, there are no specifications regarding which kind of relations should be added. One annotator decides to annotate them always as “emen” (coordinating conjunction) and the other annotator chooses it depending on the sentence. For instance, in example [4] annotator 1 uses the disjunctive relation “haut” and annotator 2 the coordinating conjunction “emen”.

> [4] Nik uste dut aldi honetan indarkeria ez dela bakarrik Gaza, Zisjordania **edo** Jerusalem Ekialdeko palestinarren kontra.
>
> I think that this time violence is not only against the Palestinian of Gaza, Zisjordania **or** East Jerusalem.

--- Excluded sentences

▪ In the manual the possibility of excluding sentences is previewed. One of the reasons for excluding a sentence is its length. Nevertheless, this criterion is not exactly defined and each annotator is free for excluding sentences based on his/her subjectivity. Although they agree in many cases, there are some differences. Sometimes, the sentence is long but not difficult, and the tagging results easy. In this case, one annotator decides to tag it, and the other one does not. Maybe, we should specify a number of words to take into account to exclude long sentences.

▪ Annotator's error for not following the manual. In the manual sentences' boundaries are clearly defined. However, an annotator may not take it into account and s/he may exclude a sentence for being "wrong-limited sentence" when it is actually well delimited. In example [5], the colon is not considered a boundary by one annotator (although it is defined as such in the manual) and s/he excludes it.

[5] Bi dira ezaugarri garrantzitsuenak Zenarruzabeitiaren arabera:

There are two main characteristics, according to Zenarruzabeitia:

--- Ambiguity

There are some ambiguous sentences that can have more than one interpretation. Ambiguous sentences can be, therefore, differently annotated.

In example [6], ambiguity lies in "*these last days*" since it can be "*it is obvious these last days*" or "*these last days has attacked*".

[6] Begi bistakoa da **azken egunotan** Israelek palestinarren kontra egindako eraso oldeak esanahi argia duela guretzat .

It is obvious **these last days** Israel has harshly attacked against Palestinian and this attack has a clear meaning for us.

In example [7], the word "*albistea*" ('new') can be considered either the subject (*ncsubj*) or the predicate (*ncpred*) of the verb "*izan*" ('to be'). There is not any clear criterion in the Basque grammar to disambiguate this kind of attributive relations with the verb "*izan*" ('to be'). Neither the manual has a concrete rule for that. Therefore, annotators may not agree when tagging this kind of sentences[5].

[7] Nazioarteko laburretako lehen **albistea** da argazkia duen bakarra.

The first brief international **new** is the only one having a photo.

[5] When the annotator realizes that the sentence is in fact ambiguous, s/he chooses one analysis and then marks the sentence as ambiguous.

5 Conclusions

In this paper we have presented the results obtained from the evaluation of the syntactic annotation of the EPEC corpus. The results have been satisfactory: there has been a substantial agreement-rate between the two annotators, following [27] coefficients. However, we think the result would have been better if we had chosen another verb different to "*izan*" ('to be'), since this verb shows ambiguity in the tags *ncsubj* and *ncpred*. This will be an interesting issue to analyze in the future. On the other hand, it should be necessary to evaluate a bigger set of sentence to get more reliable data and confirm or refuse either the statements or the evaluation data we present here. This is another planned task for the near future.

The evaluation process has made explicit some gaps to be improved in the annotation manual. Therefore, if we consider the disagreements caused because the phenomena was not clearly specified in the manual and we clarify those issues, the results would improve in 33 % (table 6). In fact, the development of an exhaustive annotation manual is an ongoing work: it is almost impossible to cover all linguistic phenomena a priori, although we think we have got a good base.

One objective way to improve the results would be to exclude too long sentences, sentences consisting of more than a concrete number of words (i.e. more than 40 words). Furthermore, the state of mind of the annotator is not always the same, which is also influential in the annotation task.

To get a 100 % of agreement-rate between the two annotators is in fact a utopian goal, since language is intrinsically ambiguous and open to different but correct analysis.

References

1. Aduriz, I., Aranzabe, M.J., Arriola, J.M., Atutxa, A., Díaz de Ilarraza, A., Ezeiza, N., Gojenola, K., Oronoz, M., Soroa, A., Urizar, R.: Methodology and steps towards the construction of EPEC, a corpus of written Basque tagged at morphological and syntactic levels for the automatic processing. In: Wilson, A., Rayson, P., Archer, D. (eds.) Corpus Linguistics Around the World, Rodopi, Netherland, pp. 1–15 (2006a)
2. Landis, J.R., Koch, G.G.: The measurement of Observer Agreement for Categorical Data. Biometrics 33, 159–174 (1977)
3. Bengoetxea, K., Gojenola, K.: Desarrollo de un analizador sintáctico estadístico basado en dependencias para el euskera. In: Procesamiento del Lenguaje Natural, SEPLN 2007. Universidad de Sevilla (2007)
4. Alegria, I.: Euskal morfologiaren tratamendu automatikorako tresnak. Doktoretza-tesia, Euskal Herriko Unibertsitatea (UPV/EHU) (1995)
5. Aldezabal, I., Ceberio, K., Esparza, I., Estarrona, A., Etxeberria, J., Iruskieta, M., Izagirre, E., Uria, L.: EPEC (Euskararen Prozesamendurako Erreferentzia Corpusa) segmentazio-mailan etiketatzeko eskuliburua, UPV/EHU / LSI / TR 11-2007 (2007a)
6. Aduriz, I., Díaz de Ilarraza, A.: Morphosyntactic disambiguation and shallow parsing in Computational Processing of Basque. In: Oyharcabal, B. (ed.) Inquiries into the lexicon-syntax relations in Basque. ASJUren gehigarria. Euskal Herriko Unibertsitatea (UPV/EHU), Bilbo (2003)

7. Aduriz, I., Aranzabe, M.J., Arriola, J.M., Díaz de Ilarraza, A.: Sintaxi partziala. In: Fernández, B., Laka, I. (eds.) Andolin gogoan: Essays in Honour of Professor Eguzkitza, UPV/EHUko Argitarapen Zerbitzua, Bilbo (2006b)
8. Aldezabal, I., Aranzabe, M.J., Arriola, J.M., Díaz de Ilarraza, A., Estarrona, A., Fernandez, K., Iruskieta, M., Uria, L.: EPEC (Euskararen Prozesamendurako Erreferentzia Corpusa) dependentziekin etiketatzeko eskuliburua. UPV/EHU / LSI / TR 12-2007 (2007b)
9. Aranzabe, M.J.: Dependentzia-ereduan oinarritutako baliabide sintaktikoak: zuhaitz-bankua eta gramatika konputazionala. Doktoretza-tesia. Euskal Herriko Unibertsitatea, UPV/EHU (2008)
10. Agirre, E., Aldezabal, I., Estarrona, A., Pociello, E.: A methodology for the joint development of the Basque WordNet and Semcor. In: Dutch SemCor Workshop, Amsterdam (2008)
11. Pociello, E.: Euskararen ezagutza-base lexikala: Euskal WordNet. Doktoretza-tesia, Euskal Filologia Saila (UPV/EHU). Leioa (2008)
12. Aduriz, I., Aldezabal, I., Alegria, I., Artola, X., Ezeiza, N., Urizar, R.: EUSLEM: A Lemmatiser / Tagger for Basque. In: Proc. EURALEX 1996, Part 1, pp. 17–26. Góuml;teborg, Sweden (1996)
13. Karlsson, F., Voutilainen, A., Heikkilä, J., Anttila, A.: Constraint Grammar: A Language-independent System for Parsing Unrestricted Text. Mouton de Gruyter, Berlin (1995)
14. Tapanainen, P., Voutilainen, A.: Tagging Accurately – Don't guess if you know. In: Proceedings of the 4th Conference on Applied Natural Language Processing, ANLP 1994 (1994)
15. Ezeiza, N.: Corpusak ustiatzeko tresna linguistikoak. Euskararen etiketatzaile morfosintaktiko sendo eta malgua. Doktoretza-tesia, Euskal Herriko Unibertsitatea (UPV/EHU) (2003)
16. Aduriz, I.: EUSMG: morfologiatik sintaxira murriztapen gramatika erabiliz. Euskararen desanbiguazio morfologikoaren tratamendua eta azterketa sintaktikoaren lehen urratsak. Doktoretza-tesia, Euskal Herriko Unibertsitatea (UPV/EHU) (2000)
17. Abeillé, A.: Treebanks: Building and Using Parsed Corpora. Kluwer Academic Publisher, Dordrecht (2003)
18. Sleator, D., Temperley, D.: Parsing English with a link grammar. In: Third International Workshop on Parsing Technologies (1993)
19. Järvinen, T., Tapanainen, P.: A Dependency Parser for English. Technical Report, nº TR-1, Department of General Linguistics. University of Helsinki (1997)
20. Bunt, H., Carroll, J., Satta, G.: New Developments in Parsing Technology. Text, speech and language technology, vol. 23. Kluwer Academic Publishers, Dordrecht (2004)
21. Montemagni, S., Barsotti, F., Battista, M., Calzolari, N., Corazzari, O., Lenci, A., Zampolli, A., Fanciulli, F., Massetani, M., Raffaelli, R., Basili, R., Pazienza, M., Saracino, D., Zanzotto, F., Mana, N., Pianesi, F., Delmonte, R.: Building the Italian Syntactic-Semantic Treebank. In: Abeillé, A. (ed.) Building and Using Parsed Corpora, pp. 189–210. Kluwer Academic Publisher, The Netherlands (2003)
22. Skut, W., Krenn, B., Brants, T., Uszkoreit, H.: An Annotation Scheme for Free Word Order Languages. In: Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington, DC, USA, pp. 88–95 (1997)
23. Järvinen, T., Tapanainen, P.: Towards an Implementable Dependency Grammar. In: Proceedings of the Workshop on Processing of Dependency-Based Grammars, COLING-ACL 1998, Montreal (1998)

24. Oflazer, K., Zeynep, D., Tür, H., Tür, G.: Design for a Turkish Treebank. In: Proceedings of Workshop on Linguistically Interpreted Corpora. Bergen (1999)
25. Böhomovà, A., Hajic, J., Hajicova, E., Hladka, B.: The PDT: a 3-level annotation scenario. In: Abeillé, A. (ed.) Treebanks: Building and Using Parsed Corpora. Kluwer Academic Publishers, Dordrecht (2003)
26. Carroll, J., Briscoe, T., Sanfilippo, A.: Parser evaluation: a survey and a new proposal. In: Proceedings of the First International Conference on Language Resources and Evaluation, Granada, Spain, pp. 447–454 (1998)
27. Cohen, J.: A coeffcient of agreement for nominal scales. Educational and Psychological Measurement, 37–46 (1960)

Reducing Noise in Labels and Features for a Real World Dataset: Application of NLP Corpus Annotation Methods

Rebecca J. Passonneau*, Cynthia Rudin†, Axinia Radeva‡, and Zhi An Liu§

Columbia University, New York, NY 10027, USA
*becky@cs.columbia.edu, (†cr2363,§zl2153)@columbia.edu,
‡axinia@hotmail.com

Abstract. This paper illustrates how a combination of information extraction, machine learning, and NLP corpus annotation practice was applied to a problem of ranking vulnerability of structures (service boxes, manholes) in the Manhattan electrical grid. By adapting NLP corpus annotation methods to the task of knowledge transfer from domain experts, we compensated for the lack of operational definitions of components of the model, such as *serious event*. The machine learning depended on the ticket classes, but it was not the end goal. Rather, our rule-based document classification determines both the labels of examples and their feature representations. Changes in our classification of events led to improvements in our model, as reflected in the AUC scores for the full ranked list of over 51K structures. The improvements for the very top of the ranked list, which is of most importance for prioritizing work on the electrical grid, affected one in every four or five structures.

1 Introduction

This paper illustrates how a combination of information extraction, machine learning, and NLP corpus annotation techniques was applied to a problem of ranking vulnerability of structures (manholes and service boxes) in the Manhattan electrical grid. Institutions of all sorts collect and archive large amounts of data. The value of the data to the institution depends in part on whether methods can be developed to make use of it. Information extraction, defined as the task of organizing and normalizing data taken from unstructured text in order to populate tables in structured databases, has obvious relevance, as does machine learning. Automated techniques for corpus annotation clearly support the task of information extraction. What is less obvious, and potentially of greater impact, is that the practices developed in the NLP community for manual annotation and classification of documents provide an effective means to arrive at clearer problem definitions. By adapting NLP corpus annotation methods to the task of knowledge transfer from domain experts, we compensated for the lack of operational definitions of components of the model, such as *serious event*.

During the 1970s, the Consolidated Edison Company instituted a program of Emergency Control System (ECS) tickets to document calls from customers about potential problems in the electrical grid. Beginning in 1986, after Hurrican Gloria, the ECS program expanded and became more fully utilized. As the

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 86–97, 2009.

archive grew, Con Ed hypothesized that information in the ECS tickets could be used to rank the vulnerability of manholes and other structures in the network to serious events. We worked with Con Ed to refine and test this hypothesis.

A discussion of related work is in the next section, followed by a section presenting background on the scope of the problem, and an example of an ECS ticket for a moderately serious event. The next two sections describe the information extraction (section 4) and machine learning (section 5). In section 6, we present the human annotation task, and the impact on our labeling and feature representation of examples. Section 7 presents the results of a comparison of alternative labelings: a baseline, our initial rule-based labeling, and the rule-based labeling that we now use, derived from the results of the human annotation task. In the final section we discuss the implications and conclusions.

2 Related Work

Machine learning has been used to predict failures in the primary (high voltage) network for Consolidated Edison [1], but not using data extracted from free text fields. Con Ed trouble tickets [2], maintenance logs for complex machinery [3], naval equipment reports [4], aeronautic safety reports [5] and similar sets of documents have been handled using the methods of natural language processing, knowledge modeling, and machine learning for a wide range of goals. Relatively early work [4] showed it was possible to handle fragmentary text using full syntactic and semantic parsing, but involved a much smaller dataset than current work on ticket databases. Devaney and Ram [3] deal with 10,000 logs, all of which pertain to the same machines. They combine unsupervised text clustering with a domain representation modeled in OWL/RDF to classify tickets, then develop a Case-Based Reasoning approach to predict failures. Liddy and her colleagues [2] developed an application for the same type of trouble ticket data we address, but their goal was to assign ECS tickets with a miscellaneous categorization to a more specific ticket type.

Oza [5] is the only work we have seen that deals with a similarly large dataset, and where disagreements among human experts made it difficult to define document classes. They look at two aeronautics report databases that have a combined size of 800,000 reports. Their reports, unlike ECS tickets, generally have a single author, and consist of a readable, discursive narrative. Their end goal is to arrive at a comprehensive, topic-based document classification, whereas our classification task is to scale the severity of events, and we ignore ticket content not relevant to this task. They rely on an existing thesaurus (PLADS) to merge distinct forms of a single term, such as acronyms, abbreviations and phrases, making their documents amenable to a bag-of-words (BOW) document representation, and they use two learning techniques, Support Vector Machines and Non-negative Matrix Factorization. Our early attempts to use BOW features foundered due to the high noise content. In ongoing work, we have been looking at decision trees for document classification, and clustering methods to generate string normalization rules.

3 Background and Example

We have 1,036,732 ECS tickets for 1996 through 2006, from four New York City boroughs. They fall into hundreds of distinct *trouble types*, a ticket category Con Ed assigns to each ECS ticket. The first line of the ticket in Figure 1 explicitly names the category for this ticket, which is SMH (smoking manhole); MHF is for a manhole fire, LV is for a low voltage problem, and so on. For Manhattan, we investigated twenty-two trouble types, comprising 61,730 tickets for the ten-year period. A ticket represents a report about an event or problem that the caller judged to involve the electrical grid. These data are noisy: there are far more tickets than there are distinct *events*, and many more distinct events than those we use for labeling and feature representation. Furthermore, not all tickets mention a specific structure (manhole or service box), and they are not intended to contain a complete description of every event.

To use the ECS ticket data, we addressed four tasks: identifying structures mentioned in the ECS tickets, pruning the tickets to a set of unique events relevant for our modeling task, labeling the events as serious or not, and deriving ECS-based and other features to represent structures. The final three tasks depended heavily on the outcome of the human annotation task.

One of the questions we faced at the outset was what time frame we could make predictions about. In separate work [6], we report exploratory data analysis indicating that, given the data made available to us, we could make predictions based on longer term hotspots. The Manhattan machine learning model described there, and presented to Con Ed, ranks structures for a given one-year period, based on data from prior years. The ranking criterion is the likelihood that a structure will experience a serious event within the current year.

Counts of structures and events give a sense of the scope of our task. There are 51,219 structures in Manhattan. In the 61K tickets we investigate, we extract mentions of 27,235 structures (see section 4). Depending on the definition of *serious event*, we estimate that there are on the order of seventy-five to five hundred serious events per year in Manhattan. Defining more precisely what counts as a "serious" event is the focus of this work. The trouble type of a ticket is a good but not perfect indicator of seriousness for three trouble types (MHX, MHF, MHO), and a moderate indicator for a fourth (SMH). In our most

1 01/21/*YR* 18:45 FDNY-190 REPORTS A SMH *STREET_1* & *STREET_2*
2 01/21/*YR* 19:35 *PERSON* REPORTS THE **TROUBLE HOLE** IS **SB-00001**
3 N/W/C *STREET_1* & *STREET_2*.....FOUND ON ....SMOKING LIGHTY
4 01/21/*YR* 21:55 *PERSON* REPORTS IN **SB-00001** HE FOUND 1 LEG
5 ON THE 5 WIRE NORTH BURNING IN THE STRUCTURE.....CUT/CLEAR
6 ED & RETIED SAME .....COMPLETE.....SS
7 ELIN REPT ADDED FOR INCIDENT:SMH 01/21/*YR* 22:02BY *PERSON_ID*
8 REPORTED BY: FIRE DEPT
9 STRUC MSPLATE TYPE NUMBER COND COVTYP COVFOUND DISTANCE

Fig. 1. Sample ECS ticket (anonymized): serious smoker

recent rule-based classification of events into serious and non-serious, which we know overgenerates, there are 5,115 serious events in Manhattan for the ten year period. If a ticket is one of the most serious trouble types–MHF (fires), MHX (explosions) and MHO (open manholes, or possible explosions)–chances are we classify it as serious (1,481 out of 1,506 tickets, or 98.3%). The additional 3,634 serious events include 3,397 SMHs (smoking manholes)–a somewhat less serious trouble type that we classify as serious about 75% of the time, ACBs (AC burnounts, N=162)–which we classify as serious 2.9% of the time, and a mixed set of thirteen trouble types (N=75). Here we use these six categories of trouble types: MHX, MHF, MHO, SMH, ACB, all others.

ECS tickets can have multiple entries from different individuals, some of which is free text, some of which is automatically entered. In Figure 1, a slightly modified version of a relatively readable ticket that we currently classify as serious, we show only the free text lines we are concerned with here. ECS tickets exhibit the fragmentary language, lack of punctuation, acronyms and special symbols characteristic of trouble tickets [4]. They have a high rate of misspellings (see "lighty" in line 12, a typical misspelling) and line breaks within words (see "clear {newline} ed" in lines 6-7). Evidence to classify this event as serious is the mention of a smoking manhole in line 1 and the degree of smoke in line 3. Evidence that in another context could point to a non-serious event is that the smoking manhole is the report of someone other than an engineer (line 1), and that the cover is on ("found on," line 3). The structure is named as the "trouble hole" (line 2), a domain expression meaning the event in question occurred in this structure. Tickets rarely identify the trouble hole explicitly, and often mention multiple structures.

4 Information Extraction from ECS Tickets

The ECS tickets in our subset range in length from 1 to 550 lines, with a similarly wide range of information content. The information we extract falls into two categories. One category corresponds to domain-specific named entities, where a text string names an object in the domain. The entities we extract consist of cables and structures. Structures have a type (service box versus manhole) and a numeric identifier; see the service box (SB 00001) in Figure 1. The combination of type, number and location provides a triple that is used in retrieving the unique identifier for the structure from Con Ed's asset table. Our structure extraction achieves about 90% accuracy against a known subset [6]. We do not use existing named entity recognizers because the types of entities we identify are domain-specific, and because the text is too divergent from standard orthography (e.g., a high degree of misspellings, acronyms, non-standard symbols, and words with linebreaks within them).

Given a structure involved in an event, the two most critical questions for our task are, was work performed on the structure, and was the structure involved in a serious event. How we addressed the second question is the subject of this paper, and is described in section 6.

5 A Ranking Approach to the Learning Task

The goal of our collaborative effort with Con Ed is to produce a ranking of structures according to vulnerability to serious manhole events. We formulated the task as a *supervised bipartite ranking* problem. Within this framework, machine learning algorithms are used to provide a real-valued score for each structure. It requires a set of *examples* with *labels*. The goal is to construct a model that ranks the positively-labeled examples above the negatively labeled examples, and this model should generalize to other (non-labeled) examples chosen from the same probability distribution.

In this domain, a structure changes over time as events occur, insulation breaks down, cables are repaired or replaced. Thus our examples consist of structures paired with a given time frame, which in our case is a year. A structure gets a positive label if it was the trouble hole of a serious event during the relevant year (Y_i), and a negative label otherwise.

A structure is represented by a set of features that characterize the structure before the year Y_i from which the model is built. Figure 2 lists the features we use for the learning models in this paper. The cable feature (F5) is one of several alternate features we have used to capture density of cables in a structure.

Much of our effort has been devoted to developing an accurate, streamlined, interpretable and intuitive model, as described in [6]. Four of five features pertain to the events a structure has been involved in. These ECS-based features play a key role for the top of the ranked list, but provide little information regarding mid- or low-ranked structures; conversely, the cable feature (F5) has little impact at the top of the ranking, but plays a large role for mid-ranked structures. A central issue for our work is that the labels and ECS-based features both depend on our ability to identify relevant events, and to classify them as serious or not.

We construct a training set (Y_i=2005) and a test set (Y_i=2006). We determine whether the model is statistically predictive by constructing the model from the training set and using it to predict the labels in the test set. Consider the structure SB-00001 mentioned in the ticket in Figure 1, which was classified as a serious ticket. If the ticket date is in 2006, then this ticket pertains to the structure's label in the test data. If the ticket date is in 2005, the ticket pertains instead to the label of SB-00001 in the training data, and to its feature representation in the test data. In the latter case, the structure is the trouble hole of the event, so the values of F1-F4 would all be incremented.

	ECS-based features
F1	number of times structure is a trouble hole between the start of 1996 and Y_i
F2	number of ECS tickets mentioning the structure between the start of 1996 and Y_i
F3	number of times structure is a trouble hole in the 3 year period before Y_i
F4	number of ECS tickets mentioning the structure in the 3 year period before Y_i
	Other feature: cable-based
F5	number of neutral mains cables in the structure

Fig. 2. Five features (F1-F5) used to represent structures

The machine learning algorithm in essence maximizes a proxy for the AUC (area under the Receiver Operator Characteristic curve), or a weighted version of the AUC, which can be viewed as a measure of ranking quality. Machine learning algorithms for this task include RankBoost [7] the P-Norm Push [8] and IR-Push (used in [6]) and SVM-perf [9], used in this work.

6 Human Annotation Task

There are three decisions we make for each ECS ticket that enable us to label and represent examples. First, we must decide if the ticket documents a distinct event. Multiple customers might call to report the same event, generating a new ticket for each call, but with the repair work on one ticket that the other tickets will cross-reference. We refer to the latter category as *referred* tickets, and filter them from the dataset. Second, we must decide if the documented event is relevant to the status of the secondary electrical grid (as opposed to the higher voltage primary grid). Third, for each relevant event, we must decide whether it is serious. The annotation task pertains to decisions two and three.

The three domain experts we consulted with had far more than adequate expertise to interpret tickets for us. One of them, who is now a manager, was the programmer and representative to code maps and input ECS databases for 1985-1991. We gradually realized that due to the variation in tickets, the complexity of the domain, and time limitations, we would never acquire sufficient domain expertise through interviews. An even greater obstacle was a lack of operational definitions of relevant, serious and non-serious events. Approximately eighteen months into the project, we are still learning constraints on relevant events, such as the fact that larger buildings occasionally have 265 volt service, but with no connection to the secondary grid (120 volt).

6.1 Assembling and Assessing the Annotations

Human annotations of natural language data are collected for a wide range of purposes, although the most common one today is probably to assemble training and testing data for supervised machine learning tasks. The goal is often to use machine learning to replicate directly a human classification task. Our goal is to use the experts' annotations of trouble tickets to develop a manually derived rule-based classifier, and subsequently to use the features we derive from the trouble tickets to support our distinct machine learning task, namely ranking the vulnerability of structures within a given time frame. The main obstacle we faced in also developing a machine learning approach for classifying the tickets is that we lacked the domain knowledge to create training data ourselves, and lacked access to experts' time. As we describe below, we were able to show significant gains in our machine learning task by extrapolating rules to classify tickets from a small set of 171 expert-annotated tickets.

One hundred and seventy one tickets were selected for the annotation task in a fashion that biased the set in two ways, but that otherwise aimed for an unbiased

selection. First, we created a bias towards a somewhat higher proportion of serious events than in a random subset. The motivation was that after filtering out referred tickets, the proportion of the most serious trouble types (MHX, MHF, MHO) is relatively low (4% for the ten year period versus 14% if SMH is included; 1.6% or 15% for 2005, 2.6% or 12% for 2006), possibly too low to provide a sufficiently general set of examples, given the relatively small dataset that the experts agreed to label. Second, we created a bias towards a higher proportion of tickets in locations that experienced a series of events within a few months. Here the motivation was that we had observed that a structure was more likely to experience a serious event if it had already had a history of serious or non-serious events (see [6]). The way we created this bias was that we first generated ticket histories consisting of a ticket, along with all tickets for locations within a sixty meter radius of the original ticket that occurred in the preceding two month period, and the first ticket for the same location that occurred in the following month, if there was one. We eliminated tickets whose histories contained no serious trouble types. Then we made a random selection of one hundred histories. The final set of histories contained one hundred seventy-one tickets of assorted trouble types, with approximately 20% being serious.

Two experts agreed to do the annotation task. To test whether each expert had self-agreement, four of the tickets were repeated at random positions within a scrambled ordering of the set of tickets. Each was given the tickets in a different order. The two experts worked independently. They had no access to the trouble type, and were asked not to consult any other data sources. The annotation guidelines consisted of one page of instructions asking the experts to classify each ticket into exactly one category: serious, not serious, or not relevant.

Interannotator agreement (IA) coefficients measure agreement above chance, using the formula below, where $p(A_O)$ is the proportion of observed agreement, and $p(A_E)$ of expected agreement.

$$\frac{p(A_O) - p(A_E)}{1 - p(A_E)} \tag{1}$$

Agreement coefficients differ in how to estimate the probability that annotators will agree [10]. The NLP community often relies on Cohen's κ [11] or Krippendorff's α [12]. In practice, their values are often quite close. How to interpret IA values is the subject of much debate (see review in [10]). For Landis and Koch [13] in the medical arena, values between 0.40 and 0.60 are considered moderate; Krippendorff [12] recommends that for Content Analysis, values above 0.67 support *cautious conclusions* and lower values are suspect.

Trouble type is not a good means of classifying the seriousness of events: IA between a baseline classification based on trouble type and the expert consensus (for the tickets where there is consensus) is poor (κ=0.25; α=0.20). Table 1 gives a breakdown of the 171 tickets. It shows that experts identified 29 serious tickets compared with a baseline of 36, 82 precursor tickets compared with a baseline of 127, 21 irrelevant tickets versus 8, and disagreed on 39 (22.8%). In thirteen of the disagreements, one expert classified the ticket as serious; we asked them to resolve the disagreements, yielding four more serious tickets.

Table 1. Human classification of 171 events

	Serious		Non-serious		Not relevant		
	Baseline	Expert	Baseline	Expert	Baseline	Expert	Disagree
ACB	0	3	21	16	0	0	2
MHO	2	2	0	0	0	1	0
MHF	5	3	1	0	0	1	0
MHX	2	2	0	0	0	0	0
SMH	27	15	0	7	0	3	2
Other	0	4	106	58	8	17	35
Totals	36	29	127	82	8	21	39

IA is measured on the 171 unique tickets, using the experts' first response to the one ticket where they disagreed with themselves. Here, κ is 0.4863 and α is 0.4865, or about halfway between 0 (chance distribution) and 1 (perfect agreement). For the four tickets that were randomly duplicated, experts agreed with themselves three of four times.

The moderate IA values between the experts indicates they agree well above chance, yet there is also a fair degree of subjectivity; data from different annotators might lead to different results for some cases. We concluded from the poor performance of the baseline classification of events against the yardstick of expert judgements that we needed an alternative classification of ECS events, despite the relatively modest IA between the experts. We had already begun developing a rule-based approach to sorting tickets into the three categories, based on the domain knowledge that we were slowly acquiring. To show the evolution of our event classification, we use three alternative classification methods: a baseline based on trouble type and a single length constraint (at least three lines per ticket), our pre-annotation rule-based method (Rules1), and the rule-based method we developed based on the annotation task (Rules2).

6.2 Reducing Noise in Labels and Features

We aimed to improve the precision of event classification overall, and to improve the recall of serious events. For tickets both experts agreed on, we used regularities observed within the three classes of tickets to hypothesize constraints to add to our classification rules. We applied the constraints to our database of tickets, manually evaluated random samples, then reiterated until the results of random sampling appeared to meet our two goals. The changes to our rules affected all three classes: non-relevant events, serious events, and precursor events.

Table 2 shows changes in the distribution of serious versus non-serious tickets by trouble type across the three classification methods. The most dramatic changes are the reduction in the number of relevant tickets, which reduces the size of both the non-serious and serious classes by 26.5%, and the big shift in the classification of SMHs from 100% serious down to 75.5% serious. The number of relevant events decreased from the baseline of 38,911 to 28,987 in the first rule based approach (Rules1), to 28,587 in the final rule-based approach (Rules2).

Table 2. Changes in the distribution of serious versus non-serious tickets

	Non-Serious			Serious		
Type	Baseline	Rules1	Rules2	Baseline	Rules1	Rules2
MHX	0	2	0	179	177	160
MHF	0	79	21	1011	931	829
MHO	0	38	4	595	549	492
SMH	0	584	1105	4906	4284	3397
ACB	6171	4594	5364	192	582	162
Other	25776	14967	16978	81	2210	75
Totals	31947	20264	23472	6964	8733	5115

Tickets were dropped on the basis of several criteria, including the voltage value (see above), and changes to length constraints on several categories of tickets.

The number of precursor events decreased from 31,947 in the baseline to 20,264 in Rules1, then increased to 23,472 in Rules2. The percentage of all relevant events that are precursors decreased to 69.91% in Rules1 from 82.10% in the baseline, and increased back to the same percentage of 82.11% in Rules2.

The reduction in the number of SMH tickets classified as serious involved manual identification of a dictionary of phrases indicative of serious events, such as "manhole explosion," which is unambiguous but rare, or "mh smoking heavy" which has many variants (including "smoking lightly"), and other phrases with the words "fire" and "blown." For each *dictionary entry*, we identified variant patterns, and relevant contexts, including contexts with negation.

7 Results

To highlight the comparison of the three methods, we distinguish between structures that have or have not been mentioned in any ECS tickets prior to Y_i. In the testing data (2006), there are 21,471 structures that have been mentioned in ECS tickets from prior years back through 1996, versus 29,748 that have not been mentioned. Using the baseline classification of events, the proportion of positively labeled instances in the mentioned structures is $\frac{232}{21,471}$, and in the not-mentioned structures it is $\frac{197}{29,748}$. This gives a ratio of 1.6317, which is well above random labeling (a ratio of 1). Using Rules1, the proportion of positively labeled structures that have been mentioned to positively labeled structures that have not been, is 1.6788. For Rules2, the proportion of positively labeled structures in the mentioned set is 1.9602, or almost double the baseline.

Table 3. AUC scores for the three event classification methods

	Train	Test
Baseline	67.63	65.01
Rules1	66.80	63.72
Rules2	68.29	67.55

Table 3 shows the AUC values for the three methods. Rules2 exhibits the least difference between training and test, and has the highest values. Due to many factors, for instance the high skew in the data, and the fact that information derived from ECS tickets primarily affects the top of the ranked list of structures, the changes to the event classification that reduce the noise in labels and features do not lead to a dramatic difference in AUC values. The impact of the improvements shows up in a more qualitative analysis of the ranked lists.

To illustrate the change in quality of the rankings, we present details on two structures that shifted rank, and which exemplify the changes in the ranked list. We will refer to the two structures as D (for demotion) and P (for promotion). The largest shifts in position were demotions, and D illustrates why. Using the baseline classification, structure D was mentioned in eleven precursor tickets, and was the trouble hole five times. In contrast, using our Rules2 event classification, D had only eight precursor tickets, and was the trouble hole twice. It changed in rank from 759/38911 (at 1.95% from the top) in the baseline, to 3823/28997 (13.18%) in Rules1, to 3105/28587 (10.86%) in Rules2. The structure's label does not change, but the ECS features shift in the direction of lower vulnerability.

In contrast, the promotion of structure P does not reflect changes in the feature representation of P; rather, it is a side-effect of the demotions that occurred around it. In the baseline feature representation, P was a trouble hole seven times and mentioned in twelve tickets, versus five times a trouble hole out of nine mentions in Rules1, and back to seven times a trouble hole in Rules2, out of eleven mentions. It changed from rank 69/38911 (0.18%) in the baseline to 205/28997 (0.71%) in Rules1 to 45/28587 in Rules2 (0.16%).

The demotion of structures that are not as serious as they might appear, and the correlated promotion of structures that are genuinely serious, is exactly the type of change we hoped for, and that makes it easier for Con Ed to fold our results into the way they prioritize structure repairs. Small improvements in AUC scores are not relevant to Con Ed unless they can also see changes in the ranked list, such as a higher proportion of obviously problematic structures.

Tables 4-5 highlight differences in the ranked lists. Table 4 shows the Jaccard distance [14] between the top N structures of the Rules2 ranked list versus the baseline's, where we look at various values of N. For two sets, Jaccard is the ratio of the size of set intersection to the size of the set union, thus is closer to 0 when the sets have fewer members in common and is 1 when the two sets are identical. From the AUC values in Table 3, we know that Rules2 is more predictive than Rules1 or the baseline, but the Jaccard values tell us more specifically that the highest ranked structures in the Rules2 ranking are most different up to the top N structures, with only one a quarter to a third of the structures in common,

Table 4. Jaccard distance of Rules 2 from the baseline on the test data for the top N of the ranked lists

N	5	10	15	20	25	100	300	500	1000	2500	5000	10000
Jaccard	0.25	0.33	0.50	0.60	0.56	0.72	0.74	0.75	0.81	0.81	0.86	0.89

Table 5. Average vulnerability on the test data for the top N of the ranked lists

N	Baseline	Rules1	Rules2
5	36.40	35.80	48.30
10	46.40	42.00	48.30
100	46.49	46.07	46.42
500	41.60	41.58	41.34
5000	35.08	35.61	35.11
50000	25.34	25.34	25.34

and that even at N=500, every fourth structure in Rules2 is not included in the baseline ranking (or every fifth structure for N=2500). For ConEd, the top of the list has a different status; all structures are inspected on a five year cycle, but they would like to identify a subset that needs attention first.

As an investigative tool, we had earlier constructed a vulnerability score (a non-statistical measure) for structures, in close consultation with domain experts; the features it uses do not help the learning, but do help explain the ranking results. It ranges from 0 to 100, with scores above 65 being rare. Table 5 shows that for the top N structures in the ranked lists from the learning models, the average vulnerability per structure decreases from about 50 for the top 100 to about 25 for the full list. Rules 2 gives the highest average vulnerability for the top 10 structures, and about the same as the baseline for the rest of the list. Thus Rules2 pushes more highly vulnerable structures to the very top of the list.

8 Discussion and Conclusion

Our modeling task depended heavily on information found in an extremely noisy and disparate set of documents. Our document classification determined both the labels of examples and their feature representations, an interdependence not often found in learning tasks addressed within NLP. Machine learning has become a very powerful tool, but successful learning requires a knowledge transfer from the human to the machine. In NLP, this often takes the form of human annotations on documents, which serve either as the relevant labels for a learning task, or the features. Since both labels and features depended on the document classification, our model is doubly sensitive to it. The experts' annotations helped us propose manually derived classification rules: our human intelligence became the mechanism of knowledge transfer from the experts to the learning. As a result, the model is more predictive, as described in section 7. The AUC, which we are using as a general measure of ranking quality, does not reflect the dramatic improvement at the top of the ranked list. One out of four or five structures in the top 500-2500 of our ranking was not present in the corresponding part of the baseline ranking. Also, a non-predictive but interpretable scoring of vulnerability shows that the top 5 to 10 structures in the Rules2 ranking have a higher average vulnerability score. Thus the new ranking is qualitatively better than the AUC scores indicate on their own.

References

1. Gross, P., Boulanger, A., Arias, M., Waltz, D.L., Long, P.M., Lawson, C., Anderson, R., Koenig, M., Mastrocinque, M., Fairechio, W., Johnson, J.A., Lee, S., Doherty, F., Kressner, A.: Predicting electricity distribution feeder failures using machine learning susceptibility analysis. In: The 18th Conference on Innovative Applications of Artificial Intelligence IAAI 2006, Boston, Massachusetts (2006)
2. Liddy, E.D., Symonenko, S., Rowe, S.: Sublanguage analysis applied to trouble tickets. In: Proceedings of the Florida Artificial Intelligence Research Society Conference, pp. 752–757 (2006)
3. Devaney, M., Ram, A.: Preventing failures by mining maintenance logs with case-based reasoning. In: Proceedings of the 59th Meeting of the Society for Machinery Failure Prevention Technology (MFPT-59) (2005)
4. Hirschman, L., Palmer, M., Dowding, J., Dahl, D., Linebarger, M., Passonneau, R., Land, F., Ball, C., Weir, C.: The PUNDIT natural-language processing system. In: Proceedings of the Annual AI Systems in Government Conference, pp. 234–243 (1989)
5. Oza, N., Castle, J.P., Stutz, J.: Classification of aeronautics system health and safety documents. IEEE Transactions on Systems, Man and Cybernetics, Part C (accepted for publication)
6. Rudin, C., Passonneau, R.J., Radeva, A., Dutta, H., Ierome, S., Isaac, D.: Predicting vulnerability to serious manhole events in manhattan: A preliminary machine learning approach (submitted for publication)
7. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research 4, 933–969 (2003)
8. Rudin, C.: The P-Norm Push: A simple convex ranking algorithm that concentrates at the top of the list. Journal of Machine Learning Research (accepted, 2008)
9. Joachims, T.: A support vector method for multivariate performance measures. In: Proceedings of the Internat'l. Conf. on Machine Learning (ICML), pp. 377–384 (2005)
10. Artstein, R., Poesio, M.: Inter-coder agreement for computational linguistics. computational linguistics (to appear)
11. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20, 37–46 (1960)
12. Krippendorff, K.: Content analysis: An introduction to its methodology. Sage Publications, Beverly Hills (1980)
13. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics 33(1), 159–174 (1977)
14. Jaccard, P.: Nouvelles recherches sur la distribution florale. Bulletin de la Societe Vaudoise des Sciences Naturelles 44, 223–270 (1908)

Unsupervised Classification of Verb Noun Multi-Word Expression Tokens

Mona T. Diab and Madhav Krishna

Columbia University, New York, NY 10115
mdiab@ccls.columbia.edu, madhkrish@gmail.com

Abstract. We address the problem of classifying multiword expression tokens in running text. We focus our study on Verb-Noun Constructions (VNC) that vary in their idiomaticity depending on context. VNC tokens are classified as either idiomatic or literal. Our approach hinges upon the assumption that a literal VNC will have more **in common** with its component words than an idiomatic one. Commonality is measured by contextual overlap. To this end, we set out to explore different contextual variations and different similarity measures. We also identify a new data set OPAQUE that comprises only non-decomposable VNC expressions. Our approach yields state of the art performance with an overall accuracy of 77.56% on a TEST data set and 81.66% on the newly characterized data set OPAQUE.

1 Introduction

A Multi-Word Expression (MWE), for our purposes, can be defined as a multiword unit that refers to a single concept, for example - *kick the bucket*, *spill the beans*, *make a decision*, etc. An MWE typically has an idiosyncratic meaning that is *more* or *different* than the meaning of its component words. An MWE meaning is transparent, i.e. predictable, in as much as the component words in the expression relay the meaning portended by the speaker compositionally. Accordingly, MWEs vary in their degree of meaning compositionality; compositionality is correlated with the level of idiomaticity. An MWE is compositional if the meaning of an MWE as a unit can be predicted from the meaning of its component words such as in *make a decision* meaning *to decide*. If we conceive of idiomaticity as being a continuum, the more idiomatic an expression, the less transparent and the more non-compositional it is. Some MWEs are more predictable than others, for instance, *kick the bucket*, when used idiomatically to mean *to die*, has nothing in common with the literal meaning of either *kick* or *bucket*, however, *make a decision* is very clearly related to *to decide*. Both of these expressions are considered MWEs but have varying degrees of compositionality and predictability.

MWEs are pervasive in natural language, especially in the ever more abundant web based texts and speech genres. Identifying MWEs and understanding their meaning is essential to language understanding, hence they are of crucial importance for any Natural Language Processing (NLP) applications that aim at

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 98–110, 2009.

handling language meaning and use. In fact, the seminal paper [1] refers to this problem as a *key* issue for the development of high-quality NLP applications.

[2] note that the majority of MWEs are verbal expressions, such as light verb constructions (LVC), verb particle constructions (VPC), and verb noun constructions (VNC). To date, most research has addressed the problem of MWE *type* classification for VNC expressions [3, 4, 5, 6, 7, 8], not *token* classification. For example: *he spilt the beans over the kitchen counter* is most likely a literal usage. This is given away by the use of the prepositional phrase *over the kitchen counter*, since it is plausable that beans could have *literally* been spilt on a location such as a kitchen counter. Most previous research would classify *spilt the beans* as idiomatic irrespective of usage. A recent study by [9] of 60 idiom MWE types concluded that almost half of them had clear literal meanings and over 40% of their usages in text were actually literal. Thus, it would be important for an NLP application such as machine translation, for example, when given a new token of an MWE, to be able to determine whether it is used idiomatically or not.

In this paper, we address the problem of MWE classification for verb-noun (VNC) token constructions in running text. We investigate the binary classification of an unseen VNC token expression as being either **Idiomatic** (IDM) or **Literal** (LIT). An IDM expression is certainly an MWE, however, the converse is not necessarily true. We attempt to handle the problem of *sparsity* for the purpose of MWE classification. We explore several vector similarity metrics. We exploit more linguistically oriented feature sets to model the VNC vector space. We evaluate our results against a standard data set from the study by [10]. We achieve state of the art performance in classifying VNC tokens as either literal (F-measure: F_{β_1}=0.69) or idiomatic (F_{β_1}=0.83), corresponding to an overall accuracy of 77.56%. Recognizing the gray zone in such a binary classification set up, another thrust of our work focuses on a new evaluation set we term OPAQUE. The OPAQUE set comprises MWEs that have a clear distinction between their idiomatic senses and their literal ones.

This paper is organized as follows: In Section 2 we describe our understanding of the various classes of MWEs in general. Section 3 is a summary of previous related research. Section 4 describes our approach. In Section 5 we present the details of our experiments. We discuss the results in Section 6. Finally, we conclude the paper with some future directions in Section 7.

2 Multi-Word Expressions

MWEs are typically not productive, though they allow for inflectional variation [1]. They have been conventionalized due to persistent use. MWEs can be classified based on their semantic types as follows:

Idiomatic: This category includes expressions that are semantically non-compositional. For example, these include fixed expressions such as *kingdom come, ad hoc,* and, *ins and outs*. Fixed expressions tend to be more or less frozen in

form. Idiomatic expressions also include non-fixed expressions such as *break new ground, speak of the devil*, and *break the ice*. Non-fixed expressions may undergo inflectional variations and lexical insertions.

Semi-idiomatic: This class includes expressions that seem semantically non-compositional, yet their semantics are more or less transparent. This category consists of Light Verb Constructions (LVC) and Verb Particle Constructions (VPC). An example of an LVC is *make a living*. The verb *make* would *prefer* a physical entity as an argument [11]. Examples of VPCs are *write-up, call-up* and *phone-up*. The particle *up* is an aspectual modifier of the verb rather than a preposition.

Non-Idiomatic: This category includes expressions that are semantically compositional: Compound nominals such as *prime minister*, proper nouns such as *New York Yankees*, and collocations such as *machine translation*. These expressions are *statistically idiosyncratic*. For instance, *traffic light* is the most likely lexicalization of the concept and would occur more often in text than, say, *traffic regulator* or *vehicle light*.

3 Previous Related Work

Several researchers have addressed the problem of MWE classification [5, 12, 13], however the most similar work to ours is the research by [10] and [7].

Cook *et al.* [10] develop an unsupervised technique that classifies a token instance of a VNC expression as idiomatic if the similarity between its context vector and that of its idiomatic usages is higher than the similarity between its context vector and that of its literal usages. They define the vector dimensions in terms of the co-occurrence frequencies of 1000 most frequent content bearing words (nouns, verbs, adjectives, adverbs and determiners) in the corpus. A context vector for a VNC expression is defined in terms of the words in the sentence in which it occurs. They employ the cosine measure to estimate similarity between contextual vectors. They assume that every instance of an expression occurring in a certain *canonical* syntactic form is idiomatic, otherwise it is literal. This assumption holds for many cases of idiomatic usage since many of them are conventionalized, however in cases such as *spilt the beans on the counter top*, the expression would be misclassified as idiomatic since it does occur in a canonical form though the meaning in this case is literal. They estimate the context vectors of literal usages in two ways: by either using those for the 'non-canonical' forms of the expression, or by adding the co-occurrence vectors of the component words. Their method achieves an accuracy of 52.7% on a data set containing expression tokens used mostly in their literal sense, whereas it yields an accuracy of 82.3% on a data set in which most usages are idiomatic. Further, they report that a classifier that predicts the idiomatic label if an expression (token) occurs in a canonical form achieves an accuracy of 53.4% on the former data set (where the majority of the MWEs occur in their literal sense) and 84.7% on the

latter data set (where the majority of the MWE instances are idiomatic). This indicates that these 'canonical' forms can still be used literally. They report an overall system performance accuracy of 72.4%. We note that the use of accuracy as a measure for this work is not the most appropriate since accuracy is a measure of error rather than correctness, hence we report F-measure in addition to accuracy (to be able to compare with previous work). Their work is similar to this paper in that they explore the VNC expressions at the token level, even though they notably use type characteristics when assigning a class label to a token expression.

Fazly and Stevenson [7] correlate compositionality with idiomaticity. They measure compositionality as a combination of two similarity values: firstly, the similarity (cosine similarity) between the context of a VNC and the contexts of its constituent words; secondly, the similarity between an expression's context and that of a verb that is morphologically related to the noun in the expression, for instance, *decide* for *make a decision*. Context $context(t)$ of an expression or a word, t, is defined as a vector of the frequencies of nouns co-occurring with t within a window of ± 5 words. The resulting compositionality measure yields an $F_{\beta=1}=0.51$ on identifying literal expressions and $F_{\beta=1}=0.42$ on identifying idiomatic expressions. However, their results are not comparable to ours since it is type-based study.

4 Our Approach

Recognizing the significance of contextual information in MWE token classification, we explore the space of contextual modeling for the task of classifying the token instances of VNC expressions into literal versus idiomatic expressions. Inspired by works of [7, 12], our approach is to compare the context vector of a VNC, as an MWE, with the composed vector of the verb and noun (V-N) component units of the VNC when they occur in isolation of each other (i.e., not as a VNC). For example, in the case of the MWE *kick the bucket*, we compare the contexts of the instances of the VNC *kick the bucket* against the combined contexts for the verb (V) *kick*, independent of the noun *bucket*, and the contexts for the noun (N) *bucket*, independent of the verb *kick*. The intuition is that if there is a high similarity between the VNC and the combined V and N (namely, the V-N vector) contexts, then the VNC token is compositional, hence a literal instance of the MWE, otherwise the VNC token is idiomatic.

Previous work, [7], restricted context to within the boundaries of the sentences in which the tokens of interest occurred. We take a cue from that work but define '$context(t)$' as a vector with dimensions as **all** word types occurring in the same sentence as t, where t is a verb type corresponding to the V in the VNC, noun type corresponding to N in the VNC, or VNC expression instance. Moreover our definition of context includes all nouns, verbs, adjectives and adverbs occurring in the same paragraph as t. This broader notion of context should help reduce sparseness effects, simply by enriching the vector with more contextual information. Further, we realize the importance of some closed class

words occurring in the vicinity of t. [10] report the importance of determiners in identifying idiomaticity. Prepositions too should be informative of idiomaticity (or literal usage) as illustrated above in *spill the beans over the kitchen counter*. Hence, determiners and prepositions occurring in the same sentence as t are also included in its context. The composed V-N contextual vector combines the co-occurrence of the verb type (aggregation of all the verb token instances in the whole corpus) as well as the noun type with this predefined set of dimensions. The VNC contextual vector is that for a specific instance of a VNC expression. Our objective is to find the best experimental settings that could yield the most accurate classification of VNC expression tokens. To that end, we explore the space of possible parameter variation. We experiment with five different parameter settings: the extent of context considered to model the vectors; the context vector dimensions for both V-N and VNC; the context content type; the vector similarity measure; and the method for combining the vectors for the verb type and the noun type to create the V-N composed contextual vector. Throughout the description below, a token (T) of interest could be a VNC, a (N)oun or a (V)erb. These parameters are detailed as follows:

Context-Extent: The definition of context for T is restricted to: $Context_{Broad}$ comprises all the open class or *content* words (nouns, verbs, adjectives and adverbs) as well as the determiners and prepositions in the sentence containing T, in addition to the content words from the paragraph surrounding T. $Context_{Narrow}$ comprises all the open class words as well as the prepositions and determiners for the same sentence as T.

Dimension: This parameter is varied in three ways: $Dimension_{NoThresh}$ includes all the words that co-occur with T in the specified Context-Extent. $Dimension_{Freq}$ sets a threshold on the co-occurrence frequency for the words to include in the dimensions thereby reducing the dimensionality of the vectors. $Dimension_{Ratio}$ is inspired by the utility of the *tf-idf* measure in information retrieval, we devise a threshold scheme that takes into consideration the salience of the word in context as a function of its relative frequency. Hence the raw frequencies of the words in context are converted to a ratio of two probabilities as per equation (1).

$$ratio = \frac{p(word|context)}{p(word)} = \frac{\frac{freq(word\ in\ context)}{freq(context)}}{\frac{freq(word\ in\ corpus)}{N}} \quad (1)$$

In equation (1), N is the number of words (tokens) in the corpus and $freq(context)$ is the number of *context*s for a specific T occurs. The numerator of the ratio is the probability that the word occurs in a particular context. The denominator is the probability of occurrence of the word in the corpus. Here, more weight is placed on words that are frequent in a certain context but rarer in the entire corpus. In case of the V and N contexts, a suitable threshold, which is independent of data size, is determined on this ratio in order to prune context words.

The latter two pruning techniques, $Dimension_{Freq}$ and $Dimension_{Ratio}$, are not performed for a VNC token's context, hence, all the words in the VNC token's contextual window are included. These thresholding methods are only applied to V-N vectors.

Context-Content: This parameter had two settings: words as they occur in the corpus, $Context-Content_{Words}$; or some of the words are collapsed into named entities, $Context-Content_{Words+NER}$. $Context-Content_{Words+NER}$ attempts to perform dimensionality reduction and sparsity reduction by collapsing named entities in the context of the VNC as well as those in the context of the V-N vectors. The intuition is that if we reduce the dimensions in semantically salient ways we will not adversely affect performance.

We employ BBN's *IdentiFinder* Named Entity Recognition (NER) System[1]. The NER system reduces all proper names, months, days, dates and times to NE tags. NER tagging is done on the corpus before the context vectors are extracted. For our purposes, it is not important that ***John*** *kicked the bucket on* ***Friday*** *in* ***New York City*** – neither the specific actor of the action, nor the place where is occurs is of relevance. The sentence ***PERSON*** *kicked the bucket on* ***DAY*** *in* ***PLACE*** conveys the same amount of information.

IdentiFinder identifies 24 NE types. We deem 5 of these inaccurate based on our observation, and exclude them. We retain 19 NE types: *Animal, Contact Information, Disease, Event, Facility, Game, Language, Location (merged with Geo-political Entity), Nationality, Organization, Person, Product, Date, Time, Quantity, Cardinal, Money, Ordinal* and *Percentage*. The written-text portion of the BNC contains 6.4M named entities in 5M sentences (at least one NE per sentence). The average number of words per NE is 2.56, the average number of words per sentence is 18.36. Thus, we estimate that by using NER, we reduce vector dimensionality by at least 14% without introducing the negative effects of sparsity.

V-N Combination: In order to create a single vector from the units of a VNC expression, we need to combine the vectors pertaining to the verb type (V) and the noun type (N). After combining the word types in the vector dimensions, we need to handle their co-occurrence frequency values. Hence we have two methods: *addition* where we simply add the frequencies in the cases of the shared dimensions which amounts to a union where the co-occurrence frequencies are added; or *multiplication* which amounts to an intersection of the vector dimensions where the co-occurrence frequencies are multiplied, hence giving more weight to the shared dimensions than in the *addition* case. In a study by [14] on a sentence similarity task, a multiplicative combination model performs better than the additive one.

Similarity Measures: We experiment with several standard similarity measures: Cosine Similarity, Overlap similarity, Dice Coefficient and Jaccard Index, as

[1] http://www.bbn.com/technology/identifinder

defined in [15]. A context vector is converted to a set by using the dimensions of the vector as members of the set.

5 Experiments and Results

5.1 Data

We use the British National Corpus (BNC),[2] which contains 100M words, because it draws its text from a wide variety of domains and the existing gold standard data sets are derived from it. The BNC contains multiple genres including written text and transcribed speech. We only experiment with the written-text portion. We syntactically parse the corpus with the *Minipar*[3] parser in order to identify all VNC expression tokens in the corpus. We exploit the lemmatized version of the text in order to reduce dimensionality and sparseness.

The standard data used in [10] (henceforth CFS07) is derived from a set comprising 2920 unique VNC-Token expressions drawn from the whole BNC. In this set, VNC token expressions are manually annotated as *idiomatic*, *literal* or *unknown*. The annotators were presented with the sentence that contained the VNC token only. The *unknown* class was used only in cases when the context did not seem enough to discern idiomaticity. The 2920 VNC expressions correspond to 53 VNC expression types, 28 of which have $\sim$ 60% of their token instances labeled idiomatic while $\sim$ 40% are labeled literal. The remaining 25 VNC expression types (corresponding to 1309 VNC tokens) are skewed in their distribution, almost all instances of a given expression are either idiomatic or literal.

For our purposes, we discard 127 of the 2920 token gold standard data set either because they are derived from the speech transcription portion of the BNC, or because *Minipar* could not find them. Similar to the CFS07 set, we exclude expressions labeled *unknown* by the annotators or pertaining to the skewed data set. Therefore, our resulting data set comprises 1125 VNC token expressions (CFS07 has 1180). We then split them into a development (DEV) set and a test (TEST) set. The DEV set comprises 564 token expressions corresponding to 346 idiomatic (IDM) expressions and 218 literal (LIT) ones (CFS07 dev has 573). The TEST set comprises 561 token expressions corresponding to 356 IDM expression tokens and 205 LIT ones (CFS07 test has 607). There is a complete overlap in types between our DEV and CFS07's dev set and our TEST and CFS07's test set. They each comprise 14 VNC type expressions with no overlap in type between the TEST and DEV sets. That means that the techniques developed and tested to address MWE problem in both our work and the work of CFS07 are robust and generalizable since no tuning of parameters is done on any VNC types that are present in the TEST data. We divide the tokens between the DEV and TEST maintaining the same proportions recommended in CFS07. Our DEV set has 61.5% while CFS07 has 60.9% idiomatic expressions. Our TEST set has 63.7% idiomatic expressions compared to 63.3% reported in CFS07. Even though

[2] http://www.natcorp.ox.ac.uk/

[3] http://www.cs.ualberta.ca/ lindek/minipar.htm

Table 1. VNC Expression types in OPAQUE data set

move goalpost, pull weight, pull leg, make hay, hit roof, hold horse, blow smoke, kick heel, get sack, give sack, blow whistle, blow trumpet, get drift, get wind

the number of instances is less in our TEST set, we believe that the results are generally comparable with those obtained by CFS07.

Following the intuition that idiomaticity is not a binary property, we create a new test data set, OPAQUE. The OPAQUE data set comprises those expressions that are at the high idiomaticity extreme of the spectrum. An opaque expression is one whose *idiomatic* meaning is highly non-compositional; [1] have called such expressions 'non-decomposable'. For instance, the expression *kick the bucket* is non-decomposable as its idiomatic meaning is completely unrelated to its literal meaning. On the other hand, *make a face*, though idiomatic, is decomposable and transparent to a certain degree.

To this end, we create a set of OPAQUE expressions from the VNC gold standard data set identified in work by [9]. The OPAQUE set comprises 14 VNC expression types listed in Table 1. The set was created in the following manner. All 53 VNC expression types in the gold set are judged by two annotators independently according to two diagnostics: if the verb and noun in the VNC expression are not indicative of their idiomatic meaning; if the idiomatic and literal meaning of the expression are completely distinct.[4] The resulting set of 14 expressions is the intersection between the two annotators, i.e. 100% agreement between the two annotators. Five of the OPAQUE expressions overlap with the skewed set in the gold standard, i.e. they are not in either our DEV or TEST sets: *hold horse, blow smoke, get drift, give sack*, and *move goalpost*.

The opaque set comprises 428 tokens (224 literal and 204 idiomatic) corresponding to 9 VNC types from the DEV and TEST sets, in addition to the 5 VNC types added from the the skewed data set. In our evaluation, we exclude the opaque expressions that come from the DEV set and the skewed data set, leaving only 282 expressions. Accordingly, the final OPAQUE set includes 142 idiomatic and 140 literal tokens. In order to maintain the 61-64% ratio (idiomatic to total number of tokens) as in CFS07, we employ a *bootstrapping* scheme. With the number of idiomatic tokens fixed at 142, 83 literal tokens are selected at random from the set of 140 literal expressions to form a data set containing 225 tokens with the ratio of IDM expressions to total number of tokens being 63.1%. This random selection process is repeated 1000 times. The results reported below are obtained after averaging over a 1000 trials.

5.2 Experimental Set-Up

We vary four of the experimental parameters: Context-Extent, Context-Content, Dimension and V-N compositionality, to create 9 experimental conditions. In the

[4] All meanings are looked up in the Cambridge Dictionaries Online, http://dictionary.cambridge.org/

following experiments, the thresholds (for $Dimension_{Freq}$ and $Dimension_{Ratio}$) are tuned on all the similarity measures collectively. It is observed that the performance of all the measures improved/worsened together, illustrating the same trends in performance, over the various settings of the thresholds evaluated on the DEV data set. Once the thresholds are tuned using the DEV set, they are applied to the TEST and OPAQUE data sets with no further tuning. Optimal thresholds for the similarity measures are also tuned on DEV. We note that different experimental conditions warranted different frequency and ratio thresholds. The experimental conditions are detailed as follows:

nT-A-W-N: The Dimension parameter is set to $Dimension_{NoThresh}$ (nT) and the V-N compositionality is addition (A). Context-Content is set to $Context - Content_{Words}$ (W) and Context-Extent is set to $Context_{Narrow}$ (N).
nT-M-W-N: The Dimension parameter is set to $Dimension_{NoThresh}$ (nT), and the V-N compositionality used is multiplication (M). Context-Content is set to $Context - Content_{Words}$ (W) and Context-Extent is set to $Context_{Narrow}$ (N).
F-M-W-N: $Dimension_{Freq}$ (F) is set to a threshold on the raw co-occurrence frequency of a word with the V-N composed vector. The optimal threshold is determined empirically on the DEV set to be 175. Multiplicative compositionality (M) is used, and Context-Content is set to $Context - Content_{Words}$ (W). Context-Extent is set to $Context_{Narrow}$ (N).
R-M-W-N: The Dimension parameter is set to the Ratio $Dimension_{Ratio}$ (R). The ratio is from Equation (11) is used and its optimal threshold value is 27 by tuning on the DEV set. Multiplicative compositionality mode for the V-N vectors (M), and the Context-Content is set to $Context - Content_{Words}$ (W). Context-Extent is set to $Context_{Narrow}$ (N).
F-A-W-N: The Dimension parameter $Dimension_{Freq}$ (F) is set to a threshold on the raw co-occurrence frequency of a word with the V-N composed vector. The optimal threshold is determined empirically on the DEV set to be 200. The V-N compositionality mode used is addition (A), and the Context-Content is set to $Context - Content_{Words}$ (W). Context-Extent is set to $Context_{Narrow}$ (N).
R-A-W-N: The Dimension parameter is set to the Ratio $Dimension_{Ratio}$ (R). An optimal threshold value for the ratio is determined as 75 based empirically on the DEV set. The V-N compositionality mode is addition (A), and the Context-Content is set to $Context - Content_{Words}$ (W). Again, Context-Extent is set to $Context_{Narrow}$ (N).
R-A-W-B: The parameter settings are the same as R-A-W-N except for Context-Extent which is set to $Context_{Broad}$ (B). The optimal threshold for the ratio is 265.
R-M-W-B: The parameter settings are the same as R-M-W-N except for Context-Extent which is set to $Context_{Broad}$ (B). The optimal threshold for the ratio is 150.
R-A-NE-B: Similar to the R-A-W-B experimental set up except that the Context-Content is set to $Context - Content_{Words+NER}$ (NE). The optimal value for the threshold on the ratio values is 275.

Table 2. Evaluation on of different experimental conditions on DEV

Experiment	Dice Coefficient			Jaccard Index			Overlap			Cosine		
	F-measure		Accuracy	F-measure		Accuracy	F-measure		Accuracy	F-measure		Accuracy
	IDM	LIT		IDM	LIT		IDM	LIT		IDM	LIT	
nT-A-W-N	0.45	0.44	44.39%	0.47	0.43	44.92%	**0.50**	**0.56**	**53.30%**	0.49	0.42	45.63%
nT-M-W-N	0.48	0.46	46.88%	0.48	0.46	46.88%	**0.58**	**0.57**	**57.78%**	0.46	0.47	46.52%
F-M-W-N	0.48	0.48	47.77%	0.48	0.48	47.59%	**0.58**	**0.57**	**57.75%**	0.49	0.49	49.19%
R-M-W-N	0.63	0.60	61.50%	0.63	0.60	61.50%	**0.72**	**0.63**	**68.45%**	0.65	0.61	63.10%
F-A-W-N	0.48	0.48	47.95%	0.48	0.48	47.95%	**0.54**	**0.53**	**53.65%**	0.52	0.52	51.69%
R-A-W-N	0.66	0.63	64.17%	0.66	0.63	64.17%	**0.78**	**0.68**	**73.80%**	0.76	0.64	71.12%
R-M-W-B	0.50	0.61	56.33%	0.62	0.61	61.50%	**0.84**	**0.73**	**79.68%**	0.66	0.66	65.78%
R-A-W-B	0.70	0.63	67.20%	0.70	0.63	67.20%	**0.84**	**0.76**	**81.10%**	0.77	0.69	73.62%
R-A-NE-B	0.72	0.66	69.05%	0.72	0.66	69.05%	**0.85**	**0.75**	**81.22%**	0.78	0.71	75.00%

5.3 Results

We use $F_{\beta=1}$ (F-measure) which is the harmonic mean between precision and recall, as well as accuracy to report the results. We report the results separately for the two classes IDM and LIT on all the data sets, DEV, TEST and OPAQUE. As mentioned above, throughout the experiments, all the thresholds are tuned on the DEV set. The tables below illustrate the results obtained using all four similarity measures.

6 Discussion

As shown in Table 3, we obtain a classification accuracy of 77.56% (R-A-W-N) on TEST using the Overlap similarity measure, with $F_{\beta=1}$ values for the IDM and LIT classes being 0.83 and 0.69, respectively. These results are comparable to state-of-the-art results obtained by CFS07 who report an overall system accuracy of 72.4% on their test set. Hence, we improve over state-of-the-art results by 5.16% absolute. Even if we compare the results yielded by the Cosine measure (as this was the measure used in CFS07, we note an increase of 4.22% absolute improvement at an accuracy of 76.66% for our classification approach. It is worth noting that the differences among all possible pairs of mean F-measure and accuracy values (within each experiment), except for the cases when the Dice and Jaccard measures perform equivalently, are found to be statistically significant.

The highest accuracy figures across all experimental conditions are obtained using the overlap similarity measure across all three data sets. It is also worth noting that for each similarity measure, the highest accuracy values are associated with the highest F-measure performance on IDM and LIT classification. Moreover, in those conditions, our system is always yielding better performance of identifying IDM expressions than literal expressions with significantly large difference in performance. Contrary to previous work, we note that the Cosine similarity is outperformed by the Overlap measure.

Comparing the different experimental conditions, results suggest that $Dimension_{Ratio}$ outperforms $Dimension_{Freq}$ and $Dimension_{NoThresh}$ within all data sets. We recognize that in the $Dimension_{Ratio}$, we vary the ratio threshold value depending on experimental condition which might render the results

Table 3. Evaluation of different experimental conditions on TEST

Experiment	Dice Coefficient			Jaccard Index			Overlap			Cosine		
	F-measure		Accuracy	F-measure		Accuracy	F-measure		Accuracy	F-measure		Accuracy
	IDM	LIT		IDM	LIT		IDM	LIT		IDM	LIT	
nT-A-W-N	0.58	0.48	53.50%	0.62	0.49	56.37%	0.43	0.50	46.32%	**0.63**	**0.48**	**56.37%**
nT-M-W-N	0.58	0.46	52.60%	0.53	0.48	50.45%	0.53	0.50	51.71%	**0.55**	**0.51**	**52.78%**
F-M-W-N	0.56	0.48	52.06%	0.56	0.48	52.06%	0.50	0.44	47.04%	**0.60**	**0.51**	**47.04%**
R-M-W-N	0.64	0.61	62.48%	0.64	0.61	62.48%	**0.72**	**0.61**	**68.04%**	0.68	0.62	65.35%
F-A-W-N	**0.57**	**0.46**	**52.24%**	**0.57**	**0.46**	**52.24%**	0.44	0.39	41.29%	0.57	0.45	51.53%
R-A-W-N	0.73	0.65	69.30%	0.73	0.65	69.30%	**0.83**	**0.69**	**77.56%**	0.82	0.66	76.66%
R-M-W-B	0.51	0.60	55.54%	0.64	0.58	61.07%	**0.80**	**0.64**	**74.46%**	0.66	0.60	63.04%
R-A-W-B	0.69	0.57	64.11%	0.69	0.57	64.11%	**0.82**	**0.66**	**76.07%**	0.76	0.61	70.00%
R-A-NE-B	0.70	0.58	64.93%	0.70	0.58	64.93%	**0.83**	**0.65**	**76.62%**	0.76	0.62	70.86%

Table 4. Evaluation of different experimental conditions on OPAQUE

Experiment	Dice Coefficient			Jaccard Index			Overlap			Cosine		
	F-measure		Accuracy	F-measure		Accuracy	F-measure		Accuracy	F-measure		Accuracy
	IDM	LIT		IDM	LIT		IDM	LIT		IDM	LIT	
nT-A-W-N	0.51	0.53	52.09%	0.53	0.52	52.40%	0.48	0.48	47.96%	**0.55**	**0.52**	**53.45%**
nT-M-W-N	0.29	0.46	38.87%	0.29	0.48	39.70%	**0.73**	**0.56**	**66.45%**	0.32	0.50	42.57%
F-M-W-N	0.34	0.50	42.71%	0.34	0.50	42.71%	**0.66**	**0.40**	**56.88%**	0.44	0.50	47.47%
R-M-W-N	0.69	0.65	67.22%	0.69	0.65	67.22%	**0.77**	**0.68**	**72.85%**	0.69	0.65	66.97%
F-A-W-N	0.41	0.37	38.65%	0.41	0.37	38.65%	0.46	0.23	36.15%	**0.44**	**0.38**	**41.18%**
R-A-W-N	0.71	0.66	68.42%	0.71	0.66	68.42%	**0.85**	**0.75**	**81.10%**	0.80	0.72	76.76%
R-M-W-B	0.24	0.54	42.69%	0.56	0.54	54.95%	**0.79**	**0.70**	**75.00%**	0.46	0.58	52.72%
R-A-W-B	0.58	0.60	58.82%	0.58	0.60	58.82%	**0.83**	**0.76**	**80.23%**	0.69	0.66	67.83%
R-A-NE-B	0.61	0.61	61.06%	0.61	0.61	61.06%	**0.86**	**0.76**	**81.66%**	0.70	0.65	67.71%

not directly comparable across the different R conditions in the same data set. We would argue that the results are directly comparable however, since *Ratio* as characterized by our definition is a relative threshold that will have to depend on the other parameters, for example, using *addition* warrants a very different ratio threshold from using *multiplication*, therefore it is more condition dependent. Hence we can grossly compare across conditions that apply dimensionality reduction using *some* ratio threshold. However, we emphasize that the results are directly comparable across data sets with the same condition.

Accordingly, for vector compositionality for the V-N vector, *addition* clearly outperforms *multiplication* for the task of MWE classification. This indicates that union is better than intersection for combining the V and N vectors for this task. *multiplication* seems to increase vector sparsity. We note that $Context_{Narrow}$ does better than $Context_{Broad}$ on the TEST and OPAQUE data sets, though this is clearly the opposite in the DEV data set where the $Context_{Broad}$ conditions outperform their $Context_{Narrow}$ counterparts, R-A-W-B yields better results than R-A-W-N. This may indicate that the parameter tuned for the $Context_{Broad}$ conditions on DEV is overfitted for the DEV data set. Comparing results (accuracy, and F-measure on IDM and LIT) using $Context - Content_{Words}$ versus $Context - Content_{Words+NER}$, in R-A-W-B against R-A-NE-B, we note that in all data sets, $Context - Content_{Words+NER}$ is closely comparable or even outperforms using $Context - Content_{Words}$ across all similarity measures. This strongly suggests that dimensionality reduction using NER has a significant positive impact on MWE classification.

The best performing condition for the DEV and OPAQUE data is R-A-NE-B across all similarity measures. However, this does not hold for the TEST data set. For the latter data set, we note that R-A-W-N yields the best performance for all the measures followed closely by R-A-NE-B. These results suggest that R-A-W-N and R-A-NE-B are the best experimental conditions for classification. R-A-W-N is not the 2nd best condition for all similarity measures, in the case of DEV. The variation in experimental results between the DEV, TEST and OPAQUE sets may be attributed to the fact that the tuning parameters are tuned on the DEV data and that there are no shared MWE types between the DEV and OPAQUE and TEST data sets.

The highest yielded results are obtained on the OPAQUE data set, at an accuracy of 81.66%, an IDM F-measure classification of 86%, and a LIT F-measure of 76% in the R-A-NE-B experimental condition using overlap similarity. These results are even higher than those obtained in the best performing condition on the DEV set. These results are significantly higher than those obtained on the best condition of the TEST set. This suggests that the OPAQUE set indeed has a naturally clear distinction between idiomatic and literal usages of MWE expressions.

7 Conclusion

In this study, we explored a set of features that contribute to VNC token expression binary classification. We applied dimensionality reduction heuristics inspired by information retrieval (*tf-idf* like ratio measure) and linguistics (named-entity recogniiton). These contributions improve significantly over experimental conditions that do not manipulate context and dimensions. Our system achieves state-of-the-art performance on a set that is very close to a standard data set. Different from previous studies, we classify VNC token expressions in context. We include function words in modeling the VNC token contexts as well as using the whole paragraph in which it occurs as context. We also designate a new data set, OPAQUE, that reflects the more non-decompositional aspect of VNC MWEs' idiomaticity. The results suggest that our approach is able to reliably capture the discriminatory features for MWE classification. As expected the results on the OPAQUE set outperform the results yielded on the TEST set due to the clear separability of the idiomatic senses from the literal ones for the VNC tokens. Further, these results reaffirm the notion that idiomaticity is not a discrete binary property.

References

1. Sag, I.A., Baldwin, T., Bond, F., Copestake, A.A., Flickinger, D.: Multiword expressions: A pain in the neck for NLP. In: Gelbukh, A. (ed.) CICLing 2002. LNCS, vol. 2276, pp. 1–15. Springer, Heidelberg (2002)
2. Villavicencio, A., Copestake, A.: On the nature of idioms. In: LinGO Working Paper No. 2002-2004 (2002)

3. Melamed, D.I.: Automatic discovery of non-compositional compounds in parallel data. In: Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing (EMNLP 1997), Providence, RI, USA, pp. 97–108 (1997)
4. Lin, D.: Automatic identification of noncompositional phrases. In: Proceedings of ACL 1999, Univeristy of Maryland, College Park, Maryland, USA, pp. 317–324 (1999)
5. Baldwin, T., Bannard, C., Tanaka, T., Widdows, D.: An empirical model of multiword expression decomposability. In: Proceedings of the ACL 2003 Workshop on Multiword Expressions, Morristown, NJ, USA, pp. 89–96 (2003)
6. na Villada Moirón, B., Tiedemann, J.: Identifying idiomatic expressions using automatic word-alignment. In: Proceedings of the EACL 2006 Workshop on Multiword Expressions in a Multilingual Context, Morristown, NJ, USA, pp. 33–40 (2006)
7. Fazly, A., Stevenson, S.: Distinguishing subtypes of multiword expressions using linguistically-motivated statistical measures. In: Proceedings of the Workshop on A Broader Perspective on Multiword Expressions, Prague, Czech Republic, Association for Computational Linguistics, pp. 9–16 (2007)
8. Van de Cruys, T., Villada Moirón, B.n.: Semantics-based multiword expression extraction. In: Proceedings of the Workshop on A Broader Perspective on Multiword Expressions, Prague, Czech Republic, Association for Computational Linguistics, pp. 25–32 (2007)
9. Cook, P., Fazly, A., Stevenson, S.: The VNC-Tokens Dataset. In: Proceedings of the LREC Workshop on Towards a Shared Task for Multiword Expressions (MWE 2008), Marrakech, Morocco (2008)
10. Cook, P., Fazly, A., Stevenson, S.: Pulling their weight: Exploiting syntactic forms for the automatic identification of idiomatic expressions in context. In: Proceedings of the Workshop on A Broader Perspective on Multiword Expressions, Prague, Czech Republic, Association for Computational Linguistics, pp. 41–48 (2007)
11. Resnik, P.: Selectional preference and sense disambiguation. In: Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?, Washington, DC, USA (1997)
12. Katz, G., Giesbrecht, E.: Automatic identification of non-compositional multi-word expressions using latent semantic analysis. In: Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties, Sydney, Australia, Association for Computational Linguistics, pp. 12–19 (2006)
13. Schone, P., Juraksfy, D.: Is knowledge-free induction of multiword unit dictionary headwords a solved problem? In: Proceedings of Empirical Methods in Natural Language Processing, Pittsburg, PA, USA, pp. 100–108 (2001)
14. Mitchell, J., Lapata, M.: Vector-based models of semantic composition. In: Proceedings of ACL 2008: HLT, Columbus, Ohio, Association for Computational Linguistics, pp. 236–244 (2008)
15. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)

Semantic Mapping for Related Term Identification

Rafael E. Banchs

Barcelona Media Innovation Centre,
Av. Diagonal 177, Planta 9, 08018 Barcelona, Spain
rafael.banchs@barcelonamedia.org

Abstract. In this work, we explore the combined use of latent semantic analysis (LSA) and multidimensional scaling (MDS) for identifying related concepts and terms. We approach the problem of related term identification by constructing low-dimensional embeddings where related terms are clustered together, and such clusters are spatially arranged according to the semantic relationships among the terms they include. In this work, we demonstrate the proposed methodology for a specific part-of-speech (verbs) of the Spanish language, by using dictionary-based definitions. We also comment on the future use of this experimental framework in the context of other natural language processing tasks such as opinion mining, topic detection and automatic summarization.

Keywords: Vector Space Model, Latent Semantic Analysis, Multidimensional Scaling, Related Term Identification.

1 Introduction

The vector-space model has been extensively used in information retrieval and other text-mining applications. Within this particular context, some prominent techniques such as latent semantic analysis [4] and probabilistic latent semantic analysis [6] have been developed in order to obtain more efficient vector-space representations in terms of space dimensionality reduction and feature-based structural characterization. In this work, we attempt to combine latent semantic analysis along with multidimensional scaling with the objective of further reducing dimensional complexity while preserving structural characterization.

The methodology being proposed in this article is not intended to constitute a new solution for any specific text analysis problem, but rather to provide an experimental framework for exploring and evaluating text processing alternatives which could benefit from the tractability of very-low-dimensional data representations. In this work, the proposed methodology is presented and illustrated with the problem of related term identification [1]. An alternative approach for concept association representations by means VOS projections has been already proposed in [7].

The paper is structured as follows. First, in section 2, a brief overview of latent semantic indexing and multidimensional scaling is presented. Then, in section 3, the proposed methodology is described in detail. In section 4, the experimental work is presented and discussed. Finally, section 5 presents some relevant conclusions and recommendations for future work, including some brief comments about the possible

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 111–124, 2009.

application of the proposed methodology in other well known natural language processing tasks.

2 Semantics and Space Reduction

In this section we will present a very brief overview of the space reduction techniques that are involved in the proposed procedure. First, we will discuss latent semantic analysis (LSA), followed by multidimensional scaling (MDS). For a more complete and detailed description on each technique, the reader can refer to [4] and [3], respectively.

2.1 Latent Semantic Indexing

A typical term-document matrix, used for vector-space modeling, is generally very sparse and noisy. LSA, first proposed by [4], constitutes a linear space reduction technique which operates under the same principle as principal component analysis. Both space reduction methods rely on singular value decomposition (SVD) in order to project a given data set into an ortho-normal set of basis vectors. Such representation is optimal in the sense that it finds the linear transformation that maximizes the data variance (information content) along the new ortho-normal set of basis vectors. Then, efficient space reduction is a natural consequence of (SVD) analysis because the less informative components of the new data representation can be dropped while introducing a minimal loss of information.

The mathematical formulation of LSA directly follows SVD, so, any given term-document matrix $\boldsymbol{M}$ can be decomposed as:

$$\boldsymbol{M} = \boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V}^{T} \tag{1}$$

where $\boldsymbol{U}$ is a unitary matrix referred to as the *output* basis vectors, $\boldsymbol{V}$ is a unitary matrix referred to as the *input* basis vectors, and $\boldsymbol{\Lambda}$ is a diagonal matrix containing the singular values.

In natural language processing, LSA has been extensively used for obtaining reduced-space representations of term-document matrices which provide a less sparse and noise reduced representation of the given data set. When dimensionality reduction is applied to the terms or words in the term-document matrix, each reduce-space dimension represents a new feature (or concept) which actually constitutes a linear combination of the original terms. These features are supposed to capture the semantic relationships among the terms, which are implicitly defined by means of their co-occurrences within the document collection.

2.2 Multidimensional Scaling

MDS refers to a family of methods for data structure visualization and/or representation. Given a set of observed dissimilarities, similarities or ordinal relations among a group of objects, the main objective of MDS is to find an embedding (commonly, a set of Euclidean coordinates) for the objects such that the set of observed relations is preserved as much as possible [3].

Commonly, two categories of MDS are generally distinguished: metric MDS, for which distances among the points in the embedding should match as much as possible original object dissimilarities; and non-metric MDS, for which distances among the points in the embedding are only required to match a monotonic transformation of the original object dissimilarities (actually, this second case constitutes a relaxation of the first one).

In practice, MDS is implemented via an optimization procedure, for which an objective function referred to as the *stress* function should be minimized. In this way, distances among all pair of points in the embedding are adjusted such that the *stress* function is minimized. The general form of this function is as follows:

$$stress = sqrt\{ 1/k \; \Sigma\Sigma(f(s_{ij}) - d_{ij})^2 \} \tag{2}$$

where s_{ij} represents the input data dissimilarities, d_{ij} are the distances among the points in the embedding, $f(\cdot)$ is a monotonic transformation (which is only used for non-metric MDS), and k is a scaling factor.

Although MDS is not, by definition, a space reduction technique, it can be certainly used for space reduction purposes. Indeed, a dissimilarity matrix can be always constructed for a given set of vectors in a high-dimensional space by means of any distance metric. Then, MDS can be used for constructing an embedding for such set of vectors in some Euclidean space of lower dimensionality.

Notice that different from LSA, space reduction by means of MDS constitutes a non-linear projection technique. Additionally, the minimum achievable value for the *stress* function, at a given dimension, provides an indication of how much of the original data structure has been lost due to dimensionality reduction.

3 The Proposed Methodology

In this section we will describe in detail the proposed methodology, which is intended to combine the advantages of LSA and MDS for constructing low-dimensional representations of terms and concepts that are able to provide both, an underlying semantic structure, and a more efficient and human-readable organization of the data.

One of the main disadvantages of LSA is the clear difficulty in interpreting the meaning of obtained reduced-space dimensions, with the additional drawback of the still high-dimensionality of the resulting reduced space. Several authors have reported that appropriate performance of LSA requires the use of reduced spaces around 400 and 600 dimensions. Such a space dimensionality, which is certainly lower than the one of a typical original vector space, is still too high for both human interpretation and efficient natural language processing. This problem also affects probabilistic latent semantic analysis [6], a method which has successfully addressed other limitations of LSA.

The main idea of the proposed method is to use LSA's output as an intermediate-space representation which can be further reduced by means of a non-linear projection method such as MDS. In figure 1, the proposed methodology is illustrated in terms of a block diagram.

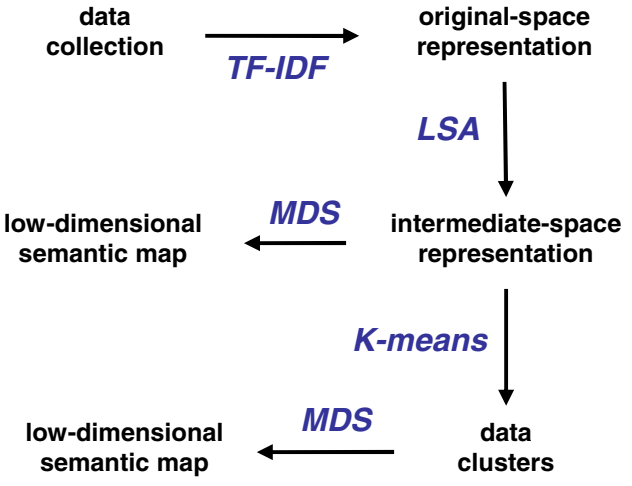


Fig. 1. Block diagram of the proposed methodology

As seen from the picture, the first two steps of the proposed methodology are the standard construction of a term-document matrix and the application of LSA for obtaining an intermediate-reduced-space representation for the data. Then, this intermediate-space representation is mapped into a lower dimensional space by means of MDS. Two different alternatives are depicted for this final projection in figure 1. It can be directly applied to individual samples in the dataset, or it can be applied to a clustered version of it. Ideally, in any of both alternatives, the final space dimensionality is expected to be very low: two or three dimensions.

4 Experimental Work

In this section we will present some preliminary experimental results that aim at demonstrating the principal features and properties of the proposed methodology. More specifically, semantic maps will be generated for different sets of Spanish verbs by applying the proposed methodology to dictionary-based definitions. First, in section 4.1, a brief description of the considered data set is presented. Then, in sections 4.2 through 4.6, some illustrative experiments and their corresponding results are presented and discussed.

4.1 Data Set Description

Since semantic mapping is the main objective of this work, for the sake of illustrating the proposed methodology, we considered important to use a dataset where semantic roles of terms were well described and represented. For this reason, a collection of dictionary-based definitions was considered. More specifically, a Spanish dictionary was used.

The collection of dictionary-based definitions was constructed as follows: for all terms in the Spanish dictionary, their corresponding part-of-speech and definitions

were extracted. For those terms with more than one associated definition (such as polysemes and homonyms), an individual entry in the collection was created for each available definition.

For example, the two resulting entries for the term *preguntar (to ask)* [1] were:

1.- *preguntar | VB | hacer o hacerse preguntas (to make questions to others or to oneself)*
2.- *preguntar | VB | exponer algo en forma de interrogación para expresar duda o para darle mayor expresividad (to expose something in a form of a question for expressing doubt or for making it more expressive)*

The resulting data collection was divided according to the part-of-speech of each individual entry in four different sub-collections: verbs, adjectives, nouns and others. Table 1 presents the basic statistics for each sub-collection and the complete data collection; more specifically, the total amount of terms and definitions, as well as the average length (in number of words) of definitions are provided in the table. The minimum and maximum definition lengths for the complete data collection were 1 word and 113 words, respectively.

Table 1. Dictionary-based data collection

Collection	Terms	Definitions	Aver. length
verbs	4,800	12,414	6.05
adjectives	5,390	8,596	6.39
nouns	20,592	38,689	9.56
others	5,273	9,835	8.01
complete	36,055	69,534	8.32

The verb sub-collection constitutes the dataset that was actually used for all the experimental work presented in this article. By considering the 12,414 definitions within this sub-collection as documents, a term-document matrix was constructed for it. The resulting space dimensionality was 12,913 and it was pruned down to 7,198 dimensions by eliminating all singleton dimensions. Standard TF-ITF was used for weighting and normalization purposes.

Finally, it is important to mention that, although also available in the dictionary, nor synonymic neither antonymic information about the terms and their definitions was used. This was done because we wanted our conclusions about the proposed method's applicability to be generalizable to a more general term-context setting. However, it has been proven that the incorporation of synonymic and antonymic information as additional features into the data space representation improves the performance of related-term identification tasks [5].

4.2 Intermediate Dimension Selection

The first problem we want to address is the incidence of the intermediate-space dimensionality in the overall performance of the method. In this section we illustrate

[1] For all Spanish terms and sentences appearing in this paper, the most appropriate English translations will be provided in parentheses.

Table 2. Two groups of semantically related verbs

Group	Verbs
A	ayudar (to help), compartir (to share), beneficiar (to benefit), colaborar (to collaborate), salvar (to save), apoyar (to support), cooperar (to cooperate), favorecer (to favour)
B	agredir (to threaten), destruir (to destroy), aniquilar (to eliminate), atacar (to attack), arruninar (to ruin), matar (to kill), perjudicar (to perjudice)

with simple examples how the construction of good semantic maps is subject to an appropriate intermediate-space dimension selection.

Consider the two groups of verbs presented in table 2. Notice that each group is related to a different semantic category. While group *A* contains verbs that describe solidarity-related actions, group *B* includes verbs that describe destructive-related actions [2].

By using the proposed methodology, a two-dimensional map can be constructed for the verbs presented in table 2. Figure 2 presents four maps obtained by applying MDS to four different LSA outputs: no LSA dimensionality reduction (original 7,198 dimensions), and LSA projections into 800, 400 and 100 dimensions. Verbs corresponding to groups *A* and *B* are represented with squares and dots, respectively.

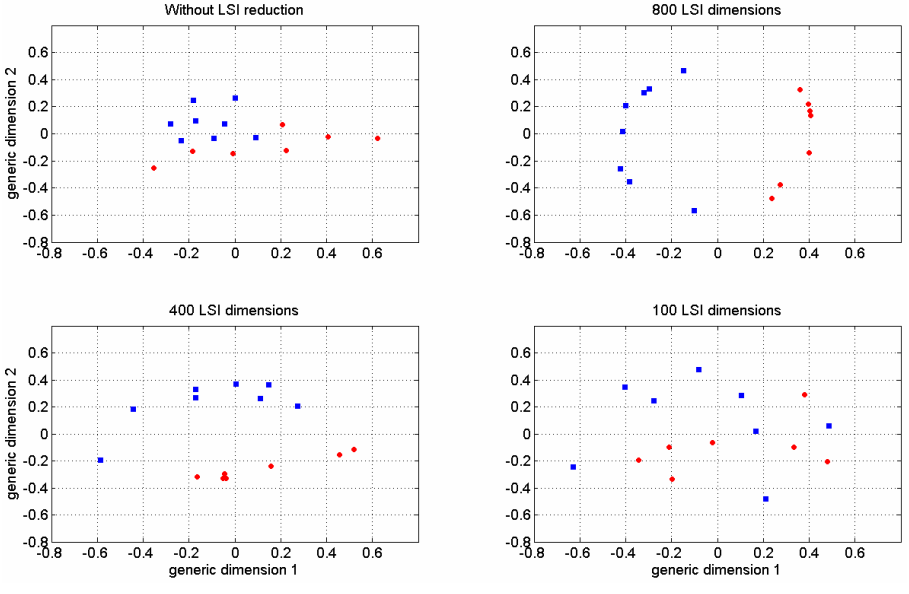


Fig. 2. Resulting maps for two semantic-categories of verbs and different intermediate-space dimensionalities

[2] In both cases, group *A* and group *B*, only the definition (or sense) that was the most representative of the semantic category under consideration was selected for those verbs with more than one definition available.

As observed from the figure, applying MDS to the original term-document matrix does not allow for a good discrimination between the two considered data classes. Similarly, applying MDS to the 100-dimensional representation resulting from LSA does not allow data discrimination. However, when the maps are generated from 800- and 400-dimensional LSA outputs, both data classes under consideration became linearly separable. Such a simple exercise illustrates the existence of a range of intermediate-space dimensions for which appropriate semantic mapping is attainable.

If the intermediate-space dimension is too high, the vector-space representation is too noisy and sparse, so no good maps can be constructed. On the other hand, if the intermediate dimension is too low, semantic features are mixed up during LSA and semantic mapping is not possible either. Only when the intermediate-space dimension is within the appropriate range, LSA is able to provide a good representation for the implicit semantics and MDS is then able to generate good semantic maps by projecting the implicit semantic representation into a low-dimensional space.

4.3 Final Dimension Selection

Projecting an intermediate-space representation into a very-low-dimensional space without loosing data-structure information is only possible if this information can be appropriately represented by a reduced set of features. As one might expect, this should not be always the case. Indeed, such a low-dimensional map should be considered just as a rough representation for the implicit semantics within the dataset under consideration, unless there is clear evidence that data-structure information has been preserved during the mapping process. But, how can we know whether data-structure information is preserved or not?

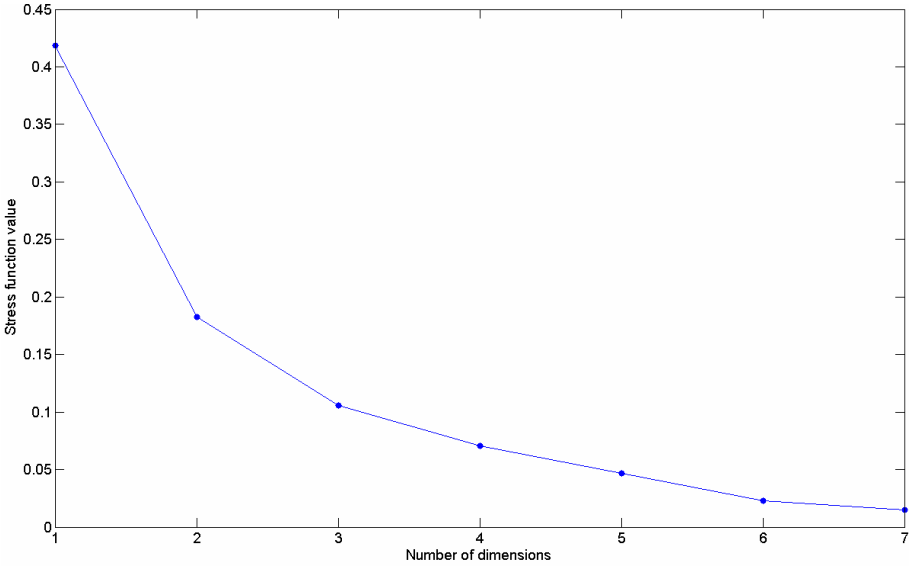


Fig. 3. Stress values vs. number of dimensions for MDS projections of verbs in table2

An answer to this question can be found in the *stress* function that MDS minimizes, which was presented in equation 2. As already mentioned in section 2.2, the minimum achievable value for the *stress* function at a given dimension provides an indication of how much of the original data structure has been lost due to dimensionality reduction. So, this value can be used to select the best or, more appropriately, the "less-distorting" number of dimensions for semantic mapping.

Typically, dimensionality selection for MDS analysis is achieved by plotting *stress* values vs. the number of dimensions [2]. In this kind of plots, a monotonic decrease in *stress* values is observed when dimensionality increases. The best guess for the dimension of the dataset under consideration is generally identified by an elbow-region in which the rate of *stress* reduction becomes smaller. Figure 3 illustrates this *stress*-vs-number-of-dimensions curve for the case of the two verbal groups presented in the previous subsection. More specifically, this curve was generated for the MDS projections computed from 400-dimensional LSA outputs. As seen from the figure, for this particular case, the elbow-region seems to be around 2 and 3 dimensions.

4.4 The Problem of Projecting New Terms

In this subsection we present an example on how new observed lexical forms can be associated to already existent semantic categories by using the sort of semantic maps described in previous subsections. In this case, we will consider the same two semantic categories of verbs defined in table 2 and their corresponding semantic map. Here, we are interested in associating to these two categories of verbs (placing in the map) some new verbs that were not used for the generation of the original map.

The main problem we face here is that MDS is a nonlinear data-dependent projection method. So, there is not a direct way for mapping new terms into an already constructed map. The trivial solution for this problem is to generate a new map by combining both, original and new, term sets into a single data collection and projecting the resulting new dataset into a new map. Although this can be done, the problem with this approach is that original-term locations in the new map will not be equal, or even similar, to original-term locations in the original map. If this represents a problem for the framework analysis under consideration, this solution is not viable.

An alternative method for projecting new terms into an existent map without altering the original-term locations in the map would be as follows. For each new term to be projected, an augmented intermediate-space dataset must be constructed by including all original terms that generated the map along with the new term to be projected. A new map is generated from this new dataset by using MDS. Although the new map is different from the original one, some correspondences are expected to exist between them since they differ only in one single sample.

According to this, the location of the new term in the new map is transformed into a location in the original map by means of the following expression:

$$\boldsymbol{l}_{ori} = [\boldsymbol{M}_{ori}\, \boldsymbol{pinv}(\boldsymbol{M}_{new})]\, \boldsymbol{l}_{new} \tag{3}$$

where $\boldsymbol{M}_{ori}$ and $\boldsymbol{M}_{new}$ are dxm matrices containing the coordinates of all m original terms in the original and new d-dimensional maps, respectively; $\boldsymbol{l}_{ori}$ and $\boldsymbol{l}_{new}$ are $dx1$ vectors containing the coordinates of the newly added term in the original and new maps, respectively; and $\boldsymbol{pinv(\cdot)}$ is the pseudo-inverse matrix operator.

What equation 3 is actually doing is to estimate a linear transformation operator for mapping term locations from the new map into the original map. Such estimation is carried out by only considering the original set of terms. Then, the estimated operator is used for projecting the new term location from the new map into the original map.

In this subsection, we will illustrate the use of this procedure for placing new verbs in the semantic map already generated for verbs in table 2. More specifically, the two-dimensional map computed in subsection 4.2 from the 400-dimensional LSA output was used. Table 3 presents the list of the new verbs (eight in total) to be placed in the map, as well as their corresponding semantic association to either category *A* or *B*.

Table 3. A list of new verbs and their related categories

Group	Verbs
A	solidarizar (to solidarize), apadrinar (to uphold), proteger (to protect), defender (to defend)
B	vencer (to defeat), dominar (to dominate), someter (to enslave), destrozar (to destroy)

Figure 4 presents the resulting locations for the eight new verbs (marked with stars) in the original map. The verb lexical forms are also indicated in the map for reference. As seen from the figure, new verbs have been placed close to those regions occupied by the original verbs in their same semantic categories.

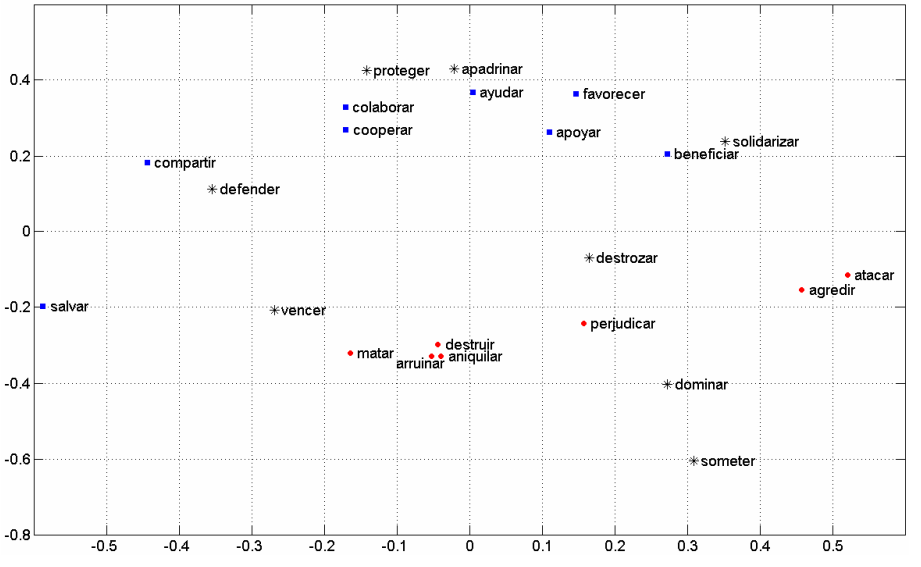


Fig. 4. Placement of eight new verbs (stars) into the original map created with verbs in table 2

4.5 An Example on Semantic Mapping

In this section, we further illustrate the applicability of the proposed methodology for semantic mapping of terms within a more general context. In this case we consider a

list of 53 verbs that have been identified as adequate for redacting objectives in educational curricula. The original list of verbs[3] is actually much more extensive and includes sub-lists of verbs according to six different objective categories.

For the example presented here, we selected only two of these categories: *comprensión (comprehension)* and *evaluación (evaluation)*. Although this categorization does not necessarily imply a semantic affinity among the verbs within each category; in this experiment we are interested in observing how the semantic mapping approach deals with such a set of terms.

Figure 5 presents the two-dimensional map obtained for the list of 53 verbs considered. The verbs included in the categories of evaluation and comprehension are identified with dots and squares, respectively. The verb lexical forms are also indicated in the map for reference.

As seen from the figure, the predefined categories appear mixed together in the map. At a first glance, the map does not seem to be revealing any semantic relations among the terms it includes; however, after a detailed inspection of the map it can be observed that it is actually performing some interesting term discrimination.

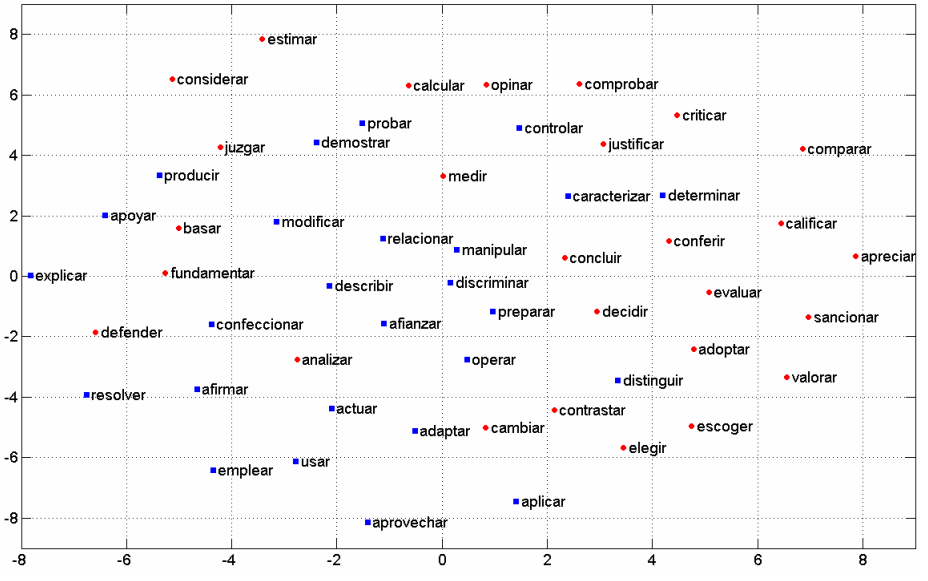


Fig. 5. A two-dimensional semantic representation for objective-redaction recommended verbs

Indeed, the horizontal axis seems to be discriminating evaluative actions from expositive actions. For instance, notice how verbs such as *comparar (to compare)*, *apreciar (to appreciate)* and *calificar (to grade)* appear at the right-end of the map; while verbs such as *explicar (to explain)*, *resolver (to solve)* and *defender (to defend)* appear at the left-end of the map.

On the other hand, the vertical axis seems to be discriminating pragmatic actions from theoretical ones. For instance, notice how verbs such as *aprovechar (to take*

[3] Which is available at http://www.iutlv.edu.ve/investig/archivos/ verbos.pdf.

advantage of), *aplicar (to apply)* and *emplear (to use)* appear at the bottom-end of the map, while verbs such as *estimar (to estimate)*, *calcular (to calculate)* and *comprobar (to verify)* appear at the top-end of the map.

4.6 Semantic Mapping of Clusters

In all previous experiments presented so far, MDS has been used for projecting individual term representations into some low-dimensional maps. In this subsection, we will illustrate the second of the two alternatives depicted in figure 1, for which MDS is used for projecting a clustered version of the original dataset.

In this example, we will consider the totality of the 12,414 entries available in the verb sub-collection. In this case, LSA was applied to reduce the original space representation from 7,198 dimensions into an intermediate-space representation of 800-dimensions. Then, k-means clustering was performed in order to group the 12,414 entries into 1,000 classes or clusters (the basic statistics of the resulting cluster sizes are as follows: minimum and maximum cluster sizes, 2 and 36, respectively; mean and variance, 12.4 and 4.7, respectively). Finally, MDS was applied to project the clustered dataset into a two-dimensional map.

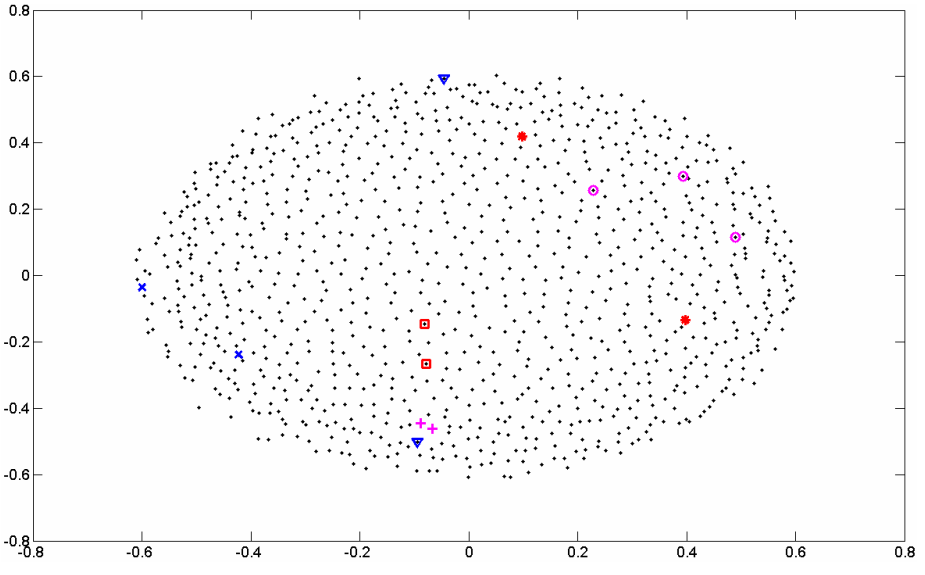


Fig. 6. Semantic map for the 12,414 entries contained in the verb sub-collection

Figure 6 shows the obtained map. For illustrative purposes, the clusters containing some selected term definitions have been identified with different markers (the list of these terms, as well as their corresponding cluster marker is detailed in table 4). As observed from the figure, an interesting spatial representation for the totality of verb definitions has been achieved in terms of clusters, or groups of definitions. In this spatial representation, it is expected for semantically related clusters to appear close to each other, while it is expected for unrelated (or opposite) clusters to appear far from each other.

Table 4. Selected terms and their corresponding cluster marker

Marker	Verbs
square	andar (to walk), saltar (to jump)
circle	leer (to read), escribir (to write), estudiar (to study)
triangle	llover (to rain), solear (to put under the sun)
plus	regar (to water), cultivar (to raise crops)
star	reír (to laugh), llorar (to cry)
ex	navegar (to sail), nadar (to swim)

By considering the examples illustrated in table 4, and looking for their related clusters in figure 6, it can be verified that the expected properties are somehow held by the obtained spatial representation. However, as can be logically suspected, it is not possible to represent the complete set of Spanish verbal senses in such a simple two dimensional map. Indeed, by further exploring the obtained map and considering a larger set of terms and clusters, it can be easily verified that some inconsistencies are also present in the map.

In order to determine an appropriate low-space dimensionality for the 1,000 clusters considered in this experiment, we computed the *stress* function values for different number of dimensions. The resulting curve is presented in figure 7. Notice from the figure that the elbow-region previously described seems to be occurring somewhere between 5 and 10 dimensions. This clearly suggests that a more appropriate representation for the clustered version of the complete verb sub-collection can be obtained if a dimensionality within this range is used for semantic mapping.

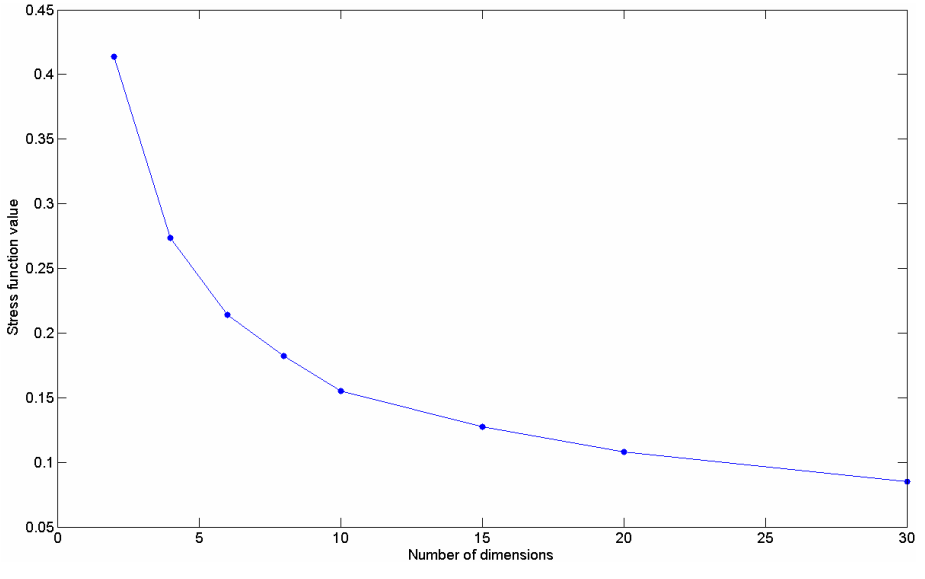


Fig. 7. Stress values vs. number of dimensions for MDS projections of verb clusters

5 Conclusions and Future Work

In this work, we presented an experimental framework in which latent semantic analysis is combined along with multidimensional scaling for allowing further reduction of dimensional complexity while preserving the structural characteristics of a given dataset. The proposed methodology allows for the non-linear projection of vector-space representations into very-low-dimensional maps, in which the implicit structural relationships among the data samples, as well as the spatial representation itself, are more tractable and easier to interpret.

Although the proposed methodology was demonstrated here in the particular context of related term identification, it is intended to constitute a general experimental framework for exploring and evaluating text-mining applications which could benefit from the tractability of very-low-dimensional data representations.

Along the experiments presented in this work some of the most important issues regarding the methodology have been addressed. More specifically, the problems of intermediate-space dimension selection, final-space dimension selection, and new data-sample projection were discussed and illustrated. Additionally, examples on both, individual-sample and clustered, data projections were presented.

As future work, we plan to explore about the applicability and usefulness of the proposed framework within the context of other natural language processing tasks such as document classification, topic detection and tracking, opinion mining and automatic summarization.

In the same way we used the method here for related term identification, it can also be used for related document retrieval and ranking. Similarly, the results from this work can also be exploited for performing vocabulary sparseness reduction by mapping the terms in a document collection into a reduced set of term categories according to the obtained semantic maps.

Acknowledgments

This work has been partially founded by the Spanish Department of Education and Science through the “Ramon y Cajal” fellowship program. The author also wants to thank Barcelona Media Innovation Centre for its support and permission to publish this research.

References

1. Bensch, P.A., Saviteh, W.J.: An occurrence-based model of word categorization. Annuals of Mathematics and Artificial Intelligence 14, 1–16 (1995)
2. Cattell, R.B.: The scree test for the number of factors. Multivariate Behavioral Research 1, 245–276 (1966)
3. Cox, M.F., Cox, M.A.A.: Multidimensional Scaling. Chapman and Hall, Boca Raton (2001)
4. Deerwester, S., Dumais, S., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)

5. Evans, D.A., Handerson, S.K., Monarch, I.A., Pereiro, J., Delon, L., Hersh, W.R.: Mapping Vocabularies Using Latent Semantics. In: Grefenstette, G. (ed.) Cross-Language Information Retrieval, pp. 63–80. Kluwer Academic Publishers, Dordrecht (1998)
6. Hofmann, T.: Probabilistic Latent Semantic Analysis. In: Proceedings of Uncertainty in Artificial Intelligence, UAI 1999, pp. 289–296 (1999)
7. van Eck, N., Waltman, L., van den Berg, J.: A novel algorithm for visualizing concept associations. In: Proceedings of the 16^{th} International Workshop on Database and Expert System Applications, pp. 405–409 (2005)

An Improved Automatic Term Recognition Method for Spanish

Alberto Barrón-Cedeño[1,2], Gerardo Sierra[1],
Patrick Drouin[3], and Sophia Ananiadou[4]

[1] Engineering Institute,
Universidad Nacional Autónoma de México, Mexico
[2] Department of Information Systems and Computation,
Universidad Politécnica de Valencia, Spain
[3] Observatoire de Linguistique Sense-Texte,
Université de Montréal, Canada
[4] University of Manchester and National Centre for Text Mining, UK
alberto@pumas.ii.unam.mx, gsierram@ii.unam.mx,
patrick.drouin@umontreal.ca, sophia.ananiadou@manchester.ac.uk

Abstract. The *C-value/NC-value* algorithm, a hybrid approach to automatic term recognition, has been originally developed to extract multiword term candidates from specialised documents written in English. Here, we present three main modifications to this algorithm that affect how the obtained output is refined. The first modification aims to maximise the number of real terms in the list of candidates with a new approach for the stop-list application process. The second modification adapts the *C-value* calculation formula in order to consider single word terms. The third modification changes how the term candidates are grouped, exploiting a lemmatised version of the input corpus. Additionally, size of candidate's context window is variable. We also show the necessary linguistic modifications to apply this algorithm to the recognition of term candidates in Spanish.

1 Introduction

The *C-value/NC-value* algorithm [3] is the base of the Termine suite for automatic multiword terms recognition in specialised documents in English[1]. TerMine is a text mining service developed by the UK National Centre for Text Mining for the automatic extraction of terms in a variety of domains. This algorithm has been applied to Automatic Term Recognition (ATR) over different languages such as English [3] and Japanese [10]. Additionally, it is the base for an algorithm designed for term extraction in Chinese [4]. A first essay has started to adapt it to handle documents in Spanish [1].

In this paper, we describe the improvements carried out over different stages of the algorithm. Additionally, we show the necessary adaptations for exploiting this algorithm on ATR of terms in Spanish texts. About the distribution of

[1] http://www.nactem.ac.uk/software/termine/

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 125–136, 2009.

the paper, Section 2 gives a brief description of the original algorithm. Section 3 describes the corpora exploited during the design and test of our method. Section 4 describes the modifications we have made to the algorithm including a description of the resources we have exploited. Section 5 contains the evaluations. Finally, Section 6 draws some conclusions and future work.

2 The *C-value/NC-value* Algorithm

The *C-value/NC-value* algorithm was originally developed by Frantzi et al. [3] for multiword ATR on English texts. This hybrid (linguistic-statistical) algorithm is divided into two main stages: *C-value* and *NC-value*. We summarise below the main ideas developed in that paper. (Subsections 2.1 and 2.2, respectively).

2.1 *C-value*: the Hybrid Stage

The task of the *C-value* algorithm is to process an input corpus (composed of a set of specialised texts) in order to generate a list of candidate terms. This list is ranked according to the potential of each candidate of being a real term: its *termhood*.

Figure 1 shows a simplified version of the *C-value* algorithm. The entire version can be found in [3]. The linguistic steps that we have modified will be described in Section 4. In our current approach, we have not defined any threshold in order to previously remove strings (fourth line). The threshold exceeding conditions that let us decide if the *C-value* of a candidate is high enough to consider it as a good term candidate are not used either. The reason is that we do not want to discriminate candidates based on their frequency or *C-value* because, until now, our corpus is not big enough to reach a significant threshold.

The Linguistic Stage. The linguistic filter recognises noun phrases (combination of nouns with prepositions and adjectives), which are potential terms. Section 5.1 contains a comparison of the results obtained by using an open versus a closed filter.

The stop-list is composed of a set of words which are not expected to occur inside of a term on the studied domain (due to this fact the stop-list is domain-dependent). Those candidates with words in the stop-list are removed from the list (Section 4.2 shows the improvement made to this filtering process). The list of candidates obtained by this linguistic process is ranked on the basis of the next statistical process.

The Statistical Stage. The purpose of this part of the algorithm is to measure the termhood of every candidate string. The list of candidate terms is ranked based on this value (*C-value*). The *C-value* calculation considers four aspects:

1. The frequency of the candidate in the entire corpus.
2. The frequency of the candidate when it appears nested in longer candidates.
3. The number of those longer candidates.
4. The length of the candidate (in words).

Algorithm 1: Given the analysis corpus:

$outputList = [\,]$
tag the corpus
extract strings using linguistic filter
remove strings below frequency threshold
filter rest of strings through stop-list
For each string a given that $length(a) = max$
    $C\text{-}value(a) = log_2|a| * f(a)$
    If $C\text{-}value(a) \geq Threshold$
        add a to $outputList$
        For each substring $b \in a$
            revise $t(b)$ and $c(b)$
For each string a given that $length(a) < max$
    If a appears for the first time
        $C\text{-}value(a) = log_2|a| * f(a)$
    Else
        $C\text{-}value(a) = log_2|a|(f(a) - \frac{1}{c(a)}t(a))$
    If $C\text{-}value(a) \geq Threshold$
        add a to $outputList$
        For each substring $b \in a$
            revise $t(b)$ and $c(b)$

Fig. 1. Simplified *C-value* algorithm

The absolute frequency of a term candidate in a corpus is an initial parameter to define if it is a real term. However, it is not enough. In fact, it is common that long terms appear just once even in long corpora. After this appearance, references to this kind of terms often appear only as truncated versions of themself.

For example, in a sample of 139,027 words from the Computer Science Corpus in Spanish [7], the term *computadora* (computer) appeared 122 times. Meanwhile, *computadora personal* (personal computer) appeared only 15 times. In this case, we could guess that, at least in some cases, *computadora* is a simplified version of *computadora personal*, so the nested appearance of the former in the latter decreases the probability of *computadora* of being a real term.

Nevertheless, in the same sample corpus there are some other candidates that are built having to the string *computadora* as the root, such as *computadora portátil* (laptop) and *computadora de uso general* (general purpose computer) appearing 15 and 2 times, respectively. These three different term candidates with the same root, reflect the possibility of *computadora* of being by itself a real term. The other three strings could be varieties with its own concept associated in the subject area (as it is). In such a case, it is considered that all four candidates could be real terms. For these reasons, three considerations are made:

1. The high frequency of a candidate in a corpus is beneficial for its termhood.
2. The length of a string is also beneficial (due to the fact that the probability of a long string of appearing in a corpus decreases as it is longer).

3. The appearance of a candidate nested into another detriments its termhood.
4. If a candidate appears nested in multiple candidate strings, the detrimental effect becomes weaker.

The *C-value* is calculated as in Eq. 1.

$$C\text{-}value = \begin{cases} log_2|a| * f(a) & \text{if } a \notin nested \\ log_2|a| * \left(f(a) - \frac{1}{P(T_a)} \sum_{b \in T} f(b)\right) & \text{otherwise} \end{cases}, \quad (1)$$

where a is the candidate string, $f(\cdot)$ is the frequency of the string $\cdot$ in the corpus, T_a is the set of extracted candidates containing a, and $P(T_a)$ is the number of those candidates. The *nested* set is composed of all those candidates appearing inside of longer candidates.

This process generates the list T_1 of term candidates. T_1 contains the set of candidates ranked by their termhood (*C-value*).

2.2 *NC-value*: Considering the Terms Context

It is hard to think in a word without relating it to some others that interact with it. Sager [11] has stated that terms tend to be accompanied by a strict set of other words (including more terms). In order to illustrate, consider the term *hard disk*. This term will hardly appear with words such as *cook* on its neighbourhood, but it will frequently appear with words such as *GB*, *format*, *install* or *capacity*, which are related to it.

If a term appears with a "closed" set of neighbour words, the existence of these words in the context of a candidate must be positive clues for its termhood. The *NC-value* method extends *C-value* by considering the candidates context with the so called *context weighting factor*. Term context words are those appearing in the neighbourhood of the candidates. However, not all the words in the neighbourhood must be considered as context words. Only nouns, adjectives and verbs (other words do not add significant information to a term).

A list of obtained context words is obtained and ranked according to their "relevance" over the terms. This relevance is based on the number of terms that appear in their contexts. The higher this number, the higher the probability that the word is related to real terms. It is expected that these words appear with other terms in the same corpus. The context weighting factor (that expresses the probability of a word w of being a term context word) is calculated as in Eq. 2.

$$weight(w) = \frac{t(w)}{n}, \quad (2)$$

where w is a term context word, $weight(w)$ is the weight assigned to the word w (expressed as a probability), $t(w)$ is the number of terms w appears with, and n is the total number of terms considered.

The weight assigned to a context words must be calculated after getting the list T_1 that, as we have said, is ordered by the *C-value*. In order to extract the term context words, the top candidate terms in T_1, which present a high

Precision (contains a good proportion of real terms), is used. These top terms produce a list of term context words weighted on the basis of Eq. 2.

The rest of the context words may or may not have an associated context weight. In the case where a context word w has been seen earlier, it retains its associated weight. Otherwise, $weight(w) = 0$. The *NC-value* for the term candidates is calculates as in Eq. 3, which considers the previously calculated *C-value* as well as the context words weights:

$$NC\text{-}value = 0.8C\text{-}value(a) + 0.2 \sum_{b \in C_a} f_a(b) weight(b), \tag{3}$$

where a is the current term candidate, C_a is the set of context words associated to a, b is each one of those context words, and $f_a(b)$ is the frequency of b as a context word of a.

The context information is exploited in order to concentrate the real terms in the top of the list. The new list T_2 of term candidates is ranked on the basis of the *NC-value*.

3 The Corpora

We have used two corpora during the design and evaluation of our prototype: The Linguistic Corpus of Engineering and the Computer Science Corpus in Spanish, described in Subsections 3.1 and 3.2, respectively.

3.1 The Linguistic Corpus of Engineering

The Linguistic Corpus of Engineering (CLI) [9] has been created in the Language Engineering Group at the Universidad Nacional Autónoma de México (UNAM). It is composed of a set of specialised texts on Engineering (mechanics, civil and electronics, among others). Most of the documents are written in Mexican Spanish and includes some texts written in peninsular Spanish. This corpus, consisting of 23 files with 274,672 words, includes postgraduate as well as undergraduate thesis, papers and reports on this area.

Due to the fact that Engineering is a large subject area, we have opted for focusing only on the CLI Mechanical Engineering section. This section includes 5 files for a total of 10,191 words.

The CLI corpus has been used in order to define the rules for the linguistic filter definition corresponding to the candidate extraction subtask (Subsection 4.1).

3.2 The Computer Science Corpus in Spanish

The Computer Science Corpus in Spanish (CSCS) [7] was compiled in the *Observatoire Linguistique Sense-Texte* (University of Montreal). The original objective of this corpus is the development of a Spanish version of DicoInfo [5] "the fundamental computer science and Internet dictionary"[2].

[2] http://olst.ling.umontreal.ca/dicoinfo/

Table 1. Statistics of the CLI and CSCS corpora sections used in our experiments

Feature	Value
CLI	
Number of files	5
Total number of tokens	10,191
Avg. number of tokens per file	2,038
CSCS	
Number of files	200
Total number of tokens	150,000
Avg. number of tokens per file	750

The CSCS contains around 550 documents with more than 500,000 words. It mainly contains texts written in peninsular Spanish. For our experiment, we have chosen the Hardware section, with around 200 documents and almost 150,000 words.

This corpus has been used in order to define the open linguistic filter, corresponding to the candidate extraction task (Subsection 4.1), as well as for evaluation (Subsection 5.1). Some statistics for both corpora are included in Table 1.

4 Improvements to the Algorithm

After explaining the original *C-value/NC-value* algorithm, as well as describing the used corpora, we discuss the adaptations carried out to both the linguistic and statistical sections of the *C-value* method.

4.1 Creating the Linguistic Filter for Spanish

The modified prototype has been designed for ATR over documents written in Spanish. For our experiments we have considered the application of two filters: closed and open. The first one is strict and tries to retrieve only real terms, reducing the number of false positives. The latter is flexible and tries to retrieve all the terms in the corpus no matter the number of false negatives obtained.

The most frequent term patterns in Spanish are Noun -*amplificador*, *protocolo* (amplifier, protocol)-, Noun Prep Noun -*estación de trabajo*, *lenguaje de programación* (work station, programming language)- and Noun Adjective -*computadora personal*, *red neuronal* (personal computer, neural network)- [2]. These patterns compose our closed filter (this set of rules as well as the corresponding to the open filter are in NLTK format [8]):

- $NounAdj$
- $NounPrepDEAdj$
- $Noun$

In the second rule we do not consider any preposition, but only *de* (of). That is the meaning of the tag $PrepDE$.

Additionally, we have carried out a manual term extraction based on the method described in [6]. The objective was to find more flexible patterns in order to retrieve more terms (while trying to limit the generation of noise in the output). The manual extraction carried out over a section of both, CLI and CSCS, corpora resulted in the following set of rules composing the open filter:

- $(Noun|ProperNoun|ForeignWord)^{+}$
- $(NounAdj)(PrepDE(Noun|ProperNoun))^{*}$
- $NounPrepDE(Noun|ProperNoun)$
- $Noun^{?}Acrnm$
- $NounPrepDE((NounAdj)|(AdjNoun))$

Note that some terms contained foreign words (most of them in English). Other part-of-speech such as acronyms and proper nouns have appeared also. The closed or open filter depends on the interest of favouring Precision or Recall in the output (Section 5).

4.2 Modifications to the *C-value* Algorithm

We have detected some weaknesses to the *C-value/NC-value* algorithm. With the aim of reducing them, we have carried out four main modifications.

Selective Stop-Words Deletion. As we have pointed out in Subsection 2.1, a stop-list is applied during the *C-value* stage in order to reduce noise. The original method deletes an entire candidate if it contains at least one stop-word.

A stop-list in ATR is a list of words which are not expected to occur as term words in the treated domain. Our stop-list contains 223 words. It is composed of nouns and adjectives that presented a high frequency in the CSCS but it is not expected to find them inside of real terms. Some examples of these stop-words are *característica*, *compañía* and *tamaño* (feature, company and size, respectively).

Our strategy propose the deletion of stop-words instead of entire candidates. We call this strategy *selective stop-words deletion*. The reason for this selective deletion is that there are a lot of candidate terms containing stop as well as other kind of words. For instance, consider the candidate *computadora grande* (big computer). If we only delete the substring *grande* instead of the entire candidate, keeping *computadora*, the patterm of the obtained candidate is characteristic of the terms in Spanish. And, as it is the case in Computer Science, it becomes a potential real term.

However, the stop-words could be linked to functional words. In order to clarify this point, consider another example. The candidate *desarrollo de LCD* (LCD's development) contains the stop-word *desarrollo*. The POS of this candidate is *Noun PrepDE Noun*. Again, the basic option would be completely discarding this candidate, but *LCD* is a real term. On the selective stop-words deletion strategy, we only delete the stop-word (*desarrollo*). On this way, we obtain *de LCD* with POS *prepDE Noun*. However, this is not a characteristic pattern of terms in Spanish. If the stop-word is a noun, it is necessary to check

those words before and after it in order to decide if they should be deleted also. In this case, the preposition *de* must be deleted. The result is *LCD*, which is a real term.

The selective stop-words deletion strategy is described in Algorithm 2. This algorithm has been designed for Spanish terms. However, after a brief linguistic adaptation it is possible to apply it to any other language).

Algorithm 2: Given a candidate s split into words s_i:

$D = \{\}$ //The set of words that will be deleted from s
If $\mathcal{P}(s_i) = Adjective$ and $s_i \in stop\text{-}list$
  add s_i to D
Elif $\mathcal{P}(s_i) = Noun$ and $s_i \in stop\text{-}list$
  add s_i to D
  If $\mathcal{P}(s_{i-1}) = Preposition$
    add s_{i-1} to D
  If $\mathcal{P}(s_{i+1}) = Preposition$ or $\mathcal{P}(s_{i+1}) = Adjective$
    add s_{i+1} to D
    If $\mathcal{P}(s_{i+2}) = Preposition$ or $\mathcal{P}(s_{i+2}) = Adjective$
      add s_{i+2} to D
delete words in D from s
Return s

Fig. 2. Selective deletion of stop and related words $\mathcal{P}(\cdot)$ = part-of-speech of $\cdot$

In order to clarify how Algorithm 2 works, we give another example. Consider the candidate *pantalla de manera lateral* (screen in lateral way). In this case, $s = \{pantalla_{<Noun>}, de_{<PrepDE>}, manera_{<Noun>}, lateral_{<Adj>}\}$. s_2 (manera) is a stop-word, so $D = \{s_2\}$. *manera* is a noun and for this reason it is necessary to check s_{2-1}, which is a preposition. In this step $D = \{s_1, s_2\}$. The word s_{2+1} (an adjective) must be deleted. Now $D = \{s_1, s_2, s_3\}$. The resulting candidate after deleting D from s is *pantalla* (screen).

Modifying the *C-value* Calculation Formula. The *C-value/NC-value* algorithm was originally designed for the extraction of multiword terms. It is for this reason that the *C-value* calculation formula was not designed to handle terms composed of one word.

Fortunately, this limit is only mathematical. The *C-value* formula is not able to calculate termhood for candidates with $length(a) = 1$ since, in order to normalise the length relevance, it calculates its logarithm (note that $log(1) = 0$, so $C\text{-}value(a) = 0$). In order to avoid this limitation, we add a constant i to the length of a before calculating its logarithm:

$$C\text{-}value = \begin{cases} c * f(a) & \text{if } a \notin nested \\ c * \left(f(a) - \frac{1}{P(T_a)} \sum_{b \in T} f(b) \right) & \text{otherwise} \end{cases}, \qquad (4)$$

where $c = i + log_2|a|$.

On the initial experiments, we tried $i = 0.1$ in order to modify as less as possible the essence of the formula. However, real terms with $length = 1$ used to appear too far in the bottom of the output list, after a lot of bad longer candidates. It is for this reason that reason we define $i = 1$, which (experimentally) produces better rankings.

Searching on a Lemmatised Corpus. The first step in the *C-/NC-value* algorithm is POS tagging the corpus. However, the example output included in [3], includes the strings *B cell* and *B cells* in different rows of the *NC-value* ranked list. This reflects that there is no lemmatisation process involved. We consider that including such a process is important for term extraction. In the case when the lemmatised corpus is used to build the candidate terms list, different variations of the same candidate term are considered as one and its total frequency is the addition of all the variations frequencies.

In order to join the different variations of a candidate, we lemmatise the corpus before processing it. We carry out this subtask with TreeTagger [12]. This tool is a POS tagger as well as a lemmatiser.

4.3 Modifying the *NC-value* Algorithm

The *NC-value* stage is based on considering the candidates context. A word appearing frequently in the neighbourhood of a term in the top of the list ranked by *C-value* (with a high probability of being a real term) has a good probability of appearing with other real terms (no matter if they are in a lower position of the list).

The context for the candidates was originally defined as a fixed window of length 5. However, we have opted for using flexible frontiers to define the context windows. Punctuation marks (point, colon, semicolon, parenthesis) break phrases. Due to this fact a context window is broken if it contains one of these marks (no matter the length of the resulting window).

5 Evaluation

Our version of the *C-value/NC-value* algorithm for ATR has been evaluated in terms of Precision and Recall. In 5.1 we evaluate the extractor with different configuration parameters. Section 5.2 compares our adaptation to another previously designed for Chinese [4].

5.1 Varying the Parameters for the Extraction

We have randomly selected a set of documents from the CSCS [7]. The test corpus contains 15,992 words on the Hardware subject. In order to evaluate the obtained results we have carried out a manual term extraction over the same test corpus.

We have carried four experiments in order to compare different parameters combinations. These combinations are the following:

A Open linguistic filter without stop-list
B Open linguistic filter with stop-list
C Closed linguistic filter without stop-list
D Closed linguistic filter with stop-list

The open and closed filters are described in section 4.1. An open linguistic filter is flexible with the terms patterns. For this reason, it increases Recall reducing Precision. A closed linguistic filter is strict with the accepted patterns. For this reason, it increases Precision reducing Recall.

A total of 520 terms were found during the manual extraction process. The results obtained by the different automatic extractions are shown in Table 2.

Table 2. Result of automatic extractions with different parameters

Case	Candidates	Real terms	P	R
A	1,867	430	0.230	0.826
B	1,554	413	0.265	0.794
C	1,000	241	0.240	0.463
D	850	262	0.308	0.503

As it is expected, considering an open filter benefits Recall but harms Precision while the using a closed filter benefits precision but harms Recall. Looking more closely at the results obtained by experiments *C* and *D*, we can see that the Recall obtained by the latter is higher. This improvement is due to the fact that after the selective deletion, carried out in experiment *D*, more real terms (mainly of length 1) that originally appeared combined to stop-words are discovered. The original approach discards those candidates (Section 4.2).

5.2 Comparing our Adaptation with a Chinese Version

The *C-value/NC-value* method has been implemented and modified previously. [4] have developed a term extractor for texts on IT written in Chinese. In this case, the *NC-value* stage is replaced by a semantic and syntactic analysis stage. The objective of this stage is better ranking the obtained output.

The reported experiments on a sample corpus of 16 papers (1,500,000 Chinese characters) obtain $Precision = 0.67$ and $Recall = 0.42$. Our experiment *B*, obtains $Precision = 0.265$ and $Recall = 0.794$. Although their Precision is better than ours, we must consider that they use a previously obtained list of 288,000 terms of $length = 1$. This list is a filter that separates good candidates from bad ones.

Unlike them, we have opted for conserving the philosophy of the *C-value/NC-value* method: our approach only needs a POS tagger in order to carry out the extraction process.

We must say that we have not compared our algorithm to the originally described in [3], because our experiment conditions are quite different mainly from the stop-list and the corpus features points of view.

6 Conclusions

In this paper we have described a linguistic and functional adaptation of the *C-value/NC-value* algorithm for automatic term recognition. The main functional adaptations carried out are the following:

- A new algorithm for the selective elimination of stop-words in the term candidates has been designed.
- The *C-value* calculation formula has been adapted in order to allow handle candidates of one word.
- The length of the candidates context windows has been is not fixed. Unlike the default $length = 5$, it is dynamically re-sized when it includes punctuation marks.

About the linguistic adaptations, we have analysed the patterns of the terms in Spanish in order to build an open and a closed filter for candidates detection. The open filter favours Recall, while the closed filter favours Precision. Additionally a stop-list composed of around 200 nouns and adjectives has been created.

With respect to other versions of *C-value/NC-value* method, our obtained Precision has decreased. The main reason for this behaviour is that we consider candidates of one word. Moreover, we have not defined any threshold in order to eliminate candidates with low frequency or *C-value*. We have opted for supporting the noise for the sake of a minimum loss of information, resulting in a good Recall.

Finally, we have designed a selective stop-words deletion method. Our method discovers good term candidates that are ignored when considering the original stop-word deletion method.

Acknowledgements. This paper has been partially supported by the National Council for Science and Technology (CONACYT); the DGAPA-UNAM; the General Direction of Postgraduate Studies (DGEP), UNAM; and the Macro-Project *Tecnologías para la Universidad de la Información y la Computación*, UNAM.

References

1. Barrón, A., Sierra, G., Villaseñor, E.: C-value aplicado a la extracción de términos multipalabra en documentos técnicos y científicos en español. In: 7th Mexican International Conference on Computer Science (ENC 2006). IEEE Computer Press, Los Alamitos (2006)
2. Cardero, A.M.: Terminología y Procesamiento. Universidad Nacional Autónoma de México, Mexico (2003)
3. Frantzi, K., Ananiadou, S., Mima, H.: Automatic recognition of multi-word terms: the C-value/NC-value method. International Journal on Digital Libraries 3(2), 115–130 (2000)

4. Ji, L., Sum, M., Lu, Q., Li, W., Chen, Y.-R.: Chinese terminology extraction using window-based contextual information. In: Gelbukh, A. (ed.) CICLing 2007. LNCS, vol. 4394, pp. 62–74. Springer, Heidelberg (2007)
5. L'Homme, M.C.: Conception d'un dictionarie fundamental de l'informatique et de l'Internet: selection des entrées. Le langage et l'homme 40(1), 137–154 (2005)
6. L'Homme, M.C., Bae, H.S.: A Methodology for Developing Multilingual Resources for Terminology. In: Language Resources and Evaluation Conference (LREC 2006), pp. 22–27 (2006)
7. L'Homme, M.C., Drouin, P.: Corpus de Informática en Español. Groupe Éclectik, Université de Montréal, http://www.olst.umontreal.ca/
8. Loper, E., Bird, S.: NLTK: The natural language toolkit. In: ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, pp. 62–69 (2002)
9. Medina, A., Sierra, G., Garduño, G., Méndez, C., Saldaña, R.: CLI. An Open Linguistic Corpus for Enineering. In: IX Congreso Iberoamericano de Inteligencia Artificial (IBERAMIA), pp. 203–208 (2004)
10. Mima, H., Ananiadou, S.: An application and evaluation of the C/NC-value approach for the automatic term recognition of multi-word units in Japanese. International Journal on Terminology 6(2), 175–194 (2001)
11. Sager, J.C.: Commentary by Prof. Juan Carlos Sager. In: Rondeau, G. (ed.) Actes Table Ronde sur les Problémes du Découpage du Terms, pp. 39–74. AILA-Comterm, Office de la Langue Française, Montréal (1978)
12. Schmid, H.: Improvements in Part-of-Speech Tagging with an Application to German. In: ACL SIGDAT-Workshop (1995)

Bootstrapping a Verb Lexicon for Biomedical Information Extraction

Giulia Venturi[1], Simonetta Montemagni[1], Simone Marchi[1], Yutaka Sasaki[2,3], Paul Thompson[2,3], John McNaught[2,3], and Sophia Ananiadou[2,3]

[1] Istituto di Linguistica Computazionale, CNR, Pisa, Italy
{giulia.venturi,simonetta.montemagni,simone.marchi}@ilc.cnr.it
[2] School of Computer Science, University of Manchester, UK
[3] National Centre for Text Mining, University of Manchester, UK
{yutaka.sasaki,paul.thompson,jock.mcnaught,
sophia.ananiadou}@manchester.ac.uk

Abstract. The extraction of information from texts requires resources that contain both syntactic and semantic properties of lexical units. As the use of language in specialized domains, such as biology, can be very different to the general domain, there is a need for domain-specific resources to ensure that the information extracted is as accurate as possible. We are building a large-scale lexical resource for the biology domain, providing information about predicate-argument structure that has been bootstrapped from a biomedical corpus on the subject of E. Coli. The lexicon is currently focussed on verbs, and includes both automatically-extracted syntactic subcategorization frames, as well as semantic event frames that are based on annotation by domain experts. In addition, the lexicon contains manually-added explicit links between semantic and syntactic slots in corresponding frames. To our knowledge, this lexicon currently represents a unique resource within in the biomedical domain.

Keywords: domain-specific lexical resources, lexical acquisition, syntax-semantics linking, Information Extraction, Biological Language Processing.

1 Introduction

It is well known that Information Extraction applications require sophisticated lexical resources to support their processing goals. In particular, accurate applications focussed on extraction of event information from texts require resources containing both syntactic and semantic information. Many applications could benefit from lexical resources providing an exhaustive account of the semantic and syntactic combinatorial properties of lexical units conveying event information.

The need for such resources increases when dealing with texts belonging to a specialized domain such as biology. There are several reasons for requiring domain-specific lexical resources. Even more than in general language, within specialized domains, much lexical knowledge is idiosyncratically related to the individual behavior of lexical units. In particular, it can be the case that the types of events mentioned in domain-specific texts are described using predicates that do not feature prominently

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 137–148, 2009.

in the general language domain and may not be included in general language resources. Or, in the reverse case, predicates that do occur in the general language domain may have different syntactic or semantic properties within the specialized domain. Using information about such predicates from general language resources may result in incorrect analyses or interpretations.

The lexical component still remains a major bottleneck for current Information Extraction systems, especially when the target is event information in domain-specific collections of documents. So far, most lexical resources providing information on predicate-argument structure have been developed manually by lexicographers. It is, however, a widely acknowledged fact that manual work is costly and the resulting resources have limited coverage. Last but not least, porting to new domains is a labour-intensive task. Automatic or semi-automatic lexical acquisition is a more promising and cost-effective approach to take, and is increasingly viable given recent advances in NLP and machine learning technology, together with availability of corpora.

In the European BOOTStrep project (FP6 - 028099), we are building a large-scale domain-specific lexical resource [1] also providing information about predicate-argument structure that is bootstrapped from texts. The topic of this paper is the bootstrapping of predicate-argument structure information from biomedical corpora; in particular, we focussed on *verbs*, for which syntactic subcategorization and semantic event frames have been acquired from a biomedical corpus on the subject of E. Coli. Subcategorization extraction has been carried out through unsupervised learning operating on the dependency-annotated text without relying on any previous lexico-syntactic knowledge about subcategorization frames. Semantic frames are currently based on a subset of the corpus used for subcategorization extraction, which has been manually annotated with gene regulation bio-events by domain experts. The two sets of frames were obtained independently, resulting in two different and unrelated sets of subcategorization and semantic event frames. On the two sets of frames acquired for the same verbs, the syntax-semantics linking was performed manually. The resulting verb lexicon thus includes subcategorization and semantic frames information as well as the explicit linking between semantic and syntactic slots in corresponding frames. To our knowledge, such a lexicon currently represents a unique resource in the biomedical domain, which has the potential to effectively support event extraction from biomedical texts.

The paper is organized as follows: section 2 provides the background and the motivation of our work, whilst section 3 outlines our approach to lexicon construction. Sections 4 and 5 report respectively on the processes of subcategorization induction and event frame extraction. Section 6 concerns the linking of the acquired syntactic and semantic frames. Conclusions and further work are reported in section 7.

2 Background

Various research groups are currently concerned with the creation of corpus-based general-purpose lexical semantic resources providing information on predicate-argument structure; see for instance the FrameNet [2] and PropBank [3] projects.

The FrameNet project, following Fillmore's theory of *frames semantics* [4], is creating an on-line lexical resource supported by corpus evidence. It documents the

range of semantic and syntactic combinatory possibilities of each word in each of its senses, through computer-assisted annotation of example sentences and automatic tabulation and display of the annotation results. One of the major outcomes of this work is represented by the FrameNet lexical database, in which each predicative lexical unit (i.e. verb, noun or adjective) is paired with a semantic frame, i.e. a conceptual structure describing a particular type of situation or event along with its participants. For example, the lexical entry for the verb *construct* identifies the semantic frame underlying its meaning, which is "Building", and whose core frame elements are Agent, Created_entity, Components. The lexical entry also specifies the ways in which frame elements are syntactically realised in texts.

A slightly different approach has been followed within the PropBank project. Both a corpus of one million words of English text, annotated with argument role labels for verbs on the top of the Penn-II syntax trees, together with a lexicon defining those argument roles on a per-verb basis, have been created. For example, the predicate-argument structure of the verb *construct* has been annotated with the following numbered arguments: ARG0 (i.e. builder), ARG1 (i.e. construction), ARG2 (i.e. material), ARG3 (i.e. end state of ARG1).

In response to the requirement for domain-specific lexical resources, a number of attempts have been made to produce domain-specific extensions of the resources described above, e.g. BioFrameNet [5] and PASBio [6]. BioFrameNet is a domain-specific FrameNet extension, mainly focused on the domain concepts of intracellular transport. PASBio, extending a model based on PropBank to molecular-biology domain, takes the role of a reference resource in the stage of corpus annotation for creating training examples for machine learning (i.e. Event Extraction). Currently, these resources are reasonably small-scale (PASBio currently contains 30 predicates, whilst BioFrameNet was carried out as dissertation work).

To our knowledge, the only existing computational lexicon specifically developed for the biomedical domain is the SPECIALIST lexicon [7]. Unlike the previously mentioned cases, the lexicon is built and maintained manually and is not corpus-driven. It is a large lexicon of general English words and biomedical vocabulary, designed to provide the lexical information needed for the SPECIALIST Natural Language Processing System (NLP). Lexical entries in this lexicon also include verb complementation patterns providing important syntactic information.

3 Our Approach

We are building a *verb lexicon* to address the requirement for a large-scale resource that is specific to the biomedical domain, and includes *both* syntactic subcategorization *and* semantic event frame information. Our approach to the construction of the lexicon has a number of defining features, which set it apart from the other resources described above.

Firstly, in contrast to the SPECIALIST lexicon, our own lexicon construction technique is corpus-based. This ensures that the most relevant verbs are included within the lexicon, and their encoded behaviour is domain-specific.

Secondly, in contrast to the purely manual construction method of many other lexical semantic resources, the information in our lexicon has been derived semi-automatically,

using different techniques and different sizes of corpora to obtain each type of information. The extraction of subcategorization frames was carried out using an unsupervised learning technique, using a dependency annotated corpus of approximately 6 million tokens (consisting of both MEDLINE abstracts on the subject of E.Coli, in addition to full papers). In contrast, event extraction was carried out based on a subset of this corpus (677 abstracts), which was manually annotated with bio-event information. This annotation was carried out on top of linguistic annotations covering morphosyntax and shallow syntax ("chunking"). The final step of the process was to link the syntactic arguments of predicates to their semantic counterparts in the event frames, thus facilitating the automatic labelling of syntactic arguments of verbs with semantic roles. In the current work, this linking step has been carried out manually.

In the following sections, we discuss the different techniques of obtaining syntactic and semantic information for inclusion within the lexicon, together with the merging and linking of the results.

4 Extraction of Subcategorization Frames

For the purposes of the extraction of subcategorization frames (hereafter referred as SCFs), we adopted a "discovery" approach to SCF acquisition, based on a looser notion of subcategorization frame, which includes typical verb modifiers in addition to strongly selected arguments. Such an approach took into account the desideratum within the biomedical field that subcategorization patterns should also include strongly selected modifiers (such as location, manner and timing), as these are deemed to be essential for the correct interpretation of texts [8].

In order to meet this basic requirement, we used the Enju syntactic parser for English [9][1], characterised by a wide-coverage probabilistic HPSG grammar and an efficient parsing algorithm, and whose output is returned in terms of predicate-argument relations. In particular, we used the Enju version adapted to biomedical texts [10]. The SCF induction process was performed through the following steps:

- syntactic annotation of the acquisition corpus with Enju (v2.2). The acquisition corpus included both MEDLINE abstracts and full papers containing a total of approximately 6 million word tokens;
- for each verbal occurrence, extraction of the observed dependency sets (ODSs). Each ODS is represented as a set of dependencies described in terms of relation type (e.g. ARG1, ARG2, etc.) complemented in some cases with information concerning the morpo-syntactic category of the head (this information type is useful to further specify generic dependency relations like MOD). For what concerns prepositional and sentential complements, rather than using the general Enju labels (i.e. ARG1, ARG2), a representation was reconstructed in which the preposition or conjunction introducing the complement was made explicit: due to its crucial role in the subcategorization induction process, this information type is part of the dependency label (e.g. PP-in or that-CL) used in the ODS. The order of the dependencies in each ODS is normalised and does not reflect their order of occurrence in context;

[1] http://www-tsujii.is.s.u-tokyo.ac.jp/enju/

- induction of relevant SCF information associated with a given verb. For each observed dependency set, the conditional probability given the verb type *v* was computed: thresholding was used to filter out noisy frames (i.e. frames containing not only arguments and strongly selected modifiers, but also adjuncts) as well as possible errors of either parsing or ODS extraction. After careful examination of the results obtained with different thresholds, ODSs with an associated probability score >= 0.03 were selected as eligible SCFs to be included in the resulting verb lexicon.

For each acquired SCF, the following information types are specified: its conditional probability given the verb (i.e. "p(subcat|v)") and the percentage of times it occurs with the verb in the passive voice (i.e. "Pass"). It should be noticed that each SCF has been extracted for one *normalised verb token*, i.e. the extraction process makes abstraction from the passive usages. Thus, the latter information is particularly useful to account for SCFs typically associated with the verb used in the passive voice; this is the case, for instance, of the SCFs ARG1#ARG2#TO-INF# and ARG1#ARG2#that-CL# frames which with the verb *find* appear to be typically associated with the verb used in the passive voice (e.g. *This was found to be interesting*). Such information has been exploited during the syntax-semantics linking in order to reconstruct the full syntactic realisations of bio-verb arguments even though some of them do not have any semantic counterpart explicitly mentioned in the text.

Table 1. Subcategorization frame examples

Verb	SFC	p(subcat|v)	Pass
abolish	ARG1#ARG2#	0.8669767	0.1437768
abolish	ARG1#ARG2#MOD@VBG#	0.0390697	0.1904761
abolish	ARG1#ARG2#PP-in#	0.0939534	0.7029702
accumulate	ARG1#ARG2#	0.2940677	0.0403458
accumulate	ARG1#	0.4627118	0
accumulate	ARG1#ARG2#PP-in#	0.1084745	0.140625
accumulate	ARG1#PP-in#	0.1347457	0

5 Event Frame Extraction

This section briefly describes the automatic extraction of semantic event frames based on a corpus of 677 MEDLINE abstracts. The abstracts have been annotated with Gene Regulation events by a group of domain experts [11]. Annotation is centered on both verbs and nominalised verbs that describe relevant events within the corpus. For each event, semantic arguments that occur within the same sentence are labelled with semantic roles (see Table 2) and Named Entity types.

We chose to use a set of 13 *event-independent* semantic roles, which were defined specifically for the task though the examination of a large number of relevant events

Table 2. Semantic roles

Role Name	**Description**	**Example** (**bold** = semantic argument, *italics* = focussed verb)
AGENT	Drives/instigates event	**The narL gene product** *activates* the nitrate reductase operon
THEME	a) Affected by/results from event b) Focus of events describing states	**recA protein** was *induced* by UV radiation The **FNR protein** *resembles* CRP
MANNER	Method/way in which event is carried out	cpxA gene *increases* the levels of csgA transcription by **dephosphorylation** of CpxR
INSTRUMENT	Used to carry out event	EnvZ *functions* through **OmpR** to control NP porin gene expression in E. Coli.
LOCATION	Where *complete* event takes place	Phosphorylation of OmpR *modulates* expression of the ompF and ompC genes in **Escherichia coli**
SOURCE	Start point of event	A transducing lambda phage was *isolated* from **a strain** harboring a glpD''lacZ fusion
DESTINATION	End point of event	Transcription is activated by *binding* of the cyclic AMP (cAMP)-cAMP receptor protein (CRP) complex to **a CRP binding site**
TEMPORAL	Situates event in time/ w.r.t another event	The Alp protease activity is *detected* in cells **after introduction** of plasmids
CONDITION	Environmental conditions/changes in conditions	Strains carrying a mutation in the crp structural gene fail to *repress* ODC and ADC activities in response to **increased cAMP**
RATE	Change of level or rate	marR mutations *elevated* inaA expression by **10- to 20-fold** over that of the wild-type.
DESCRIPTIVE-AGENT	Descriptive information about AGENT of event	HyfR *acts* as **a formate-dependent regulator**
DESCRIPTIVE-THEME	Descriptive information about THEME of event	The FNR protein *resembles* **CRP**.
PURPOSE	Purpose/reason for the event occurring	The fusion strains were *used* **to study** the regulation of the cysB gene

in E. Coli abstracts. Event-independent semantic roles have previously been used in large-scale projects involving the production of semantic frames for general language verbs, e.g. VerbNet [12] and SIMPLE [13]. However, to our knowledge, our work is the first to propose a set of event-independent roles for use within the biological domain.

We used VerbNet and SIMPLE as a starting point for the definition of our role set, with the assumption that certain semantic roles are common across all domains. This assumption was confirmed through examination of examples within our corpus, resulting in our use of roles such as AGENT, THEME, and SOURCE. Whilst some general language roles do not seem relevant to the description of biological events (such as BENEFICIARY or EXPERIENCER), others are particularly important to the precise definition of complex biological relations, even though not necessarily specific to the field, e.g. LOCATION and TEMPORAL (see [8]). To the subset of relevant roles identified from VerbNet and SIMPLE, we added the role CONDITION. This corresponds to descriptions of environmental conditions, which are highly important within the domain.

5.1 Event Annotation Spans

An event annotation span is a continuous annotation associated with the same event id within an abstract. An event annotation span begins with the text span covered by the earliest semantic argument, and ends with the latest semantic argument associated with the event within the text.

For example, given the sentence "transfer operon expresses F-like plasmids", its event annotation span is as follows:

```
<SLOT eventid="9" Role="Agent"> <NE cat="DNA"> transfer
operon</NE></SLOT> <EVENT id="9"><SLOT eventid="9"
Role="Verb"> expresses </SLOT></EVENT> </SLOT> <SLOT
eventid="9" Role="Theme"> <NE cat="DNA"> F-like plas-
mids </NE></SLOT>
```

5.2 Syntactic Analysis of Event Annotation Spans

For each event, each event annotation span is syntactically analyzed as follows:

- Tokenize the span into XML tags and words where named entities (NEs) are treated as single words.
- Decide on the POS tags and lemmas of tokens. For words occurring outside of NE spans, “O” is assigned as the value of the NE category field. NEs are assigned “NN” as the value of the POS field.
- Add semantic role labels to words and NEs based on the IOB labelling scheme. That is, add B-*role* to the first word in the *role* annotation, and I-*role* to the following words in the annotation.

For example, the sentence introduced above is analyzed as shown in Table 3.

Table 3. Example syntactic analysis of event annotation span

word	POS	lemma	NE	Role
transfer operon	NN	transfer operon	DNA	B-Agent
expresses	VBZ	express	O	B-Verb
F-like plasmids	NN	F-like plasmids	DNA	B-Theme

5.3 Event Frames

Event frames take the following general form:

```
event_frame_name(
      slot_name => slot_value,
      …
      slot_name => slot_value),
```

where

- `event_frame_name` is the base form of the event verb or nominalized verb;
- `slot_names` are the names of the semantic roles within the event pattern;
- `slot_values` are NE categories, if they have been assigned within the event pattern.

5.4 Event Frame Extraction

Converting syntactically analyzed event annotation spans to semantic event frames is straightforward.

- the event frame name is the lemma of the verb;
- for each semantic role (starting with a B-*role* label and followed by I-*role* labels), use its NE as the slot value, if an NE has been assigned.

For example, the event frame corresponding to the above event annotation span example is as follows:

```
express( Agent=>DNA,
           Theme=>DNA ).
```

6 Syntax-Semantics Linking

The syntax-semantics linking was carried out manually on the basis of different information types. The starting point of this process was represented by:

- the list of 1760 subcategorization frames, acquired from the Enju annotated corpus (see section 4);
- the list of 856 verbal bio-event frames based on annotations in the Gene Regulation corpus (see section 5); it should be noticed that for the linking purposes we took into account bio-event frames including both slots which specify a named entity category, as well as those slots which do not specify such information.

The linking focussed on 168 verbs for which both subcategorization and event frame information was available, in particular on the 628 subcategorization frames and the 486 bio-event frames extracted for those verbs.

The linking process was carried out manually and it was defined by simultaneously taking into account different information types, in particular:

- we considered that a syntax-semantic mapping process is controlled by strategies which presuppose hierarchies of semantic roles and grammatical functions.

- we made use of a list of 'prototypic' syntactic realisations of semantic arguments, as provided in the annotation guidelines followed by annotators during the manual annotation of bio-event frames (provided in [14]).
- we exploited general language repositories of semantic frames containing both syntactic and semantic information as possible benchmarks,
- we also referred to the manually annotated Gene Regulation Corpus, when the evidence of the other information sources was not sufficient to perform the syntax-semantics mapping.

Firstly, we analysed the literature regarding syntax-semantics linking, according to which "Thematic Hierarchies" appear to be by far the most widely used method to explain the mapping from semantic representation to syntax. A hierarchy of "cases" (semantic relations) was first formulated by Fillmore [15] to help determine subject selection. After him, most theories make use of a mapping between an ordered list of semantic roles and an ordered list of grammatical relations. Thus, rather than having invariable correspondence relations, these approaches suggest that, given a thematic role hierarchy (agent>theme ...) and a syntactic functions hierarchy (subject>object ...), the mapping usually proceeds from left to right, mapping the semantic role further to the left onto the first available position in the syntactic hierarchy. Several proposals have been made for what concerns the thematic role hierarchy which widely differ a) with respect to the theoretical stands and b) in what is being hierarchisized. If on the one hand there is general agreement on the fact that the Agent role should be the highest ranking role, on the other hand no consensus is found in the literature (see [16] for a survey of the wide range of proposals) for what concerns the relative ordering of the remaining roles.

Another important source of information was represented by the 'prototypic' syntactic realizations of semantic arguments as defined in the annotation guidelines for event annotation in the Gene Regulation Corpus, especially for what concerns less prominent roles, typically expressed as prepositional phrases. In order to solve doubtful mapping cases, general language repositories of semantic frames containing both syntactic and semantic information were also consulted. Amongst others, we choose to exploit VerbNet [12] because, similarly to our own work (see section 5), it uses a set of frame-independent thematic roles. The Gene Regulation corpus was also taken as a further source of evidence: in particular, it was useful in dealing with verbs that do not feature in a general language repository of frames or that may have different syntactic realisations or different semantic properties within the biomedical domain.

The linking process resulted in 668 linked frames. Different types of mapping were performed, namely full and partial mapping. In full mapping cases, the arity of the subcategorization and bio-event frames is the same; that is to say that all semantic arguments of the bio-event frame have a syntactic counterpart at the level of the subcategorization frame. For what concerns partial mapping, we distinguished the following sub-cases:

1. the semantic frame contains more slots (i.e. semantic roles) than the corresponding subcategorization frame. In these cases, a mapping could only be defined for a subset of the semantic roles in the bio-event frame. For example, for the verb *express*, for which the semantic frame Agent#Theme#Location#Condition# and

the subcategorization frame ARG1#ARG2#PP-in# have been acquired the following mapping has been defined:

AGENT>ARG1#THEME>ARG2#LOCATION>PP-in#**CONDITION>0**

2. subcategorized slots do not find a semantic counterpart in the corresponding bio-event frame. This is typically the case of event frames which did not contain explicit mention of an AGENT role, which however has been reconstructed as ARG1 at the level of the subcategorization frame: this applies most frequently to passive sentences such as *The wild-type pcnB gene was cloned into a low-copy-number* plasmid, whose Enju normalised syntactic representation includes a reconstructed ARG1 which does not correspond to any filled semantic argument of the corresponding bio-event frame. Consider as an example the verb *introduce*, for which the semantic frame Theme#Destination# and the subcategorization frame ARG1#ARG2#PP-into# have been extracted; in this case the mapping presents itself as follows:

   0>ARG1#THEME>ARG2#DESTINATION>PP-into

3. a combination of cases 1) and 2) above, i.e. where the semantic frame contains more slots than the corresponding subcategorization frame on the one hand, and a reconstructed ARG1 does not have any counterpart at the level of the semantic frame on the other hand. Consider as an example the verb *delete*, for which the following mapping has been defined, operating respectively on the ARG1#ARG2#PP-from# and Theme#Source#Condition# subcategorization and event frames:

   0>ARG1#THEME>ARG2#SOURCE>PP-from#**CONDITION>0**

Table 4 below summarises the results of the linking process. Note that 28 extracted bio-event frames were discarded since they turned out to originate from errors during the semantic annotation process.

Table 4. Syntax-semantics linking results

Type of mapping		Number of cases	%
Full mapping		239	35.77
Partial mapping	Sub-case 1	123	18.42
	Sub-case 2	166	24.86
	Sub-case 3	140	20.95
	TOTAL	668	100.00

7 Conclusion

In this paper, we have described the bootstrapping of a verb lexicon for Biomedical information extraction. The verb lexicon includes both syntactic subcategorization frames and semantic event frames, together with a bridge between the two levels.

The information within the lexicon is the result of integrating information extracted from corpora of different sizes and using different techniques. Syntactic subcategorization frames were acquired from an automatically annotated corpus (dependency

annotation) of 6 million word tokens, using unsupervised learning. On the other hand, event frames were extracted from a subset of this corpus (677 MEDLINE abstracts) that was manually annotated by biologists. The link between the syntactic and semantic levels of information was also carried out manually.

The syntax-semantics linking was carried out on 168 biologically relevant verbs, for which both subcategorization and event frame information was available. A total of 628 subcategorization frames and the 486 bio-event frames had been extracted for those verbs. As a result of this linking process, 668 event frames have been fully or partially linked to subcategorization frames.

To our knowledge, the number of verbs covered by our lexicon, together with the typology of information that is available for each verb, make our resource unique amongst large-scale computational lexicons within the biomedical domain.

We are currently working on an extrinsic evaluation of the syntactic/semantic frames in bio-event IE tasks. The verb lexicon is an essential resource in these IE tasks, and is utilized as follows:

- Analyze bio-event text using the Enju full parser;
- Find predicate-argument structures that match subcategorization frames in the verb lexicon;
- Using the linking tables, map the matched predicate-argument structures to semantic event frames;
- Finally, by applying event frames to these semantic frames, event instances can be extracted.

In addition to events that are centred on verbs, our event frame corpus includes annotations corresponding to events that are centred on *nominalised verbs* such as *regulation* and *expression.* As events expressed in such a way play an important and possibly dominant role within biomedical texts [17], we plan to acquire subcategorization frame information for the annotated nominalised verbs, and link them to the event frames in the same way as for verbs. Further future work will include the investigation automatic or semi-automatic methods of linking together the syntactic subcategorization frames and semantic event frames.

Acknowledgements

The work described in this paper has been funded by the European BOOTStrep project (FP6 - 028099). The National Centre for Text Mining is sponsored by the JISC/BBSRC/EPSRC.

References

1. Rebholz-Schuhmann, D., Pezik, P., Lee, V., Kim, J.-J., del Gratta, R., Sasaki, Y., McNaught, J., Montemagni, S., Monachini, M., Calzolari, N., Ananiadou, S.: BioLexicon: Towards a Reference Terminological Resource in the Biomedical Domain. In: Proc. of 16th Ann. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB 2008), Toronto, Canada (2008)
2. Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C., Scheffczyk, J.: FrameNet II: Extended Theory and Practice (2006), `http://framenet.icsi.berkeley.edu/`

3. Palmer, M., Kingsbury, P., Gildea, D.: The Proposition Bank: An Annotated Corpus of Semantic Roles. Computational Linguistics 31(1), 71–106 (2005)
4. Fillmore, C.J.: Frame semantics and the nature of language. In: Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech, vol. 280, pp. 20–32 (1976)
5. Dolbey, A., Ellsworth, M., Scheffczykx, J.: BioFrameNet: A Domain-Specific FrameNet Extension with Links to Biomedical Ontologies. In: Bodenreider, O. (ed.) Proceedings of KR-MED, pp. 87–94 (2006)
6. Wattarujeekrit, T., Shah, P., Collier, N.: PASBio: predicate-argument structures for event extraction in molecular biology. BMC Bioinformatics 5(155) (2004)
7. Browne, A.C., Divita, G., Aronson, A.R., McCray, A.T.: UMLS Language and Vocabulary Tools. In: Proceedings of AMIA Annual Symposium, p. 798 (2003)
8. Tsai, R.T.H., Chou, W.C., Su, Y.S., Lin, Y.C., Sung, C.L., Dai, H.J., Yeh, I.T.H., Ku, W., Sung, T.Y., Hsu, W.L.: BIOSMILE: A semantic role labeling system for biomedical verbs using a maximum-entropy model with automatically generated template features. BMC Bioinformatics 8(325) (2006)
9. Miyao, Y., Ninomiya, T., Tsujii, J.: Corpus-oriented grammar development for acquiring a head-driven phrase structure grammar from the penn treebank. In: Su, K.-Y., Tsujii, J., Lee, J.-H., Kwong, O.Y. (eds.) IJCNLP 2004. LNCS, vol. 3248, pp. 684–693. Springer, Heidelberg (2005)
10. Hara, T., Miyao, Y., Tsujii, J.: Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In: Dale, R., Wong, K.-F., Su, J., Kwong, O.Y. (eds.) IJCNLP 2005. LNCS, vol. 3651, pp. 199–210. Springer, Heidelberg (2005)
11. Thompson, P., Cotter, P., Ananiadou, S., McNaught, J., Montemagni, S., Trabucco, A., Venturi, G.: Building a Bio-Event Annotated Corpus for the Acquisition of Semantic Frames from Biomedical Corpora. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008) (2008)
12. Kipper-Schuler, K.: VerbNet: A broad-coverage, comprehensive verb lexicon. PhD. Thesis. Computer and Information Science Dept., University of Pennsylvania, Philadelphia, PA (2005)
13. Lenci, A., Busa, F., Ruimy, N., Gola, E., Monachini, M., Calzolari, N., Zampolli, A., et al.: SIMPLE Linguistic Specifications LE-SIMPLE (LE4-8346), Deliverable D2.1 & D2.2. ILC and University of Pisa (2000)
14. Montemagni, S., Trabucco, A., Venturi, G., Thompson, P., Cotter, P., Ananiadou, S., McNaught, J., Kim, J.-J., Rebholz-Schuhmann, D., Pezik, P.: Event annotation of domain corpora, BOOTStrep (FP6 – 028099), Deliverable 4.1. University of Manchester, ILC-CNR and European Bioinformatics Institute (2007)
15. Fillmore, C.J.: The case for case. In: Bach, E., Harms, R.T. (eds.) Universals in Linguistic Theory, pp. 1–88. Holt, Rinehart, and Winston, New York (1968)
16. Levin, B., Rappaport Hovav, M.: Lexical Semantics and Syntactic Structure. In: Lappin, S. (ed.) The Handbook of Contemporary Semantic Theory, pp. 487–507. Blackwell, Oxford (1996)
17. Cohen, K.B., Hunter, L.: A critical review of PASBio's argument structures for biomedical verbs. BMC Bioinformatics 7(Suppl. 3), S5 (2006)

TermeX: A Tool for Collocation Extraction

Davor Delač, Zoran Krleža, Jan Šnajder,
Bojana Dalbelo Bašić, and Frane Šarić

Faculty of Electrical Engineering and Computing
University of Zagreb, Croatia
{davor.delac,zoran.krleza,jan.snajder,bojana.dalbelo,frane.saric}@fer.hr

Abstract. Collocations – word combinations occurring together more often than by chance – have a wide range of NLP applications. Many approaches for automating collocation extraction based on lexical association measures have been proposed in the literature. This paper presents *TermeX* – a tool for efficient extraction of collocations based on a variety of association measures. *TermeX* implements POS filtering and lemmatization, and is capable of extracting collocations up to length four. We address trade-offs between high memory consumption and processing speed and propose an efficient implementation. Our implementation allows for processing time linear to corpus size and memory consumption linear to the number of word types.

1 Introduction

Collocations are a lexical phenomenon that has a linguistic and lexicographic status. In [1] collocations are defined as "institutionalized phrases", whereas [2] defines them as "word combinations occurring together more often than by chance." There is a wide range of possible applications for collocation extraction in NLP such as word sense disambiguation [3,4], natural language generation [5], and machine translation [6]. However, collocation extraction is a time consuming task for a human and requires the expertise of a professional lexicographer. Therefore, many approaches for automating collocation extraction have been proposed in the literature.

This paper presents a collocation extraction tool called *TermeX*. This tool is meant for construction of terminology lexica with possible applications in NLP. *TermeX* focuses on the use of lexical association measures (AMs) to automatically extract collocations up to length four (4-grams). It provides the user with a variety of association measures to choose from as well as the ability to manually select valid collocations from those extracted automatically, allowing for construction of domain-specific terminology lexica. Besides English, *TermeX* currently supports Croatian language. In order to improve collocation extraction, *TermeX* implements POS filtering and lemmatization, the latter being important due to morphological complexity of Croatian language. This paper also covers some of the implementation issues such as trade-offs between high memory consumption and processing speed. *TermeX* is able to cope with large amounts

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 149–157, 2009.

of data by relying on optimal data structures that ensure high performance. Another important characteristic of *TermeX* is its platform independence; a version for both Microsoft Windows and Linux operating systems is provided.

The rest of the paper is structured as follows. In the next section we briefly discuss related work. In Section 3 we give an overview of the *TermeX* tool, while in Section 4 we discuss the implementation issues. Section 5 concludes the paper and mentions future work.

2 Related Work

There are several collocation extraction tools available today. *Collocate* [7] is a commercial tool that offers collocation extraction based on Pointwise Mutual Information (PMI) and Log Likelihood association measures. A span of up to twelve words (12-gram) is allowed for PMI, whereas Log Likelihood can be used only to extract 2-grams. *Collocate* provides support for wide range of languages but does not implement lemmatization. It is capable of processing previously POS-tagged corpora.

Collocation Extract [8] is a free tool similar in implementation to *Collocate*. This application also uses AMs for collocation extraction: Log Likelihood, PMI, and Chi-Square. Users can extract collocations in span of two to five words. *Collocation Extract* can process plain-text and XML files, and concordances of the desired collocation can be previewed. Lemmatization and POS filtering is not implemented, thus this tool is not language specific.

TerminologyExtractor [9] is another commercial tool that offers extraction from Word, RTF, HTML, and plain-text documents. *TerminologyExtractor* determines merely frequencies and offers sorting by frequency and alphabetically. It enables the viewing of concordances for a collocation.

Another tool presented in [10] is an advanced collocation extractor designed for computer aided translation. It differs from the aforementioned tools in that it focuses on syntactic analysis combined with AMs. This tool provides support for English and French language.

TermeX differs from the above mentioned tools in that it provides a much wider range of AMs to choose from: as much as fourteen different AMs applicable to extraction of 2-, 3-, 4-grams are provided. Furthermore, *TermeX* outperforms the majority of above mentioned tools in terms of processing speed: using *TermeX*, a corpus of 150 MB can be processed in less than 20 minutes (cf. Section 4.3). Another distinct feature of *TermeX* is its platform independence. Finally, *TermeX* is the first tool for extraction of collocations in Croatian language that provides lemmatization and POS filtering.

3 *TermeX* Tool

In *TermeX*, terminology lexicon is created by selecting collocations from the lists of automatically extracted collocation candidates. Extraction is based on association measures (AMs), statistical measures that provide information on how

likely it is for an n-gram to be a collocation. Therefore, extraction is done by creating sorted lists of n-grams based on their AM value. This way n-grams that are most likely to be a collocation become top ranked. Although this approach seems rather simple, construction of such a list from a large corpus requires substantial amount of time and memory. In order to calculate AM values for n-grams it is necessary to calculate frequencies of lemmas and n-grams up to the desired length. *TermeX* tool effectively solves this problem in time linear to corpus size. Application implements fourteen AMs [11], mostly based on PMI, Dice, and Chi-square test. Implemented measures are extensions of the corresponding bigram measures for n-grams spanning up to four words.

TermeX provides some extra features for collocation extraction. It implements lemmatization and POS filtering, making this application somewhat language specific. Two languages are currently supported: English and Croatian. Application can support other languages provided it has a suitable lemmatization dictionary and a list of abbreviations and stop-words (e.g., determiners, cardinal numbers, prepositions, conjunctions, and pronouns). We chose, however, not to implement POS tagging to make the whole process of collocation extraction as fast as possible; our alternative to POS tagging is described in Section 4.1.

Terminology lexica are created through *TermeX* front-end (GUI). In order to make processing of data using *TermeX* easier, work is organized into projects. A project represents construction of a single terminology lexicon. The graphical user interface is composed of four sections: a menu bar, a toolbar, main working

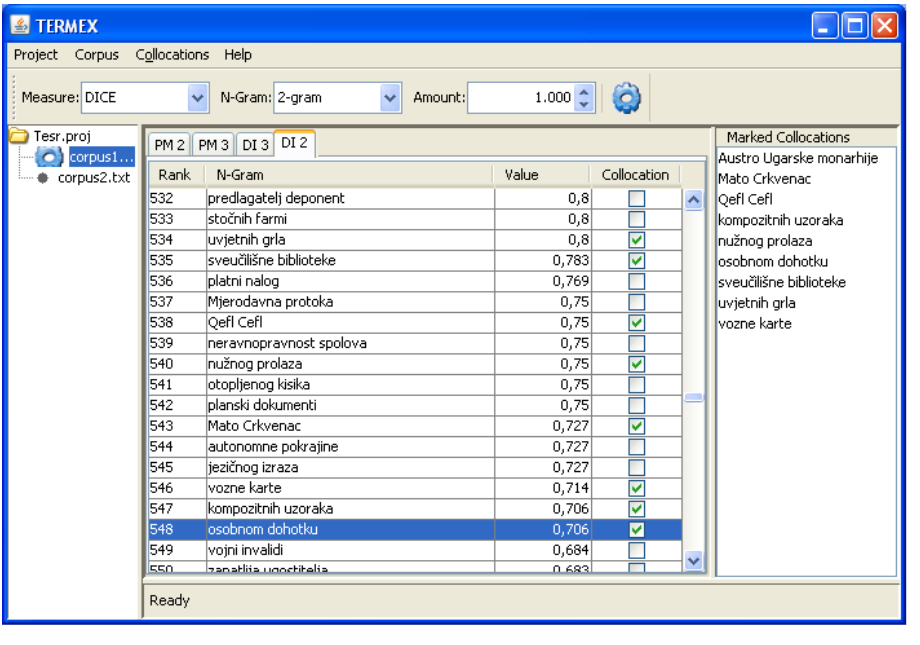


Fig. 1. Graphical user interface

area, and a status bar. Figure 1 shows a project named `TestProjekt`, which uses two corpora, `out1.txt` and `out2.txt`, for construction of a terminology lexicon. *TermeX* currently supports UTF-8 formatted plain-text corpora; support for other formats is planned for future versions.

4 Implementation

The main issue when designing an extraction tool such as *TermeX* is how to meet the demands for optimal speed and memory consumption. In particular, this is important in *TermeX* because of the implemented AMs and maximum collocation length: some of the implemented measures require information such as the frequencies of sub-sequences of an n-gram. For instance, frequencies of 2-grams may be required to calculate the AM value of a 4-gram. Additional requirement is to create a high-quality user interface that is both user-friendly and meets all the demands for this type of tool. Implementation consists of two parts: a back-end written in C++ and a GUI front-end written in Java. The back-end is a console application that takes a plain-text corpus as input and counts the occurrences of lemmas, 2-, 3-, and 4-grams. Based on these frequencies it also computes AMs for n-grams. Because of the amount of data used in these calculations and the time required to make them, C++ was the optimal choice for this part of the implementation. The main reason for implementing the front-end in Java is its portability and easy-to use Swing API. The front- and the back-end are connected using pipelines (Fig. 2). Piping is done by Java Process API, thus avoiding the need for OS specific function calls.

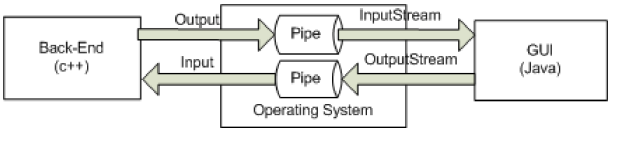


Fig. 2. Communication between front-end and back-end

This section is divided into three subsections: first subsection describes the implementation of the back-end, second the implementation of the front-end, and the last one gives application's performance results.

4.1 Back-End

In order to calculate AMs according to [11], it is necessary to determine frequencies of lemmas and n-grams from a corpus. This is done in three steps: document tokenization, lemmatization, and counting occurrences of n-grams. To this end, we use hash table and vector data structures; an example is given in Fig. 3.

Document tokenization is the process of representing text document as a vector of tokens (words and symbols). Document is read into memory and send to tokenizer generated using Lex [12]. Every word-form and punctuation token is

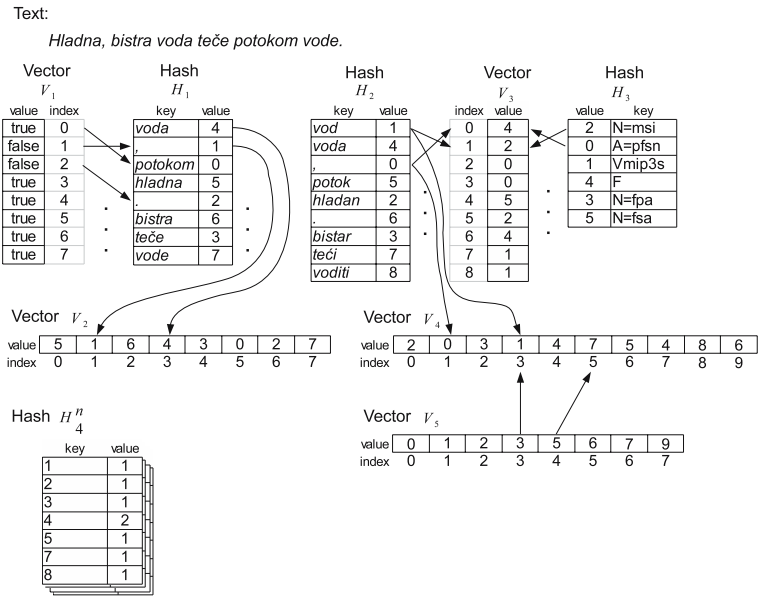


Fig. 3. Back-end data structures

inserted into hash table H_1 and assigned a unique type identifier. In order to save memory, in subsequent steps we use these identifiers instead of the string themselves. Tokenizer checks for each type whether it is a valid word-form (consists of letters only) and stores this information in a vector V_1 of Boolean values at position corresponding to this type's identifier from H_1. After tokenization, entire corpus is represented by a vector V_2 of type identifiers.

Document tokenization is followed by lemmatization of all word-form tokens in vector V_2. Lemmatization effectively conflates the inflectional forms of a token into a single representative form. This way the extraction process counts all inflectional variants of a single lemma type with a counter belonging to that lemma's identifier. Because POS tagging is not used, lemmatization of ambiguous word-forms results in a number of lemma variants, and in this case all the corresponding counters are incremented. Lemmatizer uses a lemmatization dictionary in a form of finite state transducer, making lemmatization fast and memory efficient (e.g., 1 MB for a dictionary containing 70,000 lemmas). For each word-form, lemmatizer returns a pair (*lemma*, *MSD*). In case of Croatian language, MSD is a morphosyntactic descriptor based on the MULTEXT-East Specification [13], encoding compactly the values of morphosyntactic attributes in a single string. Lemma types are stored in a second hash table H_2, while hash table H_3 contains MSD strings as keys and their unique identifiers as values. Vector V_3 binds lemma identifiers to MSD identifiers. Lemmatizer also generates a vector of Boolean values similar to V_1 that indicates which token is a stop-word. Then a vector V_4 of lemma identifiers representing the entire corpus is constructed. Since ambiguous

lemma tokens can be associated with a number of variants, an additional vector V_5 is used to mark the position of each first variant in vector V_4.

In the last step we count the occurrences of n-grams in vector V_4. This vector is scanned token by token for possible collocations as follows. A heuristic approach to segment the sentences and respect sentence boundaries is used. Valid 1-, 2-, 3-, 4-grams, with respect to current token, are inserted into hash tables H_4^1, H_4^2, H_4^3, and H_4^4, respectively. Valid n-gram is a sequence of n valid word tokens, and only a valid sequence can be considered a collocation. Hash tables H_4^n contains identifiers of valid n-grams and their frequency counters. Frequency counter of each n-gram is initially set to one and incremented each time a valid n-gram variant occurs in vector V_4. If lemmatization of an n-gram is ambiguous, counters for all its variants are incremented. Finally, the POS filter determines if an n-gram is a possible collocation. Collocation's byte offset in corpus is stored in another set of hash tables similar to H_4^n.

AMs are calculated according to predefined formulae from [11]. Each AM is calculated by a separate function using hash tables H_4^n with frequencies. A function returns a vector of pairs, each consisting of n-gram identifier and the corresponding AM values, sorted by AM value.

4.2 Front-End

Graphical user interface is implemented according to the Model-View-Controller (MVC) design pattern [14]. The design differs from the classic MVC pattern because the model and the controller are both implemented as part of the data model (Fig. 4); data model is implemented partially in `TermeXModel` class and partially in `Project` class. `TermeXModel` class is in charge of communication with views by creating change events upon data change.

The `Project` class is responsible for creating back-end processes as instances of `java.lang.Process` using the Java Runtime API [15]. This way the Java Process API can manage input and output streams of the back-end process. Data gathered

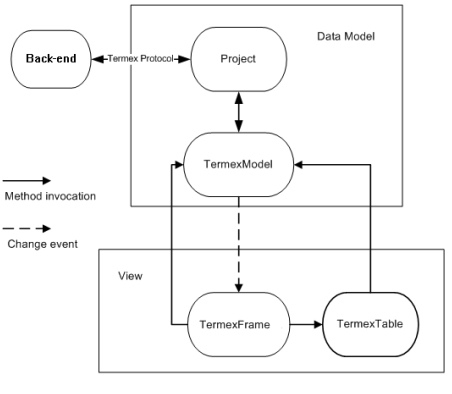


Fig. 4. Architecture of the TermeX tool

from the back-end process is stored as a `TreeSet`. The `TreeSet` data structure is defined as a part of the Java Collections API [15], which makes sure that objects are stored sorted. The view is implemented using Java Swing API.

Communication between a back-end process and graphical user interface relies on a custom made protocol. It is a simple text-based protocol defined for collocation data transfer between the front- and the back-end. It consists of four message types: query, data, status, and break messages. When a front-end creates a new back-end process, the back-end sends back four messages. First three are break messages and the last one is a status message, each showing the current progress of extraction. When a front-end makes a query to the back-end, it sends a query message. Back-end makes a list of query results and starts sending data messages for query results. When the transfer is complete, the back-end sends a break message. If the back-end receives a break message, it frees allocated memory and closes.

4.3 Performance Results

In order to assess the performance of the *TermeX* tool, experiment were performed to establish empirical complexity estimates. Experiments were performed on a corpus consisting of documents from the Official Gazette of the Republic of Croatia published from 1990 onwards. The corpus totals over 25 million words, which roughly corresponds to 170 MB. In order to evaluate how the size of the corpus affects tool performance, we created subsets of this corpus in sizes of 1 MB, 5 MB, 10 MB, and 50 MB; these subsets were created by copying the needed amount of text from the beginning of the large corpus.

Table 1. Performance results

Corpus size			Number of n-gram types			Time		
Bytes	Tokens	Types	2-gram	3-gram	4-gram	Corpus	Query	Memory
1 MB	133,768	14,501	47,700	69,760	77,924	7”	1”	11 MB
5 MB	661,760	40,734	190,786	318,587	373,178	33”	3”	34 MB
10 MB	1,327,385	66,792	361,278	624,147	736,951	1’04”	6”	64 MB
50 MB	6,676,844	206,677	1,349,254	2,596,034	3,194,694	4’54”	27”	261 MB
170 MB	25,288,620	460,855	3,585,258	7,736,772	9,958,635	19’01”	1’28”	490 MB

The experimental results are given in Table 1. Table shows corpus processing time, time needed to calculate AM values (query time), and memory consumption with respect to corpus size. Query time is the time required to determine AM value of all n-grams of selected length and produce a ranked list. The AM used in these experiments is PMI for bigrams. Testing was done on an Intel Pentium IV 3.2 GHz machine with 1 GB RAM running Microsoft Windows XP.

The test results given in Table 1 support the claim that *TermeX* extracts collocations in a time linear to corpus size. Results also reveal that the memory consumption is linear to the number of types.

5 Conclusion

TermeX is a collocation extraction tool for the construction of terminology lexica with possible in NLP. Extraction process is based on fourteen different association measures applicable to n-grams up to length four. Lemmatization and POS filtering allow *TermeX* to better cope with morphological complexity of natural languages, thus making it possible to gain new insights into language-specific aspects of collocational analysis. *TermeX* currently supports lemmatization of Croatian and English language.

The tool is optimized in terms of both processing speed and memory consumption. Algorithms with complexity linear to corpus size ensure fast and efficient extraction process. In addition to that, the user-friendly graphical interface makes processing of results and lexicon creation easier. A demo version of *TermeX* is available at `textmining.zemris.fer.hr/termex`. As part of future work, we intend to provide support for the processing of corpora in XML and XCES format, as well as previously POS-tagged corpora.

Acknowledgments

This work has been jointly supported by the Ministry of Science, Education and Sports, Republic of Croatia and Government of Flanders under the grants 036-1300646-1986 and KRO/009/06 (CADIAL).

References

1. Sag, I.A., Baldwin, T., Bond, F., Copestake, A., Flickinger, D.: Multiword expressions: A pain in the neck for NLP. In: Gelbukh, A. (ed.) CICLing 2002. LNCS, vol. 2276, p. 1. Springer, Heidelberg (2002)
2. Benson, M.: Collocations and general-purpose dictionaries. International Journal of Lexicography 3(1), 23–35 (1990)
3. Yarowsky, D.: One sense per collocation. In: Proceedings of ARPA Human Language Technology Workshop (1993)
4. Mihalcea, R., Faruque, E.: Senselearner: Minimally supervised word sense disambiguation for all words in open text. In: Proceedings of ACL/SIGLEX Senseval (2004)
5. McCardell Doerr, R.: A lexical semantic and statistical approach to lexical collocation extraction for natural language generation. AI Magazine 16, 105 (1995)
6. Orilac, B., Dillinger, M.: Collocation extraction for machine translation. In: Machine Translation Summit IX, pp. 292–298 (2003)
7. Barlow, M.: Collocate 1.0: Locating collocations and terminology. TX: Athelstan (2004)
8. Aroonmanakun, W.: Collocation extract (2000), `http://pioneer.chula.ac.th/~awirote/colloc/`
9. Chamblon Systems, Inc.: Terminology extractor (2004), `http://www.chamblon.com/terminologyextractor.htm`

10. Seretan, V., Nerima, L., Wehrli, E.: A tool for multi-word collocation extraction and visualization in multilingual corpora. In: Proceedings of the 11th EURALEX International Congress (2004)
11. Petrović, S., Šnajder, J., Dalbelo Bašić, B.: Extending lexical association measures for collocation extraction. Computer, Speech and Language (submitted, 2008)
12. Lesk, M.E.: Lex – a lexical analyzer generator. Technical report, AT&T Bell Laboratories, Murray Hill, New Jersey (1975)
13. Erjavec, T., Krstev, C., Petkevič, V., Simov, K., Tadić, M., Vitas, D.: The MULTEXT-East morphosyntactic specifications for Slavic languages. In: Proceedings of the EACL 2003 Workshop on Morphological Processing of Slavic Languages (2003)
14. Gamma, E., Helm, R., Johnson, R., Vlissides, J.M.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional Computer Series, Reading (2000)
15. Eckel, B.: Thinking in Java, 4th edn. Prentice-Hall, Englewood Cliffs (2003)

Guessers for Finite-State Transducer Lexicons

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, names or compounds of such words. In order to add new words to a lexicon, we need to indicate their inflectional paradigm. We present a new generally applicable method for creating an entry generator, i.e. a paradigm guesser, for finite-state transducer lexicons. As a guesser tends to produce numerous suggestions, it is important that the correct suggestions be among the first few candidates. We prove some formal properties of the method and evaluate it on Finnish, English and Swedish full-scale transducer lexicons. We use the open-source *Helsinki Finite-State Technology* [1] to create finite-state transducer lexicons from existing lexical resources and automatically derive guessers for unknown words. The method has a recall of 82-87 % and a precision of 71-76 % for the three test languages. The model needs no external corpus and can therefore serve as a baseline.

1 Introduction

New words and new usages of old words are constantly finding their way into daily language use. This is particularly prominent in rapidly developing domains such as biomedicine and technology. The new words are typically acronyms, technical terminology, loan words, names or compounds of such words. They are likely to be unknown by most hand-made morphological analyzers. In some applications, hand-made guessers are used for covering the low-frequency vocabulary or the strings are simply added as such.

Mikheev [2] and [16] noted that words unknown to the lexicon present a substantial problem to part-of-speech tagging and he presented a very effective supervised method for inducing a guesser from a lexicon and an independent training corpus. Oflazer & al. [3] presented an interactive method for learning morphologies and pointed out that an important issue in the wholesale acquisition of open-class items is that of determining which paradigm a given citation form belongs to.

Recently, 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 Wicentowski [4] and Goldsmith [5]. If we do not need a full analysis, but only wish to segment the words into morph-like units, we can use segmentation methods like Morfessor [6]. For a comparison of some recent successful segmentation methods, see the Morpho Challenge [7].

Although unsupervised methods have advantages for less-studied languages, for the well-established languages, we have access to fair amounts of lexical training

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 158–169, 2009.

material in the form of analyzes in the context of more frequent words. Especially for Germanic and Fenno-Ugric languages, there are already large-vocabulary descriptions available and new words tend to be compounds of acronyms and loan words with existing words. In English, compound words are written separately or the junction is indicated with a hyphen, but in other Germanic languages and in the Fenno-Ugric languages, there is usually no word boundary indicator within the compounds. It has previously been shown by Lindén [8] that already training sets as small as 5000 inflected word forms and their manually determined base forms will give a reasonable result for guessing base forms of new words by analogy, which was tested on a set of languages from different language families. In addition, there are a host of large but shallow hand-made morphological descriptions available, e.g., the Ispell collection of dictionaries [9] for spell-checking purposes, and many well-documented morphological analyzers are commercially available, e.g. [10].

In this paper, we propose a new method that takes an existing finite-state transducer lexicon and creates a guesser using only generally applicable formal properties of weighted transducers. The method is implemented using the open-source *Helsinki Finite-State Technology* [1]. In Section 2, we describe the methodology and present some formal properties. In Section 3, we present the training and test data. In Section 4, we evaluate the model on Finnish, English and Swedish transducer lexicons. In Section 5, we discuss the method and the test results in light of previous literature on guessers.

2 Methodology

Assume that we have a finite-state transducer lexicon $\boldsymbol{T}$ which relates base forms, $\boldsymbol{b(w)}$, to inflected words, $\boldsymbol{w}$. Let $\boldsymbol{w}$ belong to the input language $\boldsymbol{L_I}$ and $\boldsymbol{b(w)}$ to the output language $\boldsymbol{L_O}$ of the transducer lexicon. Our goal is to create a guesser for inflected words that are unknown to the lexicon, i.e. we wish to provide the most likely base forms $\boldsymbol{b(u)}$ for an unknown input word $\boldsymbol{u} \notin \boldsymbol{L_I}$.

In 2.1, we describe the theoretical foundation of the guesser model and, in 2.2, we prove some of the fundamental properties of the guesser model.

2.1 Guesser Model

In order to create a guesser, we first define the left quotient and the weighted universal language with regard to a lexical transducer. For a general introduction to automata theory and weighted transducers, see e.g. Sakarovitch [23].

If $\boldsymbol{L_1}$ and $\boldsymbol{L_2}$ are formal languages, the *left quotient* of $\boldsymbol{L_1}$ with regard to $\boldsymbol{L_2}$ is the language consisting of strings $\boldsymbol{w}$ such that $\boldsymbol{xw}$ is in $\boldsymbol{L_1}$ for some string $\boldsymbol{x}$ in $\boldsymbol{L_2}$. In symbols, we write the left quotient as:

$$\boldsymbol{L_1 \setminus L_2 = \{\, a \mid \exists x\, ((\, x \in L_2\,) \wedge (\, xa \in L_1\,))\,\}} \tag{1}$$

We can regard the left quotient as the set of postfixes that complete words from $\boldsymbol{L_2}$, such that the resulting word is in $\boldsymbol{L_1}$.

If $\boldsymbol{L}$ is a formal language with alphabet $\boldsymbol{\Sigma}$, a *universal language,* $\boldsymbol{U}$, is a language consisting of strings in $\boldsymbol{\Sigma^*}$. The *weighted universal language*, $\boldsymbol{W}$, is a language

consisting of strings in Σ^* with weights $p(w)$ assigned to each string. For our purposes, we define the weight $p(w)$ to be proportional to the length of w. We define a weighted universal language as:

$$W = \{ w \mid \exists w (w \in \Sigma^*) \} \text{ with weights } p(w)=C \cdot |w| , \quad (2)$$

where C is a constant.

A finite-state transducer lexicon, T, is a formal language relating the input language L_I to the output language L_O. The pair alphabet of T is the set of input and output symbol pairs related by T. An identity pair relates a symbol to itself.

We create a guesser, G, for the lexicon T by constructing the weighted universal language W for identity pairs based on the alphabet of L_I concatenating it with the left quotient of T for the universal language U of the pair alphabet of T:

$$G(T) = W\ T \setminus U \quad (3)$$

2.2 Properties of the Guesser Model

Lemma 1. For the lexicon, T, a guesser, $G(T)$, composed with an unknown word, u, generates the entry guesses $b(u) = y\ b(w)$, where $b(w)$ is a postfix of L_O and w is a postfix of L_I.

Proof. Assume that we have an unknown word $u \in \Sigma^*$, where Σ is the input alphabet of T. We decompose u into $y\ w$, such that $y \in \Sigma^*$ and $w \in \{ s \mid \exists p ((p \in L_I) \wedge (ps \in \Sigma^*)) \}$. We then have $u \circ G(T) = (y\ w) \circ (W\ T \setminus U) = (y \circ W)(w \circ T \setminus U) = y\ b(w)$. □

Lemma 2. For the weight-free transducer, T, the entry guesses with the minimal weight, $b_{min}(u)$, for an unknown word, u, composed with the guesser, $G(T)$, is generated by the set of longest matching postfixes of u and the input language of T.

Proof. Assume that we have an unknown word $u \in \Sigma^*$, where Σ is the input alphabet of T. The $b(u)$ with minimal weight is $b_{min}(u) = arg\ min_{p(v)} \{ v \mid v = u \circ G(T) \} = arg\ min_{p(v)} \{ v \mid v = y\ b(w) \}$. The weight of $y\ b(w)$ is proportional to the length of y, i.e. $|y| * C$. □

Lemma 3. If **T** is a weighted transducer, the guesses with minimal weight, $b_{min}(u)$, are the longest matching postfixes of u with minimal weight by T provided that the weight C of the symbols in the universal language W is greater than the weight of any symbol pair related by T.

Proof. Assume that we have an unknown word $u \in \Sigma^*$, where Σ is the input alphabet of T. We decompose u into $y_1\ w_1$ and $y_2\ w_2$, such that $y_1, y_2 \in \Sigma^*$ and $w_1, w_2 \in \{ s \mid \exists p ((p \in L_I) \wedge (ps \in \Sigma^*)) \}$ and $| y_1 | = | y_2 | - 1$. As $| w_1 | = | w_2 | + 1$, we have $p(b(w_1)) \geq p(b(w_2)) + C$ and consequently $p(y_1\ b(w_1)) \geq p(y_2\ b(w_2))$. □

Theorem. To create a longest matching postfix guesser, $G(T) = W\ T \setminus U$, from the weighted lexical transducer, T, we take the maximum transition weight, ω, of T and assign the prefix transition weight C to $\omega+\delta$.

Proof. The result follows directly from Lemma 3. □

For prefixing languages, we can create a guesser using the right quotient and the universal postfix. For circumfixing languages, we can concatenate the prefixing and postfixing guessers to create a circumfixing guesser.

Generally, one can characterize our weighted finite-state entry generator as inducing an ordering over the possible entries for a new and previously unseen inflected form preferring entries that have inflected forms and parts of the stem in common with previously seen entries. As a corollary, entries for already seen words will be generated first. If the forms of the lexical transducer, ***T***, are weighted according to the frequency of the paradigms in the lexicon, the most frequent paradigms are generated first if there are several paradigm candidates for the same affix.

3 Data Sets

To test the entry generator for finite-state transducer lexicons, we created transducer lexicons from existing lexical resources for three different languages: Finnish, Swedish and English using the *Helsinki Finite-State Technology* [1]. We drew words unknown to these lexicons from three language-specific text collections and manually determined their correct entries. In 3.1, we describe the lexical resources and outline the procedure for creating the finite-state transducer lexicon. In 3.2, we describe the test data and, in 3.3, we describe the evaluation method and characterize the baselines.

3.1 Lexical Data for Finite-State Transducer Lexicons

The lexical descriptions relate base forms to inflected word forms. This can be done either through each base form classified with a paradigm and a list of paradigms with model words, or it can be done as a full-form lexical description with all the inflected forms of each base form. The final lexicon and is implemented with finite-state transducer technology. Regardless of the initial form of the lexical description, the finite-state transducer lexicon maps a word in dictionary form to all of its inflected forms. For an introduction, see e.g. Koskennimiemi [11]. Essentially this means that composing the transducer lexicon with an inflected word form will create a new transducer containing all the possible base forms and the morphological analyses of how the inflected word form is related to the base form.

A weighted finite-state transducer lexicon can contain weights in many different ways. A fruitful set of weights would be to estimate the relative frequency of the word forms and encode them as a priori probabilities or weights in the lexicon. This requires a disambiguated corpus. Above we only have lexical descriptions and, assuming that there are or we have created inflectional paradigms, we can estimate the relative frequency of the paradigms. It has also been demonstrated by Karlsson [12] that it is preferable to have as few parts as possible in a multipart compound analysis. For lack of better estimates, the weighted finite-state transducer lexicon lists the analyses primarily according to the number of analyzed compound parts and secondarily in paradigm frequency order.

Most languages have ready-made inflectional paradigms with the lexical description. From this a finite-state transducer lexicon can be manually compiled. However, for languages which typically have few inflected forms for each base form, it is

feasible to have a full-form description of all the lexical entries. If we only have a full-form lexical description, we need to extract paradigms, in order to be able to generate lexical entries for new words.

Finnish. In order to create the Finnish dictionary, we used the Finnish word list *Nykysuomen sanalista* [13], which contains 94 110 words in base form. Of these, approximately 43 000 are non-compound base forms classified with paradigm information. The word list consists of words in citation form annotated with paradigm and gradation pattern. There are 78 paradigms with 13 gradation patterns. For example, the entry for *käsi* (= 'hand') is 'käsi 27' referring to paradigm 27 without gradation, whereas the word *pato* (= 'dam') is given as 'pato 1F' indicating paradigm 1 with gradation pattern F. From this description a lexical transducer is compiled with a cascade of finite-state operations [22]. For nominal paradigms, inflection includes case inflection, possessive suffixes and clitics creating more than 2 000 word forms for each nominal. For the verbal inflection, all tenses, moods and personal forms are counted as inflections, as well as all infinitives and participles and their corresponding nominal forms creating more than 10 000 forms for each verb. In addition, the Finnish lexical transducer also covers nominal compounding.

English. For English we use *FreeLing 2.1* [14]. The FreeLing English lexical resource was automatically extracted from WSJ, with manual post-editing and completion. It contains about 55 000 forms corresponding to some 40 000 different combinations of lemma and part-of-speech. For each part-of-speech, English only has a small set of forms for phonological or semantic reasons, but most often due to the fact that the form did not occur in the Brown corpus.

We extract paradigms from the full-form lexical description for English in the following manner: we automatically align the characters of the base form and the inflected forms and determine the longest common prefix for the base form and all the inflected forms. The remaining set of endings, possibly with some characters from the stem, is considered a paradigm. Since the words may have individual patterns with left out forms, the automatically extracted set of paradigms becomes relatively large. We get 489 paradigms for English out of which 151 occur more than once.

Swedish. For Swedish we use the open source full-form dictionary *Den stora svenska ordlistan* [15]. For each base form, the part of speech is given. For each part-of-speech, there is a given set of inflected forms, e.g. for nouns there are always eight forms, i.e. all combinations of singular and plural, nominative and genitive, definite and indefinite forms. For any word, there may be an empty slot, if the form is considered non-existent for some reason, e.g. phonologically or semantically. In addition, each word may have an indication of whether it can take part in compounding which is prolific in Swedish.

We use the same procedure for creating paradigms for Swedish as we used for English. We get 1333 paradigms out of which 544 occur more than once with the rest in a Zipfian distribution.

3.2 Test Data

A set of previously unseen words in inflected form serve as a test set for which we wish to determine their inflectional paradigm. In order to extract word forms that

represent relatively infrequent and previously unseen words we used various text collections for Finnish, Swedish and English. We drew 5000 word and base form pairs at random from the frequency rank 100 001-300 000 as test material for each language. Since we are interested in new words, we only counted inflected forms that were not recognized by the lexical transducers we had created. In addition, we removed strings containing numbers, punctuation characters or only upper case from the test data.

Finnish. 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. 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].

Of the selected strings, 1715 represented words not previously seen by the lexical transducer. For these strings, correct entries were created manually. Of these, only 48 strings had a verb form reading. The rest were noun or adjective readings. Only 43 had more than one possible reading.

A sample of test strings are: *ulkoasultaan, kilpailulainsäädännön, epätasa-arvoa, euromaan, työvoimapolitiikka, pariskunnasta, vastalausemyrskyn, kolmeentoista, haudatut, liioitellun, ruuanlaiton, valtaannousun, suurtapahtumaan, ostamiaan, …*

English. 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/]. The tokens consisted of 266 000 forms of 175 000 base forms.

Of the selected strings, 3100 represented words not previously seen by the lexical transducer. For these strings, correct entries were created manually for the first 25 %, i.e. 775 new entries. Of these, 60 strings had verb form readings, 610 noun readings and 161 adjective readings, and 14 adverb readings. Only 79 strings had more than one reading.

A sample of test strings are: *florin, disfranchised, chimney-pieces, Beechwood, warbled, sureness, sitting-rooms, marmoset, landscape-painter, half-burnt, Burlington, …*

Swedish. 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 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].

Of the selected strings, 1756 represented words not previously seen by the lexical transducer. For these strings, correct entries were created manually for first 25 %, i.e. 439 new entries. Of these, 37 strings had a verb form reading, 387 noun readings, 47 adjective readings. Only 48 strings had more than one reading.

A sample of the test strings are: *finrummet, chansons, översvämmande, Valören, tonsiller, Stollans, sjöfartspolitiska, reliken. oskött. Dylikt, antidopingkommitté, …*

3.3 Evaluation Measures and Baseline

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. The F-score is the harmonic mean of the precision and the recall.

The random baseline for Finnish is that the correct entry is one out of the 78 paradigms with one out of 13 gradations, i.e. a random correct guess would on the average end up in as guess number 507. For English, an average random guess ends up in position 245 and, for Swedish, in position 667.

4 Experiments

We test how well the guesser outlined in Section 2 is able to predict the paradigm for an inflected word form using the test data mentioned in Section 3. Of the randomly chosen strings from the test data range, word forms representing previously unseen words were used as test data in the experiment. The generated entries are intended for human post-processing, so the first correct entry suggestion should be among the top 6 candidates, otherwise the ranking is considered a failure. All the guessers were statistically highly significantly better than their random baseline.

4.1 Finnish Guesser

The Finnish Guesser generated a correct entry among the top 6 candidates for 82 % of the test data as shown in Table 1, which corresponds to an average position of 2.3 for the first correct entry with 82 % recall and 76 % average precision.

Table 1. Ranks of all the first correct entries by the Finnish guesser

Rank	Freq	Percentage
#1	1140	66,5 %
#2	186	10,8 %
#3	64	3,7 %
#4	17	1,0 %
#5	4	0,2 %
#6	2	0,1 %
#7-∞	302	17,6 %
Total	1715	100,0 %

Table 2. Ranks of all the first correct entries by the English guesser

Rank	Freq	Percentage
#1	477	61,5 %
#2	81	10,5 %
#3	56	7,2 %
#4	17	2,2 %
#5	14	1,8 %
#6	15	1,9 %
#7-∞	115	14,8 %
Total	775	100,0 %

Table 3. Ranks of all the first correct entries by the Swedish guesser

Rank	Freq	Percentage
#1	243	55,4 %
#2	84	19,1 %
#3	40	9,1 %
#4	10	2,3 %
#5	5	1,1 %
#6	1	0,2 %
#N-∞	56	12,8 %
Total	439	100,0 %

4.2 English Guesser

The English Guesser generated a correct entry among the top 6 candidates for 83 % of the test data as shown in Table 2, which corresponds to an average position of 2.4 for the first correct entry with 83 % recall and 72 % average precision.

4.3 Swedish Guesser

The Swedish Guesser generated a correct entry among the top 6 candidates for 87 % of the test data as shown in Table 3, which corresponds to an average position of 2.3 for the first correct entry with 87 % recall and 71 % average precision.

5 Discussion

In this section, we give a brief overview of previous and related work on guessers. In 5.1, we compare test results with previous efforts. In 5.2, we give some notes on the implementation of the methods. In 5.3, we discuss future work.

5.1 Comparison with Results from Similar Efforts

Test results on identical data are not available, but similar efforts have been made and some insights can be gleaned from a comparison between them.

Stroppa and Yvon [17] present experimental results obtained on a morphological analysis task guessing base form and morphological features for an inflected form in English with the following recall and precision: nouns 75 % and 95 %; verbs 95 % and 97 %; adjectives 28 % and 88 %, respectively. It is interesting to note that verb forms are the easiest to get right, whereas it is much trickier to guess the base forms and syntactic features of nouns and adjectives. The explanation is probably that the base forms of nouns and adjectives are much more varied, and that they partly overlap with the inflected forms.

Wicentowski [18] presents the WordFrame model, a noise-robust supervised algorithm capable of inducing morphological analyses for languages which exhibit prefixation, suffixation, and internal vowel shifts. In combination with a naive approach to suffix-based morphology, this algorithm is shown to be remarkably effective across a broad range of languages, including those exhibiting infixation and partial reduplication. Results are presented for over 30 languages with a median accuracy of 97.5 % on test sets including both regular and irregular *verbal* inflections. The excellent accuracy is partly explained by the fact that he uses a dictionary to filter the suggested base forms. His intention is to learn irregular forms which are dominant among verbal inflections, but the good results should be seen in light of the results from Yvon and Stroppa [17], where a substantial challenge seems to be in modeling the behavior of nouns and adjectives. They are also the most frequent categories among new words.

Claveau and L'Homme [19] label morphologically related words with their semantic relations using morphological prefix and postfix analogies learned from a sample of pre-labeled words with a recall of 72 % and precision of 65 % on separate test data.

Baldwin [20] acquires affix and prefix transformations achieving 0.6 F-score for English using Timbl [21] as the classifier. However, the classification was for syntactic features not for inflectional paradigm.

We recall that our model is developed for guessing the paradigms of unknown and previously unseen inflected words, i.e. their base forms cannot be tested against a lexicon. In light of the results from comparable reports from other languages, our results automatically derived guessers are very good, because the data shows that the final guessers have 68-73 % precision and 82-87 % recall, i.e. an F-score of 78-79 %, on all three languages with different morphological complexity. It is interesting that our model is slightly more precise for Finnish, which is morphologically more complex than Swedish and English, whereas the recall is lower for Finnish. The explanation may be that inflected forms of Finnish are better indicators of the paradigm to which they belong, if the ending is recognized. In English, word endings may occur both in inflected and in base forms, e.g. 'sleeping' should be regarded as an adjective in base form in 'a sleeping beauty', but as an inflected form of the verb 'sleep' in 'is sleeping'.

A quick look at the words which fail for English reveals that among them are e.g. *preacheth, Surmountheth, corruptehth,* which could not have received a correct guess as out-dated verb forms were not available as analogical models. Other words with missing correct analogues are *webbed,* which gets the base form *webb* whereas we

expect *web*, even if the word is otherwise correctly identified as a verb. Similar problems seem to afflict words ending in low frequency characters in combination with the fact that we require the correct answer to have a specific base form and a paradigm which indicates all the correct inflected forms. E.g., we require that words like *plowman* are correctly identified as having the plural *plowmen.* It is not enough just to identify it is as some noun. The same goes for other words with irregular forms or exceptional paradigms. This also demonstrates that for part-of-speech tagging, the guessing task is easier as the tagging does not require guessing all the correct forms and only the correct forms of an out-of-vocabulary word.

Table 4. Test results for Finnish, English and Swedish guessers

Language	Recall	Precision	F-score
Finnish	0,82	0,76	0,79
English	0,83	0,72	0,78
Swedish	0,87	0,71	0,78

5.2 Implementation Note

The models were implemented with a cascade of weighted finite-state transducers. For conveniently creating morphological analyzers and guessers, HFST–the Helsinki Finite-State Technology [1] is available as an Open Source toolkit. Running the guesser in forward mode may be relatively slow, whereas running the guesser in reverse is almost deterministic for the n-best results and therefore very efficient.

5.3 Future Work

The suggested model is completely general and requires no additional data except the morphological analyzer in finite-state transducer format. It would be interesting to see, whether this general model can benefit from a purely probabilistic model conditioned on analogical transformations, e.g. the one suggested by Lindén [8], or some more contextually oriented model taking the surrounding words into account.

6 Conclusion

A substantial amount of languages have been implemented as lexical transducers with the Koskenniemi two-level model or similar formalisms, which means that there is a wealth of lexical transducers available. As the entry generator model we suggest is general and requires only a lexical transducer and no additional information from external corpora, it can serve as the baseline for entry generators on a number of languages. Compared with guessers for part-of-speech tagging, the entry guessing task is more difficult as entry guessing requires all the correct forms and only the correct forms of an out-of-vocabulary word to be identified. We have tested our entry guesser on inflected forms of new words in three languages from different language families demonstrating that the model has a recall of 82-87 % and a precision of 71-76 % for

the three test languages. This corresponds to having the first correct entry on the average in position 2.3-2.4.

Acknowledgements. We are grateful to Tommi Pirinen and Anssi Yli-Jyrä for fruitful discussions and to the Finnish Academy for funding the research.

References

1. HFST–Helsinki Finite-State Technology, http://www.ling.helsinki.fi/kieliteknologia/tutkimus/hfst/index.shtml
2. Mikheev, A.: Unsupervised Learning of Word-Category Guessing Rules. In: Proc. of the 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996), pp. 327–334 (1996)
3. Oflazer, K., Nirenburg, S., McShane, M.: Bootstrapping Morphological Analyzers by Combining Human Elicitation and Machine Learning. Comp. Ling. 27(1), 59–85 (2001)
4. Wicentowski, R.: Modeling and Learning Multilingual Inflectional Morphology in a Minimally Supervised Framework. PhD Thesis, Baltimore, USA (2002)
5. Goldsmith, J.A.: Morphological Analogy: Only a Beginning (2007), http://hum.uchicago.edu/~jagoldsm/Papers/analogy.pdf
6. Creutz, M., Hirsimäki, T., Kurimo, M., Puurula, A., Pylkkönen, J., Siivola, V., Varjokallio, M., Arisoy, E., Saraçlar, M., Stolcke, A.: Morph-based speech recognition and modeling of out-of-vocabulary words across languages. ACM Trans. on Speech and Lang. Proc. 5(1), art. 3 (2007)
7. Kurimo, M., Creutz, M., Turunen, V.: Overview of Morpho Challenge in CLEF 2007. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152, pp. 19–21. Springer, Heidelberg (2008)
8. Lindén, K.: A probabilistic model for guessing base forms of new words by analogy. In: Gelbukh, A. (ed.) CICLing 2008. LNCS, vol. 4919, pp. 106–116. Springer, Heidelberg (2008)
9. Kuenning, G.: Dictionaries for International Ispell (2007), http://www.lasr.cs.ucla.edu/geoff/ispell-dictionaries.html
10. Lingsoft, Inc.: Demos, http://www.lingsoft.fi/?doc_id=107&lang=en
11. Koskenniemi, K.: Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production. Department of General Linguistics, University of Helsinki, Publication No. 11 (1983)
12. Karlsson, F.: SWETWOL: A Comprehensive Morphological Analyser for Swedish. Nordic Journal of Linguistics 15(1), 1–45 (1992)
13. Nykysuomen sanalista, http://kaino.kotus.fi/sanat/nykysuomi/
14. FreeLing 2.1–An Open Source Suite of Language Analyzers, http://garraf.epsevg.upc.es/freeling/
15. Westerberg, T.: Den stora svenska ordlistan (2008), http://www.dsso.se/
16. Mikheev, A.: Automatic Rule Induction for Unknown-Word Guessing. Comp. Ling. 23(3), 405–423 (1997)
17. Stroppa, N., Yvon, F.: An Analogical Learner for Morphological Analysis. In: Proc. of the 9th Conference on Computational Natural Language Learning (CoNLL), pp. 120–127 (2005)

18. Wicentowski, R.: Multilingual Noise-Robust Supervised Morphological Analysis using the WordFrame Model. In: Proc. of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology, ACL, pp. 70–77 (2004)
19. Claveau, V., L'Homme, M.C.: Structuring Terminology using Analogy-Based Machine Learning. In: Proceedings of the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005, pp. 17–18 (2005)
20. Baldwin, T.: Bootstrapping Deep Lexical Resources: Resources for Courses. In: Proc. of the ACL-SIGLEX Workshop on Deep Lexical Acquisition, ACL, pp. 67–76 (2005)
21. Daelemans, W., Zavrel, J., Sloot, K., Bosch, A.: TiMBL: Tilburg Memory-Based Learner, version 6.0, Reference Guide', Technical Report–ILK07-03, Department of Communication and Information Sciences, Tilburg University (2003)
22. Pirinen, T.: Open Source Morphology for Finnish using Finite-State Methods (in Finnish). Technical Report. Department of Linguistics, University of Helsinki (2008)
23. Sakarovitch, J.: Éléments de théorie des automates. Vuibert (2003)

Combining Language Modeling and Discriminative Classification for Word Segmentation

Dekang Lin

Google, Inc.
1600 Amphitheater Parkway, Mountain View, CA, USA, 94043
lindek@google.com

Abstract. Generative language modeling and discriminative classification are two main techniques for Chinese word segmentation. Most previous methods have adopted one of the techniques. We present a hybrid model that combines the disambiguation power of language modeling and the ability of discriminative classifiers to deal with out-of-vocabulary words. We show that the combined model achieves 9% error reduction over the discriminative classifier alone.

Keywords: Segmentation, Maximum Entropy, Language Model.

1 Introduction

The problem of word segmentation is to identify word boundaries in languages, such as Chinese, Japanese and Thai, where such boundaries are not explicitly marked with white spaces. Word segmentation is the first step in processing the text in these languages and its quality often affects all downstream components.

The main two challenges for word segmentation are the resolution of ambiguities and the handling of out-of-vocabulary (OOV) words.

Ambiguity is pervasive in word segmentation. Consider the following fragment of a Chinese sentence:

…中国家鼓励… (in … the country encourages …) (1)

The correct segmentation is:

中	国家	鼓励
in	country	encourages

However, another (incorrect) candidate is:

中国	家	鼓励
China	home	encourages

An obvious way to resolve the ambiguities is through language modeling. The best segmentation of an input sentence $C_1^n = C_1C_2...C_n$ is the one where the resulting sequence of words has the highest probability among all word sequences that constitute the character sequence. In other words, the best segmentation is

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 170–182, 2009.

$$\arg\max_{yield(W_1^m)=C_1^n} P(W_1^m) \tag{2}$$

where $yield(W_1^m)$ is the concatenation of the characters in the word sequence.

Although a language model helps resolve segmentation ambiguities, the search for the best segmentation is limited to sequences of known words. The out-of-vocabulary (OOV) words are segmented into smaller units or missegmented with adjacent characters. To deal with the OOV problem, language model based segmenters typically rely on manually created pattern matching rules to recognize names, or heuristic post-processing rules to combine sequences of single characters.

Word segmentation can also be treated as a tagging problem (Xue and Shen, 2003; Peng *et. al.* 2004; Tseng *et al.* 2005; Low *et al.* 2005). Each character in the input sentence is assigned a tag. The best segmentation is determined by

$$\arg\max_{T_1^n} P(T_1^n | C_1^n) \tag{3}$$

where T_i is a tag that marks whether the character C_i is the beginning of a word. In the literature, two types of tags have been proposed. One is a 2-tag set: *b* for beginning of a word, and *c* for continuation of a word. The other is a 4-tag set:

s: the character is a word itself.
b: the character is the first character of a word.
e: the character is the last character of a word.
m: the character is in the middle of a word.

The probability in (3) is typically computed by a discriminative classifier trained with the maximum entropy and conditional random fields.

The word boundaries determined by the tags do not have to agree with any vocabulary, which means that the segmenter is able to produce words that it has never seen before. This turns out to be both a blessing and a curse. It is a blessing because such a segmenter combines new word recognition and segmentation in a single unified process. For example, the machine learned classifier may pick up the fact that the character 者 (suffix -er) is typically the last character in a word, and recognize购彩者 (lotto-ticket buyer) as a single word even if the word is never seen in the training corpus.

Not having to agree with a vocabulary may also be a curse because the segmenter may make a sequence of tagging decisions that are individually quite reasonable, but globally inconsistent. Consider the example in (1). Since 中国 (China) is a common word, the classifier is likely to say there is no boundary between中 and 国. Since国家 (country) is also very common, it decides that there is no boundary between 国 and 家 either. As a result, the segmenter incorrectly outputs 中国家as a single word. One might hope that models such as Maximum Entropy Markov Mode (MEMM) or Conditional Random Fields (CRF) (Lafferty *et al.* 2000) will be able to use the preceding tags as features to ensure the global consistency of a tag sequence. However, such hope gets quickly shattered when one realizes that the conditional distribution of tags given a tag n-gram is not nearly as sharp as the tag distribution given the observations. This phenomenon was first noted by Klein and Manning (2002) in part-of-speech tagging.

They call it the **observation bias**. In word segmentation, the bias is even stronger, because there are only 2-4 types of tags.

In this paper, we present a word segmenter that combines a discriminative classifier with a language model. It is constructed by first training the discriminative classifier with a manually segmented corpus and then building an n-gram language model smoothed with distributionally similar words. The word similarities are obtained from an unlabeled corpus segmented with the discriminative classifier.

The remainder of the paper is organized as follows. The next section gives an overview of our model. We then provide details of the discriminative classifier and the language model augmentation in Sections 3 and 4, respectively. After presenting the experimental results in Section 5, we discuss the related work in Section 6.

2 A Probabilistic Segmentation Model

Our segmentation model combines a discriminative classifier and a generative language model. Theoretically, (3) is equivalent to

$$\arg\max_{yield(W_1^m)=C_1^n} P(W_1^m | C_1^n) \tag{4}$$

The reason is that once the tags are known, the word sequence is determined. In practice, however, C_1^n is first converted into features in order for the probability computation to be feasible. The features are represented as a set of atomic identifiers. The probability models, such as Maximum Entropy or CRF classifier, do not really have access to the character sequence C_1^n. We propose a different formulation of the word segmentation problem. To segment a sentence is to find

$$\arg\max_{yield(W_1^m)=C_1^n} P(W_1^m | F_1^n) \tag{5}$$

where F_1^n is a sequence of feature vectors, each of which corresponds to a character in the input sentence. The probability $P(W_1^m | F_1^n)$ is computed as follows:

$$P(W_1^m | F_1^n) = P(W_1^m, T_1^n | F_1^n) \tag{6}$$

$$= P(W_1^m | T_1^n, F_1^n) \times P(T_1^n | F_1^n) \tag{7}$$

$$\approx P(W_1^m | T_1^n) \times P(T_1^n | F_1^n) \tag{8}$$

$$\approx \prod_{i=1}^{m} P(W_i | W_{i-k}^{i-1}) \times \prod_{i=1}^{n} P(T_i | F_i) \tag{9}$$

The equality in 0 holds because the tag sequence T_1^n is logically implied by the word sequence W_1^n. The approximation in 0 is based on the assumption that if tags are known, the feature vectors are conditionally independent of the word sequence. In step 0 we assume that the probabilities of tags are independent of each other (except that their corresponding feature vectors are correlated) and the probability of a word

given all previous words is equal to the probability of the word given its previous k–1 words.

Therefore, our segmentation model is a combination of a generative k-gram language model $P\left(W_i \middle| W_{i-k}^{i-1}\right)$ and a discriminative classifier $P\left(T_i \middle| F_i\right)$.

The discriminative classifier can be used as a segmenter by itself. We will refer to it as **M1**. We first train M1 with a maximum entropy learner using a segmented corpus. We then use M1 to segment an unlabeled corpus to acquire OOV words and build a language model. The combination of the language model and the maximum entropy classifier results in a better segmenter, which will be referred to as **M2**.

3 M1: A Discriminative Classifier

Our word-boundary classifier is similar to the maximum entropy segmenter in (Low *et al.*, 2005). For each character in the segmented training corpus, we extract a training example consisting of a label (one of *s*, *b*, *e*, or *m*) and a feature vector.

3.1 Features

The features are extracted using the following templates (the numbers –1, 0, 1 refer to the previous, the current and the next character position respectively):

C_i (i = –1, 0, 1): the character unigrams
C_iC_{i+1} (–1, 0): the character bigrams

W_{kw}: true when a known word w is found in the input sentence and the position of the current character is k relative to w. The value of k ranges from –1 to |w|, the number of characters in w. When k = –1, the word w is immediately after the current character. When k = |w|, w immediately precedes the current character. Otherwise, the current character is part of w.

For example, the following features are extracted for the character 家 in a sentence fragment 中国家鼓励.

$C_{0家}$, $C_{-1国}$, $C_{1鼓}$, $C_{-10国家}$, $C_{01家鼓}$, $W_{-1鼓励}$, $W_{1国家}$, $W_{2中国}$

Our feature set differs from previous approaches in several aspects.

The feature template W includes both overlapping and adjacent word candidates. The overlapping words were used in (Low *et al.*, 2005). None of the previous segmenters seems to have used the adjacent words as features.

We extracted character bigrams as features from the 2 character window [–1, 1]. Most other systems used the window [–2, 2]. The narrower window reduces the number of features. Our experiments show that smaller window size does not compromise the accuracy. Note that, with the overlapping and adjacent words, the segmenter does consider characters more than one character away, as long as there exists a known word connecting them to the current character.

The word features in our model are triggered by the presence of a sequence of characters that form a known word. This is different from (Andrew 2006), where the semi-CRF features are keyed on words that are guaranteed to be part of the final

segmentation. For example, in the above example, the presence of the features $W_{1国家}$ and $W_{2中国}$ does not guarantee the words 国家 (country) and中国 (China) to be in the output. In fact, the output may include neither of the words.

3.2 MaxEnt Training

After extracting the tags and feature vectors from a training corpus consisting of segmented sentences, we trained a maximum entropy classifier with the GIS algorithm using exponential prior for regularization (Goodman 2004).

Let (x_j, y_j) denote a training example with feature vector x_j and label y_j, f_i (i=1…F) denote a feature, and $\Lambda = (\lambda_1, \lambda_2, ..., \lambda_F)$ denote the weight vector corresponding to the features. The training algorithm maximizes

$$\arg\max_{\Lambda} \prod_{j=1}^{n} \frac{\exp \sum_{i=1}^{F} \lambda_i f_i(x_j, y_j)}{\sum_{y'} \exp \sum_{i=1}^{F} \lambda_i f_i(x_j, y')} \times \prod_{i=1}^{F} \alpha_i \exp(-\alpha_i \lambda_i)$$

subject to the constraint that

$\lambda_i = 0$ and expected[f_i] ≥ observed[f_i] – α_i, or

$\lambda_i > 0$ and expected[f_i] = observed[f_i] – α_i.

We found that the performance of the trained segmenter is not very sensitive to the value of α_i as long as it is within (0, 1). We therefore used the same value, 0.1, for all features in all of our experiments. The number of training iterations was fixed at 300 in all experiments as well.

Unlike (Low *et al.*, 2005), we did not set a minimum threshold for feature counts. All the features extracted from the training examples are used. A nice property of the exponential prior, as pointed out by (Goodman, 2004), is that the weights of many features are 0 after training and can be discarded. The MaxEnt training algorithm is therefore capable of doing feature selection by itself.

3.3 Consistent Tag Sequences

A problem with the *s-b-e-m* tag set is that not all tag sequences are consistent. For example, *s* cannot be followed by *e* or *m*. The issue was addressed in (Xue and Shen 2003) by applying transformation-based learning to the output tag sequences. Another solution in (Low *et al.*, 2005) is to restrict the word lengths to a predefined number, such as 20, and then use a dynamic programming algorithm to find most probably tag sequences with consistent tags.

We adopted a dynamic programming solution that is more principled (no predefined limit) than (Low *et al.*, 2005). We represent each possible tag for each character as a node in a Markov network. The emission probability of a node is the probability of the tag obtained from the MaxEnt classifier. The transition probabilities are all 1s.

However, transitions only exist if the tag combination is valid. For example, the *s* node of a character has only transitions to the *s* and *b* nodes of the next character. The most probable consistent tag sequence can then be obtained with the Viterbi algorithm.

The resulting MaxEnt classifier can be used as a segmenter itself. Even though it doesn't resort to special word lists for recognizing names or OOV words and doesn't have any post processing as many of the top systems in the Chinese Segmentation Bakeoff, its performance is surprisingly good, in fact, better than or equal to any system in the Second Chinese Segmentation Bakeoff on all 4 data sets tested there (see Section 5 for details).

4 M2 = M1 + Language Modeling

The segmenter M2 is built by adding a language model to M1. There are several options for building the language model component. One is to collect the n-gram statistics from the segmented training corpus. The problem is that the segmented corpus is small and the n-gram statistics is consequently very sparse. Another option is to run the segmenter on a large corpus and gather the statistics from the automatically segmented corpus. This type of self-training has been tried on many other NLP tasks, including parsing (Charniak 1997) and part of speech tagging (Clark *et al.*, 2003). The results were mostly disappointing, with the exception of (McClosky *et. al.* 2006). The self-trained system often perpetuates and sometimes amplifies the errors in the original system. We propose a third alternative, which is to collect n-gram counts from the manually segmented corpus and smoothing the counts with distributional word similarities obtained from an unsegmented corpus. The advantage of this is that we are leveraging on the second order regularities in the corpus. Even if there are systematic errors in the n-gram counts, there has to be systematic such errors to cause the distributional similarity to be erroneous.

4.1 Distributional Similarity

Distributional word similarities are computed under the assumption that words that occur in similar contexts tend to have similar meanings (Hindle 1990; Dagan *et al.*, 1997; Lin 98). We represent the contexts of words as feature vectors. We define a feature *f* of a word *w* to be a word that occurred next to *w* on its immediate left or right. The value of the feature *f* is the point-wise mutual information between *f* and the word *w*:

$$PMI(w,f) = \log \frac{P(w,f)}{P(w) \times P(f)} = \log \frac{(|w,f| - d) \times |*,*|}{|w,*| \times |*,f|}$$

where $|w, f|$ is the frequency count of the co-occurrence of *w* and *f*, and * is a wild card, and *d* is a discounting factor (we used $d = 0.9$ in out experiments). The similarity between two words can then be computed by the cosine of the angle between their respective feature vectors.

4.2 OOV Word Extraction

The language model needs to work with a closed vocabulary. The vocabulary of the segmented training corpus is small (the training corpora in the Chinese Segmentation Bakeoff contain 55k-140k unique words). Fortunately, by running the discriminative classifier M1 on a large corpus, we may obtain a large number of 'words' that are not part of the training vocabulary. However, many of the 'words' are segmentation errors. We assume that the vocabulary of the training corpus is sufficiently large so that, for any word, there exists a set of words in the vocabulary that are distributionally similar to it. Based on this assumption, we constructed feature vectors for all words that are sufficiently frequent in the automatically segmented corpus. For each OOV word, we compute its similarity with all of the in-vocabulary words. A word candidate is discarded if it is not distributionally similar (based on a threshold) to any known word.

4.3 Similarity-Based Smoothing

We use the word similarities to generalize the language model constructed with the segmented corpus. Similarity-based smoothing was first proposed in (Dagan *et. al.* 1997), where the bigram probability $P(w_2|w_1)$ is smoothed with:

$$P_{SIM}(w_2|w_1) = \sum_{w_1' \in S(w_1)} \frac{W(w_1, w_1')}{N(w_1)} P(w_2|w_1')$$

where $S(w_1)$ is the set of top similar words of w_1, $W(w_1, w_1')$ is the similarity between w_1 and w_1' and $N(w_1)$ is the normalization factor. The problem with this scheme is that since $P(\bullet|w_1')$ is likely to be a sparse distribution as well. Many of the probabilities in the sum are also 0s. We propose a different solution, by assuming that similar word pairs have similar values of PMI.

$$P(w_2|w_1) = \frac{P(w_1, w_2)}{P(w_1)} = \frac{P(w_1, w_2)}{P(w_1)P(w_2)} P(w_2) = \exp(PMI(w_1, w_2))P(w_2)$$

$$\approx \exp\left(\sum_{w_1' \in S(w_1), w_2' \in S(w_2)} \frac{W(w_1, w_2, w_1', w_2')}{N(w_1, w_2)} PMI(w_1', w_2') \right) P(w_2)$$

where $W(w_1, w_2, w_1', w_2')$ is the similarity between two pairs of words (w_1, w_2) and (w_1', w_2'), which is compute as the geometric average of the similarities between corresponding words. The advantage of this smoothing scheme is that either or both of w_1 and w_2 may be substituted with their similar words.

4.4 Combining LM with MaxEnt

The segmenter M2 employs a dynamic programming algorithm to find the best sequence of words that maximizes the combined probabilities of the language model and the tag sequence probability. The algorithm constructs a Markov network. Each

node in the network is a pair (w, l), where w is a word known to the language model and l is a label which is either B or C. When the label l is B (begin), w starts a new segment. When the label is C (continuation), w is appended to the end of a previous segment. The best segment sequence is obtained by finding a path with highest probability from the beginning to the end of the sentence. Each B-labeled node and zero or more C-labeled nodes that follow it form a segment. A segment with a single B node corresponds to a know word in the language model. Other segments are OOV words.

The emission probability of a node (w, l) is the product of the labels of the characters in w. If the label is B, the characters in w are labeled as s or $bm…e$, depending on the number of characters in w. If the label is C, the characters in w are labeled e or $m…e$.

There is a transition from (w_1, l_1) to (w_2, l_2) if w_1 is immediately followed by w_2 in the input. The probability of the transition is computed as follows:

- If $l_1 = B$ and $l_2 = B$, the transition probability is bigram probability $P(w_2 \mid w_1)$ computed by the language model.
- If $l_1 = C$ and $l_2 = B$, the transition probability is unigram probability $P(w_2)$.
- If $l_1 = B$ and $l_2 = C$, w_2 is appended to w_1 to form a longer segment. The emission probability of (w_1, l_1) was computed under the assumption that the last character of w_1 is tagged as e or s. When w_2 is appended to w_1, the tag of the last character of w_1 needs to be changed to m or b. We therefore compute the transition probability by multiplying $P(w_2)$ with the ratio between the probabilities of last character of w being m and being e (or being b and being s).
- If $l_1 = C$ and $l_2 = C$, the tag of the last character of w_1 needs to be changed from e to m. The transition probability is the unigram probability $P(w_2)$ multiplied with the ratio between the probabilities of the last character of w_1 being m and being e.

5 Experimental Results

5.1 Data Sets

There is no universally accepted standard for Chinese segmentation. The Second Chinese Segmenter Bakeoff (Emerson 2005) tested the systems with 4 sets of data, each of which was created according to its own guidelines. The systems are evaluated by the F-measure of how well they retrieve the gold standard segments in the test set.

Table 1 summarizes the general characteristics of the data sets. Each data set consists of a training set and a test set. The baseline is obtained by maximal matching using only the words from the training data. It can be seen that the OOV rate directly correlates with baseline F-measure.

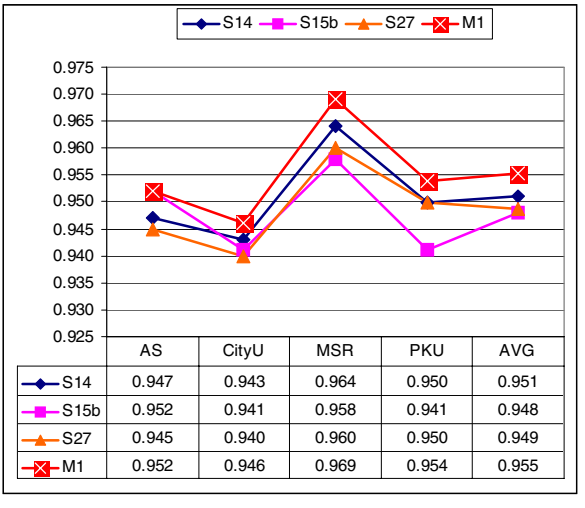
Table 1. Data Sets in the Second Chinese Segmentation Bakeoff

Data Set	Source	Chinese	Training (words)	Test (words)	OOV rate	Baseline
AS	Academia Sinica	trad.	5.45M	122K	4.3%	0.882
CityU	City University	trad.	1.46M	41K	7.4%	0.833
MSR	Microsoft	simp.	2.37M	107K	2.6%	0.933
PKU	Peking University	simp.	1.10M	104K	5.8%	0.869

Table 2. Comparison with alternative designs

	AS	CityU	MSR	PKU	AVG
M1	0.952	0.946	0.969	0.954	0.955
5-char	0.952	0.947	0.968	0.952	0.955
2-tag	0.945	0.936	0.962	0.946	0.947

There were two tracks in the Chinese Segmentation Bakeoff: closed and open. The closed track restricts the systems to use only the training data set provided to all the participants. In the open track, systems are allowed to include their own resources. Our segmenter M1 qualifies as a closed track system. We compare it with closed track systems.



	AS	CityU	MSR	PKU	AVG
S14	0.947	0.943	0.964	0.950	0.951
S15b	0.952	0.941	0.958	0.941	0.948
S27	0.945	0.940	0.960	0.950	0.949
M1	0.952	0.946	0.969	0.954	0.955

Fig. 1. Comparison with closed track winners

5.2 Evaluation of M1

Figure 1 compares M1 against the three systems with a highest score in at least one of the data sets. S14, S15b and S27 are identifiers assigned to the systems (Emerson 2005). M1 performed significantly better than the best systems in the closed track on three out of the four data sets and tied for the forth one. On average, M1 obtained an error reduction[1] of 8% compared to the S14, which had the highest average F-score among the closed track participants. This is a little surprising because:

- M1 extracts features from a narrower window than most other systems;
- M1 does not involve any language-specific features. In contrast, the top systems all included rules or features that are keyed on special character sets extracted from

[1] We use the term loosely by treating 1-F as the error rate.

the training data, e.g., the characters that tend to be suffixes or prefixes, or the characters that tend to be surname or given names.

- M1 does not include any post processing, as many top systems.
- The machine learning algorithm in M1 is simpler (MaxEnt vs. CRF) and is faster to train.

Since many other systems used a 5-character window for extracting features and 2-tag set (*b* and *c*), instead of the 4-tag set (*s-b-e-m*), Table 2 shows the F-scores of these alternatives. Expanding the feature window from 3 characters to 5 characters essentially has no impact on the F-score. The downside is that the learned models become twice as big (see Table 3) and are computationally more costly to run. The 2-tag models are smaller. The smaller size, unfortunately, comes with a big drop in F-score. Table 3 also included the number parameters in trained CRF models in (Tseng *et al.,* 2005). They are generally more than twice as big as M1 models.

Table 3. Number of parameters (in millions)

	AS	CityU	MSR	PKU
M1	7.1	3.2	4.2	2.5
5-char	14.2	6.4	8.4	4.9
2-tag	3.0	1.3	1.7	1.0
CRF	8.1	7.1	12.5	5.4

Table 4. OOV word extraction

Words	AS	CityU	MSR	PKU
candidates	497745	547143	2593349	2115040
results	170434	175855	661576	549242

5.3 Evaluation of M2

The language models in M2 are constructed by segmenting a 5GB traditional Chinese news corpus (for AS and CityU) and a 50GB simplified Chinese news corpus (for MSR and PKU) with a discriminatively trained M1. The 'candidates' row in Table 4 contains the numbers of words in the M1-segmented corpus with a minimum frequency count of 10. The numbers of words that passed the distributional similarity filter are listed in the 'results' row. The M1-segmenters trained with AS and MSR produced more words because these two corpora treat the concatenation of a family and a given name as a single segment.

Table 5 compares M2 against M1. $\mathrm{ER}_{\mathrm{M1}}^{\mathrm{M2}}$ is the relative error reduction of M2 over M1. The introduction of a language model gets a relative error reduction of 9% on average. The F-score increases on all 4 data sets, although the increase is not statistically significant with the AS data. The no-sim row contains the results obtained by using all new word candidates obtained by segmenting the unlabeled data. The average F-score remain unchanged compared to M1. The tinyLM row shows the F-scores obtained if we use the segmented training set to build an n-gram language model

without similarity based smoothing. The average F-score decreases significantly. The drop is rather dramatic for the AS data. This, we believe, is because the language model information in the training data may have already been captured by the MaxEnt model. Another way to view this is that having a separate language model component is a good thing. Even if the discriminative classifier can capture some of the language model information, it can only do so for word sequences that are present in the manually segmented corpus. Having a language model component allows us to obtain this information from much larger (practically unlimited) amount of data.

Table 5. Results with language modeling

	AS	CityU	MSR	PKU	AVG
M2	0.953	0.954	0.972	0.958	0.959
M1	0.952	0.946	0.969	0.954	0.955
ER_{M1}^{M2}	2%	15%	10%	9%	9%
no-sim	0.952	0.951	0.970	0.956	0.957
tinyLM	0.897	0.946	0.965	0.949	0.939

The rankings of M2 would have been 2nd (AS), 2nd (CityU), 1st (MSR) and 10th (PKU) in the open track of the Second Chinese Segmentation Bakeoff. A notable difference between M2 and the open track systems is that the only additional resource in M2 is an unsegmented corpus, whereas others added manually segmented corpora, larger dictionaries, specialized word lists (e.g., punctuation marks, function words of various kinds, last names, country names, title prefixes, units, dates, *etc.*). For example, many systems used a lexicon from PKU[2]. If we include that lexicon in M2 as well, the F-score for PKU data set increases to 0.965 and ranks the 4th.

6 Related Work and Discussion

Many early statistical segmenters are based on language modeling, e.g., (Sproat *et al.*, 1996) and (Luo and Roukos, 1996). More recently, discriminative classification has proven to be a very effective method for this problem. Most top systems in the Chinese Segmenter Bakeoff are based on discriminative classifiers (Low *et al.* 2005; Tseng *et al., 2005*), although the unigram language model system by (Chen *et al.*, 2005) is also one of the strongest contenders. Our experiments show that the *s-b-e-m* tag set works better than the *b-c* tag set. The main difference between them is that the former is able to distinguish single-character words, which include many frequent function words, from others. Our method for ensuring the consistency of the 4-tag-set sequences is more principled than previous proposals.

Generative language models and discriminative classifiers complement each other in segmentation. One is helpful in resolving ambiguities while the other is good at dealing with OOV words. (Gao *et al.* 2005) combines the scores from many

[2] http://ccl.pku.edu.cn/doubtfire/Course/Chinese%20Information%20Processing/Source_Code/Chapter_8/Lexicon_full_2000.zip

components, including a language model, as features in a linear classifier. Their language model consists of a sequence of *word classes*, such as person names, locations, *etc.*, whereas we model sequences of words directly. The word-class sequence model is also used in (Asahari *et al.,* 2003). They then employ a SVM classifier to make corrections to the LM segmenter.

Another hybrid model is the semi-Markov CRF (Andrew 2006) where features may be defined in terms bigrams of words, instead of character sequences. Such features, however, are restricted to the labeled training set, whereas our language model is built from a much larger unlabeled data. The semi-Markov CRF segmenter in (Andrew 2006) was evaluated with the MSR data set. Its F-score is 0.968, which is slightly below our M1's 0.969 and significantly lower than M2's 0.972.

7 Conclusion

Previously word segmenters mostly rely on one of the language modeling and discriminative classification. We presented a segmenter that combines both. Our experimental results showed that the combined model achieves 9% error reduction over the discriminative classifier alone.

References

1. Andrew, G.: A hybrid Markov/Semi-Markov conditional random field for sequence segmentation. In: Proc. of EMNLP 2006 (2006)
2. Asahara, M., Goh, C., Wang, X., Matsumoto, Y.: Combining segmenter and chunker for Chinese word segmentation. In: Proc. of the Second SIGHAN Workshop on Chinese Language Processing, pp. 144–147 (2003)
3. Charniak, E.: Statistical parsing with a context-free grammar and word statistics. In: Proc. of AAAI 1997 (1997)
4. Clark, S., Curran, J., Osborne, M.: Bootstraping POS-taggers using unlabelled data. In: Proc. of CoNLL 2003 (2003)
5. Chen, Y., Zhou, A., Zhang, G.: Unigram Language Model for Chinese Word Segmentation. In: Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing (2005)
6. Dagan, Lee, L., Pereira, F.C.N.: Similarity based methods for word sense disambiguation. In: Proc. of ACL 1997 (1997)
7. Emerson, T.: The Second International Chinese Word Segmentation Bakeoff. In: Proc. SIGHAN Workshop on Chinese Language Processing (2005)
8. Gao, J., Li, M., Wu, A., Huang, C.N.: Chinese Word Segmentation and Named Entity Recognition: A Pragmatic Approach. Computational Linguistics 31(4), 531–574 (2005)
9. Goodman, J.: Exponential Priors for Maximum Entropy Models. In: Proceedings of HLT/NAACL (2004)
10. Hindle, D.: Noun classification from predicate-argument structures. In: Proc. of ACL 1990 (1990)
11. Klein, D., Manning, C.: A Generative constituent-context model for improved grammar induction. In: Proceedings of the 40th Annual Meeting of the ACL (2002)

12. Low, J.K., Ng, H.T., Guo, W.: A maximum entropy approach to Chinese word segmentation. In: Proc. SIGHAN Workshop on Chinese Language Processing (2005)
13. Lin, D.: Automatic retrieval and clustering of similar words. In: Proc. of COLING/ACL 1998, pp. 768–774 (1998)
14. Luo, X., Roukos, S.: An Iterative Algorithm to Build Chinese Language Models. In: Proc. of ACL 1996, pp. 139–145 (1996)
15. Luo, X.: A maximum entropy Chinese character-based parser. In: Proc. of EMNLP (2003)
16. McClosky, D., Charniak, E., Johnson, M.: Effective self-training for parsing. In: Proc. NAACL 2006 (2006)
17. Peng, F., Feng, F., McCallum, A.: Chinese segmentation and new word detection using conditional random fields. In: Proc. of COLING 2004 (2004)
18. Sproat, R., Gale, W., Shih, C., Chang, N.: A stochastic finite-State word-segmentation algorithm for Chinese. Computational Linguistics 22(3) (1996)
19. Tseng, H., Chang, P., Andrew, G., Jurafsky, D., Manning, C.: A conditional random field word segmenter for SIGHAN Bakeoff 2005. In: Proc. SIGHAN Workshop (2005)
20. Xue, N., Shen, S.: Chinese word segmentation as LMR tagging. In: Proceedings of the Second SIGHAN Workshop, pp. 176–179 (2003)

Formal Grammar for Hispanic Named Entities Analysis

Grettel Barceló, Eduardo Cendejas, Grigori Sidorov, and Igor A. Bolshakov

Centro de Investigación en Computación
Instituto Politécnico Nacional
Mexico City, Mexico
{gbarceloa07,ecendejasa07}@sagitario.cic.ipn.mx,
{sidorov,igor}@cic.ipn.mx

Abstract. A task that has been widely studied in the field of natural language processing is the Named Entity Recognition (NER). A great number of approaches have been developed to deal with the identification and classification of named entity strings in specific- and open-domains. Nevertheless, external modules have to be incorporated into many of the NER systems in order to solve the interpretation problems derived from proper nouns. In this article our focus will be on the study of ambiguity in Hispanic Nominal Sequences which constitution assumes three main problems: (1) the association of given names and/or surnames; (2) the composition of such elements by means of a connector; (3) and the duality of given name/surname. In order to analyze the magnitude of the problem, two gazetteers were made, one with 93998 given names and the other with 13779 surnames. The gazetteers entries were used as terminal symbols of the proposed grammar to determine the valid interpretations in the nominal sequences; this is done by means of an automatic labeling of all the elements the nominal sequences are made of.

1 Introduction

Named entity recognition deals with the identification and the classification of strings that belong to proper nouns. Several categories have been established for the classification process; some of the most important categories are: person, organization and location. NER systems can be used independently [1] or as modules of text pre-processing in different NLP applications where the recognition efficacy impacts directly on its quality. These systems have been put to work in machine translation tasks [2], [3], information extraction [4], [5], [6], information retrieval [7], [8] and question answering [9], [10].

Based on the text analysis of several genres, studies have proved that there is a high percentage in occurrence of proper nouns. Since the origin of NER in 1995, and up to today, several heuristics and algorithms have been proposed, with hand crafted rules or statistical approaches.

However, ambiguity problems are still persistent in the identification as well as in the classification stage. Three main errors are mentioned in [11] that can be found in some categories, such as:

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 183–194, 2009.

- Entity - Noun: When an entity is a noun homograph (e.g., the word *fuente* [source] and the surname *Fuente*)
- Entity boundary detection: To determine where a named entity, that is composed of two or more words, begins and ends and that it can be identified as an independent identity (e.g.,*Juan Pérez* as a full name)
- Entity - Entity: When a string can belong to more than one of the established categories (e.g. *Hidalgo* is a surname and the name of a state)

The focus of our research is the analysis of personal names categories, and to be more specific, of Hispanic personal names.

Because of the structure of Hispanic nominal sequences, they have a high level of ambiguity. Usually, Hispanic names are made up of a given name, and two surnames. The first surname is the same as the first surname the father holds, and the second surname is the same as the first surname the mother holds (the order can be modified by a previous agreement or for reasons unknown by one of the parents, though).

Besides determining the presence of personal entities, our objective is to interpret in a correct manner every single element in all of their acceptable forms in a sequence of names. Thus, for a given sequence, it was necessary to detect where the given name, the paternal and maternal surnames began and ended (boundary detection within the same entity). In this interpretation task, the main difficulties were caused by some of the customs in Spanish-speaking countries that allow compound and/or associated given names and surnames. Another common inconvenience found in these countries is given names that are also used as surnames; this is why errors in entity - entity in this study are also obvious.

An evaluation of the ambiguity level in Hispanic names is presented in this paper by means of the implementation of an established generative grammar. By examining the structure of the officially allowed sequences, the ambiguity producing sources are determined. In order to obtain exploratory quantitative data, a specialized syntactic analyzer, that works with the proposed grammar, was designed. Using the analyzer, 745084 personal entries were extracted from the corpus used for our research, a demographic approximation of a Mexican State.

2 Problem Description

The name of any person is made up of a given name and of one or several surnames, according to the customs of each language and country. The name is given to offspring by parents when they are born or when they are baptized. The surname or the family name goes from one generation to the next; it is usually the father's or the father and the mothers'.

Figure 1 represents, in a generalized manner, the formation of Hispanic full names which is made up of three main elements. From this structure, three main problems arise that bring about ambiguity: (1) given names and/or surnames association; (2) given names and/or surnames composition; and, (3) given name/surname duality.

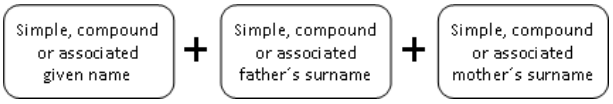


Fig. 1. Hispanic nominal sequence elements

Given names association is manifested by the union of two simple names, such as *Juan Carlos*, *Víctor Manuel*, *Sofía Miranda*, among others. An associated surname is made up by the concatenation of two simple surnames, such as *Alarcón Ruiz*, *Arteaga Jiménez*, etc.

Given names composition arises from the concatenation of simple given names and the preposition "*de*" which is commonly articulated. Examples of composed given names are: *María de los Ángeles*, *María del Carmen*, among others. The composition of surnames is presented by the union of two simple surnames by means of a connector (the preposition "*de*" or the conjunction "*y*"), such as *Montes de Oca* or *Montes y Gómez*.

There are many given names and surnames that pose a duality due to the fact that they can belong to any of the categories, such as *Santiago*, *Alfonso*, etc.

3 Empiric Heuristic

3.1 Entity Recognition and Its Elements

For the entity recognition task applied to the Hispanic nominal system, we rely on three main resources: − a gazetteer of accepted given names, − a gazetteer of permitted surnames and − a simple heuristic to indentify and classify the elements of each nominal sequence (i.e. disambiguation within the same entity)

The manual construction of different semantic classes is a tedious task. For that reason, multiple investigations have been made for the creation of automatic entity lists. Among the proposed algorithms, four approximations stand out: pattern recognition techniques [12], [13], [14] clustering [15], [16], variants of the Hidden Markov Models (HMM) [17], [18] and web page wrappers [14], [19]. Domain-specific extraction rules are applied to pattern recognition techniques. Clustering techniques usually use positioning data bases and resembling matrixes. In the HMM variants, states are organized by regions (one region for every desired class). And last, wrappers extract the elements from localized lists for the final analysis of classes.

For the gazetteers generation we apply the method proposed by Nadeau et al. [11]; an approach of web page wrappers. In this, the possibility of creating correct lists in unsupervised way is explained; the supervision is limited to a seed of four examples. In order to information retrieval from the web, a query was created by conjoining of four generated manual entities (e.g., *Hernández* AND *Martínez* AND *García* AND *Pérez*). The main algorithm concept is to extract additional surnames for each web page that is recovered.

Obtained gazetteers comprise 93998 given names and 13779 surnames, respectively. Even if the symbol list is available, two problems remain in the sequence

elements: (1) boundary detection (due to given names and/or surnames association/composition); and (2) belonging to more than one category (due to given name/surname duality). So, the document scanning strategy and the search of terms that match the dictionary's entries is not enough. A heuristic then was designed, using the following grammar, to deal with the former problems. Obtained gazetteers entries were used as target symbols of the proposed grammar.

Grammar for hispanic named entities analysis

Non target symbols:

- Initial symbol - Hispanic Nominal Sequence (HNS)
- Given Names Sequence (GNS)
- Male Names Sequence (MNS)
- Female Names Sequence (FNS)
- Male Given Name (MGN)
- Female Given Name (FGN)
- Surnames Sequence (SNS)
- Paternal Surname (PSN)
- Maternal Surname (MSN)

Target symbols:

Hispanic alphabet strings (in this case, given names and surnames)

Rewrite rules:

Stage 1

$HNS \rightarrow GNS \mid SNS \mid GNS\ SNS$

$GNS \rightarrow MNS \mid FNS$

Stage 2

$MNS \rightarrow MGN \mid MGN\ MGN \mid MGN\ FGN$

$FNS \rightarrow FGN \mid FGN\ FGN \mid FGN\ MGN$

$SNS \rightarrow PSN \mid PSN\ MSN$

Stage 3

$MGN \rightarrow mgn \in \{Juan, Francisco, Pedro, José, ...\}$

$FGN \rightarrow fgn \in \{Juana, María, Margarita, Rosa, ...\}$

$PSN \rightarrow psn \in sn$

$MSN \rightarrow msn \in sn$

$sn = \{Hernández, Martínez, García, Pérez, ...\}$

3.2 Syntactic Analyzer

The recognition system includes two main stages: the recognition stage (which only identifies certain entities or entity sequences as names). And the classification stage (which assigns labels to this type of entities). In the recognition stage,

the scanning and comparison of the text strings are performed with gazetteers entries. In the classification stage, the maximal sequences of the entities that were recognized as names in the first stage are taken and they are introduced into the syntactic analyzer.

The syntactic analyzer uses the proposed grammar and performs an automatic labeling of all the elements that make up part of the sequence; taking into account during the analysis the three main reasons for ambiguity that the Hispanic denominations may pose.

Let us analyze the result from the analyzer for the nominal Hispanic sequence: *Juana María Pérez Hernández*, after having extracted it from a document.

$$\Big(\big((Juana)_{FGN}, (María)_{FGN}\big)_{FNS}, \big((Pérez)_{PSN}, (Hernández)_{MSN}\big)_{SNS}\Big)_{HNS}$$

This case comes out as an interpretation without ambiguity because of the quantity and nature of the elements that make up the sequence. In Figure 2 the same outcome is presented in a syntactic scheme.

The first stage of the analyzer consists of the delimitation of the elements that make up the nominal sequence (HNS). Three elements can be obtained from it (given name, the paternal surname and the maternal surname). According to the structure proposed in Figure 1, if HNS is formed by more than six elements (associated/compound name + associated/compound paternal surname + associated/compound maternal surname) it is discarded.

Algorithm 1 shows the process of formation of the elements, creating groups to represent the associations and compositions. The groups obtained determine the quantity of elements in the sequence.

Algorithm 1: Groups are made of the elements that make up the nominal sequence

Input: Spanish nominal sequence, HNS
Output: The list of grouped elements, *elements*

```
for each element e in HNS
   add(e, elements) {add e to the list of elements}
size = number of elements in HNS
if size > 6
   {Invalid entry}
else if size > 3
   group = size - 3 {Quantity of groups to have 3 elements}
   {They are grouped making sure that there are in gazetteers}
   for each element e in elements and group do
      if it is in gazetteers ((e - 1) + e) { e - 1 is a previous element}
         e - 1 = e - 1 + e {they concatenate}
         erase(e) {erases e from elements since e - 1 concatenated}
```

All valid combinations can be accomplished with the groups formed in Algorithm 1. Two vectors are used, GNS and SNS, that are marked if the entry element is in the gazetteer of given names, if it is in the gazetteer of surnames or in both (ambiguity of dual given name/surname).

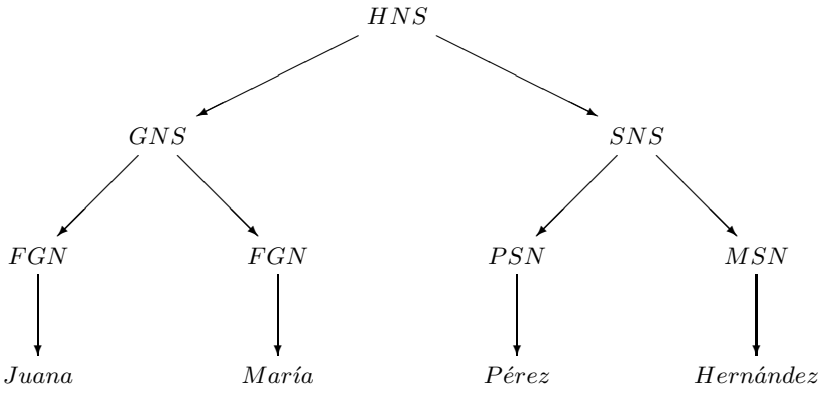


Fig. 2. Syntactic outline of a nominal sequence

Algorithm 2: Analysis of the elements that make up the nominal sequence

Input: The list of the grouped elements, *elements*
Output: Valid interpretations of the sequence

```
intersections = 0;
for each e in elements do
   GNS(e) = add(isGivenName(e))
   SNS(e) = add(isSurname(e))
{If there're more than 2 names(right) the following are marked as false}
for each e in elements do
   if position of e > 1
      GNS(e) = 0
{If there're more than 2 surnames(left)the following are marked as false}
for each e in elements do
   if position of e < size - 2
      SNS(e) = 0
{If there's an e that is neither a name nor a surname it is discarded}
for each e in elements do
   if NOT GNS(e) AND NOT SNS(e)
      {The combination is discarded}
   else if GNS(e) AND SNS(e) {The intersections are counted}
      intersections = intersections + 1
```

Besides verifying that the name exists, the function $isGivenName$ returns the extracted genre from the nomenclature ($M - masculine$, $F - feminine$, $U - undefined$). The function $isSurname$ only indicates if it is registered or not in the list of Hispanic surnames. Finally, with the GNS and SNS sequences and the number of *intersections* it is possible to propose all the valid interpretations for the HNS entry.

A unique interpretation was obtained in the above sequence, given that its elements (in quantity and semantic content), do not generate any problem of

ambiguity. But this does not always occur as such in the nominal sequences extracted from documents written in Spanish.

Now let us examine the sequence: *Carlos Alonso Salvador Ramírez*, for which the analyzer returns the following result.

$$\Big(\big((Carlos)_{MGN},(Alonso)_{MGN}\big)_{GNS},\big((Salvador)_{PSN},(Ramírez)_{MSN}\big)_{SNS}\Big)_{HNS}$$

None of the elements that make up the Hispanic structure (given name, paternal surname, maternal surname) have been omitted from the sequence; thus it has only one interpretation. However, we see what occurs if we omit the maternal surname.

$$\Big(\big((Carlos)_{MGN},(Alonso)_{MGN}\big)_{GNS},\big((Salvador)_{PSN},(\emptyset)_{MSN}\big)_{SNS}\Big)_{HNS}$$
$$\Big(\big((Carlos)_{MGN}\big)_{GNS},\big((Alonso)_{PSN},(Salvador)_{MSN}\big)_{SNS}\Big)_{HNS}$$

The second interpretation that was obtained is the result of the duality in the entity *Alonso*, given that this can be classified as a given name as well as a surname. *Alonso* is an entry in both of the developed gazetteers and cannot be disambiguated by the quantity of elements, as could be done in the complete sequence.

It can be concluded, from this example, that the validity of the omission of some of the elements that make up the nominal sequence, directly impact its interpretation. Nevertheless, this is a very common practice in Hispanic countries.

Lastly, let us analyze what happens if we also discard the given name.

$$\Big(\big((Alonso)_{MGN},(Salvador)_{MGN}\big)_{GNS},\big((\emptyset)_{PSN},(\emptyset)_{MSN}\big)_{SNS}\Big)_{HNS}$$
$$\Big(\big((Alonso)_{MGN}\big)_{GNS},\big((Salvador)_{PSN},(\emptyset)_{MSN}\big)_{SNS}\Big)_{HNS}$$
$$\Big(\big((\emptyset)_{MGN}\big)_{GNS},\big((Alonso)_{PSN},(Salvador)_{MSN}\big)_{SNS}\Big)_{HNS}$$

The analyzer returns three valid interpretations, because *Salvador*, as well as *Alonso* shows up in duality as given name/surname.

4 Results

4.1 Ambiguity Sources in the Corpus

For our analysis we considered a corpus with 745084 people registered as residents in the state of Hidalgo, Mexico, as our source of information. This number represents almost a third (31.77%) of the total population of the region, according to the last census of inhabitants and housing made by the National Institute of Statistics, Geography and Information (INEGI) in 2005. Table 1 shows the sources of ambiguity, in quantity and percentage, for each of the problems being dealt with, where: AGN - associated given names, CGN - compound given names, ASN - associated surnames, CSN - compound surnames and DGN/SN - dual give name/surname.

Table 1. Distributions of the sources that produce ambiguity

AGN	CGN	ASN	CSN	DGN/SN
75581	1581	363	315	2061
80%	2%	3%	2%	15%

4.2 Distribution of Given and Surnames

In order to assign a quantitative measure of ambiguity in respect of duality, the number of occurrences such as given name and surname of each entity, is taken into account. Two influential aspects were considered in the coefficient of ambiguity.

- The normalized distance between frequencies
- The quantity of occurrences in the corpus

The distance between the frequencies is the absolute difference between the number of times that the entity occurs as a given name and the number of times it occurs as a surname. Then for an entity e, the distance is determined thus:

$$d(e) =\mid f_{GN}(e) - f_{SN}(e) \mid$$

f_{GN} y f_{SN} being the frequency with which e appears as a given name and a surname respectively. The proportion in which e occurs is calculated by taking the maximum distance in the corpus as a reference to normalize the distance of e. Thus we obtain a first coefficient of influence in distance (C_{ID})

$$C_{ID}(e) = d(e)/d_{MAX}$$

d_{MAX} being the greatest distance obtained in the corpus.

For example, *Alejo* appears as a given name on 181 occasions and 262 as a surname. Besides, 42171 have been determined to be the maximum distance in the corpus. Therefore, the distance and the coefficient of influence for the *Alejo* entity are:

$$d(Alejo) =\mid 181 - 262 \mid= 81$$
$$C_{ID}(Alejo) = 81/42171 = 0.001921$$

The values of the coefficient C_{ID} describe the proportion in which a given name can also be a surname. This gives us a result between 0 and 1, where 0 produces the greatest influence of ambiguity and 1 the least influence. In this way, if we have two entities, the most ambiguous will be one that holds less value of C_{ID}.

Up to this point, *García* is the least ambiguous entity because the distance was the greatest found in the corpus, i.e.:

$$d(García) = 42171 = d_{MAX} \Rightarrow C_{ID}(García) = 1$$

We say "up to this point" in the previous analysis because the second aspect of influence, the quantity of occurrences in the corpus, must still be considered in the measure of ambiguity.

This aspect is important because it implies that more significant values of coefficient are awarded those entities that occur more frequently in the corpus. Thus, an entity whose number of occurrences may be, let us say, 1000 and 997 for give names and surnames respectively, it will be more ambiguous than an entity with a presence of, let us say, on 5 and 2 occasions. If only the C_{ID} was considered, both entities would possess the same measure of ambiguity ($C_{ID} = 3/42171 = 0.0000711$), being that the first occurred more commonly than the second.

To measure the coefficient of influence in quantity (C_{IC}), next expression has been used:

$$C_{IC}(e) = \frac{\frac{f_{GNMAX} - f_{GN}(e)}{r_{GN}} + \frac{f_{SNMAX} - f_{SN}(e)}{r_{SN}}}{2}$$

Where:

- f_{GNMAX} and f_{SNMAX} represent the maximum frequency of occurrence in given names and surnames respectively and
- r_{GN} y r_{SN} the range of appearance for given names and surnames respectively. The range is determined as the difference between the maximum and the minimum frequency of each set, i.e. $r_{GN} = f_{GNMAX} - f_{GNMIN}$ y $r_{SN} = f_{SNMAX} - f_{SNMIN}$

In general, that which is defined is a transformation to the interval [0,1] that measures the occurrence of an entity in the corpus; this is for the given name and surnames sets. From the formula it can be observed that the same weight is given to both sets. In this case, if we have two entities, the one that will have more occurrences will be the one with a lesser value of C_{IC}. So, the lesser the value of C_{IC}, the more influence it will have in the coefficient of final ambiguity.

According to this coefficient definition, *García* is the most recurrent entity in the corpus since it has the lowest value of C_{IC} (0.4976)

Finally, the ambiguity coefficient (C_A) is determined as the product of both influence coefficients (distance and quantity):

$$C_A(e) = C_{ID}(e) \times C_{IC}(e)$$

As it can be observed in the former expression, both influences have been given the same value over the ambiguity coefficient (even though it can be adjusted). The lesser the value of C_A is, the more ambiguous it becomes to analyze the entity.

In Figure 3, the distribution of ambiguous given names and surnames is shown by concept of duality between each other. Each one of the entities is represented by a point, with its rank in the axis of x and its ambiguity coefficient in y; both axes are logarithmic.

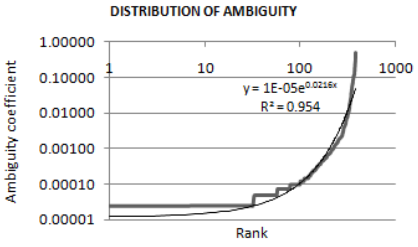


Fig. 3. Distribution of ambiguous given names and surnames by concept of duality

The most ambiguous points are for first two points in the ambiguity rank are for *Ángeles* and *Guadalupe* and the two last points are for *García* and *Pérez*.

4.3 Analysis Consideration

Aspects such as the association of more than two names, hyphenated names, and social reasons were not taken into consideration within this analysis. Socially speaking, it is still very common for wives to adopt the paternal surname of husbands. There is also the lineage phenomenon in which an especially illustrious surname descends from one generation to the next.

Another thing to consider is that the analyses have been performed in a corpus only pertaining to people that live in the state of Hidalgo (Mexico). Thus, the proportions in different states of Mexico, or other countries, can present a slightly different behavior. The reason for this is that given names and surnames are geographically distributed as a consequence of cultural and ancestral dynamics of the regions.

5 Conclusions

In this paper, we introduced a grammar to analyze named entities and specifically the category of Hispanic names. We worked with a syntactic analyzer that scanned 745084 entries so as to evaluate the ambiguity level that nominal sequences posses. In order to extract named entities from the corpus, two gazetteers were used with Hispanic given names and surnames that are allowed.

This study proves the complexity of Hispanic nominations; even when the results do not show the totality of all the diversifications that can be presented. The liberty in Hispanic countries to assign names and the inexistent regulatory bodies are, among others, the reasons for high levels of ambiguity.

The study produced the following results: 33% (241922) of the personal registries analyzed present at least two different valid interpretations; this is the result of having one or more of the ambiguity cases that were studied. 77162 (80%) sources of ambiguity were detected from the analyzed given names and 2739 (20%) from the surnames names examined; a total of 79901 given and surnames.

References

1. Dale, R., Mazur, P.: Handling conjunctions in named entities. In: Gelbukh, A. (ed.) CICLing 2007. LNCS, vol. 4394, pp. 131–142. Springer, Heidelberg (2007)
2. Babych, B., Hartley, A.: Improving machine translation quality with automatic named entity recognition. In: Proceedings of the 7th International EAMT 2003, Budapest, Hungary, pp. 1–8 (2003)
3. Huang, F.: Multilingual Named Entity Extraction and Translation from Text and Speech. PhD thesis, Carnegie Mellon University (2005)
4. Grover, C., Gearailt, D., Karkaletsis, V., Farmakiotou, D., Pazienza, M., Vindigni, M.: Multilingual xml-based named entity recognition. In: Proceedings of the International Conference on Language Resources and Evaluation LREC 2002, pp. 1060–1067 (2002)
5. Yun, B.-H.: HMM-based korean named entity recognition for information extraction. In: Zhang, Z., Siekmann, J.H. (eds.) KSEM 2007. LNCS, vol. 4798, pp. 526–531. Springer, Heidelberg (2007)
6. Ramesh, G.: From named entity recognition and disambiguation to relation extraction – learning for information extraction. In: Proceedings of National Conference on Research Prospects in Knowledge Mining NCKM 2008, Tamil Nadu, India (2008)
7. Paik, W., Liddy, E.D., Yu, E., McKenna, M.: Categorizing and standardizing proper nouns for efficient information retrieval. In: Proceedings of the Workshop on Acquisition of Lexical Knowledge from Text, Ohio, USA, pp. 154–160 (1993)
8. Thompson, P., Dozier, C.: Name searching and information retrieval. In: Proceedings of 2nd Conference on Empirical Methods in Natural Language Processing EMNLP 1997, Rhode Island, USA, pp. 134–140 (1997)
9. Mollá, D., van Zaanen, M., Smith, D.: Named entity recognition for question answering. In: Proceedings of Australasian Language Technology Workshop ALTW 2006, Sydney, Australia, pp. 51–58 (2006)
10. Noguera, E., Toral, A., Llopis, F., Muńoz, R.: Reducing question answering input data using named entity recognition. In: Matoušek, V., Mautner, P., Pavelka, T. (eds.) TSD 2005. LNCS, vol. 3658, pp. 428–434. Springer, Heidelberg (2005)
11. Nadeau, D., Turney, P., Matwin, S.: Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In: Proceedings of the 19th Canadian Conference on Artificial Intelligence CAI 2006, Quebec City, Canada, pp. 266–277 (2006)
12. Riloff, E., Jones, R.: Learning dictionaries for information extraction by multi-level bootstrapping. In: Proceedings of the 16th National Conference on Artificial Intelligence AAAI 1999, Florida, USA, pp. 474–479 (1999)
13. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th International Conference on Computational Linguistics COLING 1992, Nantes, France, pp. 539–545 (1992)
14. Etzioni, O., Cafarella, M., Downey, D., Popescu, A., Shaked, T., Soderl, S., Weld, D., Yates, E.: Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence 165, 91–134 (2005)
15. Lin, D., Pantel, P.: Induction of semantic classes from natural language text. In: Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining SIGKDD 2001, California, USA, pp. 317–322 (2001)

16. Kozareva, Z., Ferreira, J., Gamallo, P., Pereira, G.: Cluster analysis of named entities. In: Proceedings of the International Conference on Intelligent Information Processing and Web Mining IIS: IIPWM 2004, Zakopane, Poland, pp. 429–433 (2004)
17. Kubala, F., Schwartz, R., Stone, R., Weischedel, R.: Named entity extraction from speech. In: Proceedings of DARPA Broadcast News Transcription and Understanding Workshop, Virginia, USA, pp. 287–292 (1998)
18. Cohen, W.: Exploiting dictionaries in named entity extraction: Combining semimarkov extraction processes and data integration methods. In: Proceedings of the 10th ACM Sigkdd International Conference on Knowledge Discovery and Data Mining KDD 2004, Washington, USA, pp. 89–98 (2004)
19. Sigletos, G., Paliouras, G., Spyropoulos, C.D., Hatzopoulos, M.: Mining web sites using wrapper induction, named entities, and post-processing. In: Berendt, B., Hotho, A., Mladenič, D., van Someren, M., Spiliopoulou, M., Stumme, G. (eds.) EWMF 2003. LNCS, vol. 3209, pp. 97–112. Springer, Heidelberg (2004)

Automatic Extraction of Clause Relationships from a Treebank

Oldřich Krůza and Vladislav Kuboň

Charles University in Prague
Institute of Formal and Applied Linguistics
Malostranské nám. 25, Prague
Czech Republic
kruza@ufal.mff.cuni.cz, vk@ufal.mff.cuni.cz

Abstract. The paper concentrates on deriving non-obvious information about clause structure of complex sentences from the Prague Dependency Treebank. Individual clauses and their mutual relationship are not explicitly annotated in the treebank, therefore it was necessary to develop an automatic method transforming the original annotation concentrating on the syntactic role of individual word forms into a scheme describing the relationship between individual clauses. The task is complicated by a certain degree of inconsistency in original annotation with regard to clauses and their structure. The paper describes the method of deriving clause-related information from the existing annotation and its evaluation.

1 Introduction

One of the major factors which changed linguistics during the past twenty years was without a doubt a strong stress on building and exploiting large annotated corpora of natural languages. They serve nowadays as a primary source of evidence for the development and evaluation of linguistic theories and applications.

Although the corpora are extremely important source of data, they are not omnipotent. The more elaborated annotation scheme the authors use, the more problems with linguistic phenomena they have to solve. Creating a consistently annotated treebank requires making a large number of decisions about a particular annotation of particular linguistic phenomena. The more elaborated and detailed is the annotation, the easier it is to find phenomena which are annotated in a seemingly inconsistent way. If the annotation is really well-designed and consistent, then it should be possible to extract an information hidden in the corpus or treebank even in case that a particular phenomenon we are interested in was not annotated explicitly.

This paper describes an attempt to do precisely that - to extract an information which may be useful for research of a particular linguistic phenomenon from the treebank, where this phenomenon is not explicitly tagged. The axtracted information should provide a linguistic basis for research in the fields of natural language parsing and information retrieval (exploiting linguistically motivated

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 195–206, 2009.

methods in information retrieval seems can recently be found in a number of papers, cf. e.g. [1]). The treebank under consideration is the Prague Dependency Treebank (PDT)[1], [3] a large and elaborated corpus with rich syntactic annotation of Czech sentences. The reason why we concentrate on this particular corpus is very simple - it is the only existing large scale syntactically annotated corpus for Czech. The phenomenon we are interested in are Czech complex sentences, the mutual relationship of their clauses and properties of those clauses. In the following sections we would like to present a brief description of the PDT, followed by a discussion of the annotation of clauses in complex sentences (and their parts - segments) in the PDT. Then we are going to describe an automatic method how to extract the required information from PDT (where it is not explicitly marked). In the last section we are going to present a discussion concerning the methods and results of an evaluation of the method presented in the paper.

2 The Prague Dependency Treebank

The Prague Dependency Treebank is a result of a large scale project started in 1996 at the Faculty of Mathematics and Physics at the Charles University in Prague. It is a corpus annotated on multiple levels - morphological, analytical and underlying-syntactic layer (for a description of the tagging scheme of PDT, see e.g. [5], [2], [7], [8], and the two manuals for tagging published as Technical Reports by UFAL and CKL of the Faculty of Mathematics and Physics, Charles University Prague (see [6]) and available also on the website http://ufal.mff.cuni.cz).

2.1 Clauses and Segments

Unfortunately, although the annotation scheme of PDT allows for a very deep description of many kinds of syntactic relationships, there is no explicit annotation of the mutual relationships of individual clauses in complex sentences in the corpus. Even worse, even the units which the clauses consist from (segments - cf. the following informal definition) are not explicitly and consistently annotated. In order to make clear what we understand by the notion of a *segment* in the following text, let us mention that very informally a *segment* is a term describing individual part of a simple clause which is visually separated from other segments by some separator (conjunction, relative pronoun, bracket, punctuation mark etc.). The segments are of course smaller units than clauses, a clause consists from more than a single segment for example in case that it contains coordination or when it is divided into more parts for example by an embedded subordinated clause. A more detailed description of segments can be found e.g. in [9].

2.2 Annotation of Segments at the Analytic Level of the PDT

The identification of segments and their mutual relationship might be an important part of the process of the identification of whole clauses, so let us look

[1] http://ufal.mff.cuni.cz/pdt2.0/

at the annotation of segments in the PDT. More precisely, let us look at the annotation at the surface syntactic (analytic) level of the PDT, because it constitutes much more natural information source for our purpose than the deeper, tectogrammatical level (mainly due to the fact that the analytic trees of PDT correspond more directly to individual tokens (words, punctuation marks etc.) of the input sentence. The tectogrammatical level represents a meaning of the sentence, the correspondence to its surface form is not so straightforward as it is in the case of the analytic level.

Formally, the structure of a sentence at the analytic layer of PDT is represented as a dependency-based tree, i.e. a connected acyclic directed graph in which no more than one edge leads into a node. The edges represent syntactic relations in the sentence (dependency, coordination and apposition being the basic ones). The nodes – labelled with complex symbols (sets of attributes) – represent word forms and punctuation marks.

In particular, there are no nonterminal nodes in PDT that would represent more complex sentence units – such units are expressed as (dependency) subtrees, see also Figures 1, 2 and 3:

- Complex units that correspond to particular "phrases", as verb phrase, nominal phrase or prepositional phrase – such units are expressed as subtrees rooted with nodes labelled with respective governing word forms, e.g. governing verb (its "lexical" part in case of analytical verb form, or copula in verbal-nominal predicate, or modal verb), governing noun, or preposition (as a "head" of a whole prepositional phrase);
- Dependent clauses – in principle, they are rendered as subtrees rooted with a node labelled with the governing verb of dependent clause (e.g. for attributive dependent clauses), or with a node for subordinating conjunction (adverbial and content clauses);
- Coordinated structures and sentence units in apposition (whether they are sentence members or whole clauses) – they are represented by subtrees rooted with nodes that correspond to a coordinating conjunction or some formal flag for an apposition (e.g. comma, brackets).

The rules described above always concern syntactic relationships between a pair of tokens (words) in the sentence. They do not deal by any means with bigger units – not even in the case of coordination where it would be natural. Segments and clauses are not explicitly marked in the tree, nor is their mutual relationship. We can, of course, intuitively suppose that subtrees might be rooted with the governing node of the segment or a clause as a natural solution. Unfortunately, it turned out that this is not the case at the analytic level of PDT. Let us demonstrate this fact on constructions contained in brackets, visually very easily distinguishable sentential segments with a direct relationship to the structure of individual clauses (a bracketed construction is one of the phenomena dividing a clause into more segments). Unlike punctuation marks, the brackets unambiguously show the beginning and the end of a text inserted into a sentence. It is therefore quite natural to expect this easily detectable segment to be annotated in a single consistent way.

The following examples of structures assigned to segments in brackets have been borrowed from [4], see Fig. 1, 2 and 3.

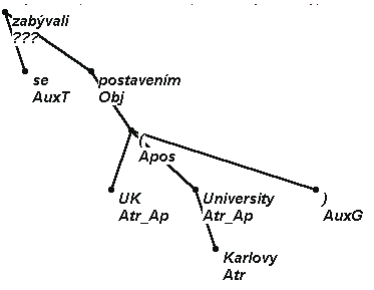


Fig. 1. PDT: Segment in parenthesis as an apposition: *zabývali se postavením UK (Univerzity Karlovy)* [(they) dealt-with *refl.* a_position of_UK (University of_Charles)]

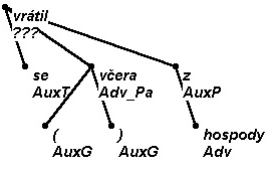


Fig. 2. PDT: Segments in parenthesis as a sentence member: *vrátil se (včera) z hospody* [(he)... returned *refl* (yesterday) from a_pub]

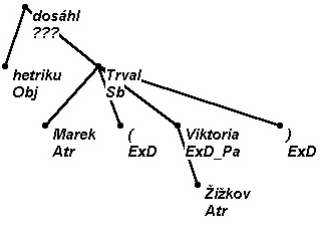


Fig. 3. PDT: Segments in parenthesis as an independent sentence part: *a hetriku dosáhl Marek Trval (Viktorie Žižkov)* [and hattrick_{Obj} achieved Marek Trval_{Subj} (Viktorie Žižkov)]

Let us point out that not only the annotation of a content of these parenthesis differs, but even the mutual position of both types of brackets in the tree is different. The situation is similar when we look at clauses, the main difference being the size - in the case of whole clauses the examples would be longer and even less transparent than the examples presented above for bracketed segments.

Although inconsistently annotated at the first sight, the trees contain an extremely valuable syntactic information which may serve as a basis for an automatic re-annotation procedure. Such a procedure should exploit the fact that in other respects, with regard to other syntactic phenomena, the corpus had been

annotated with extreme care and with a great deal of consistency. Actually, the reason for inconsistency wrt. the annotation of obvious segments is the consistency achieved wrt. a different syntactic phenomenon, in this case the endeavor to annotate the apposition and coordination in a similar way, i.e. with the whole construction depending on a conjunction / appositional comma.

3 Definition of a Clause Based on the Analytic Layer of the PDT

We base our definition of clauses on the analytic level of PDT. This definition will serve as a basis for the algorithm of clause identification. By taking advantage of the analytic annotation, we could come up with a simple definition that only minimally refers to language intuition and meaning.

A clause is defined as a subtree of a predicate, the predicate inclusive. There are two exceptions to this general rule:

1. A subordinating conjunction governing the predicate belongs to the clause.
2. A clause whose predicate is in the subtree of another clause is not considered to be a part of the governing clause.

Since not every predicate is explicitly annotated as such in the analytic level of PDT, this amounts to

1. tokens explicitly marked as predicates (have analytic function "Pred"),
2. finite autosemantic verbs,
3. tokens that govern a node of the analytic function "AuxV" (this denotes predicates formed by compound verbs) and
4. tokens that are coordinated with a predicate.

There are exceptions and special cases to these rules:

- ad 2: Finite verbs that hold coordination or apposition are not considered to be predicates for our purposes. See subsection 3.3.
- ad 3: If the token governing an AuxV-node is a coordinating conjunction, then it is not considered a predicate. In such a case though, the coordinated tokens are considered as if they governed an AuxV-node and are thus recognized as predicates.

3.1 Sections of a Sentence

The criteria introduced above state what is a clause and what tokens belong to one. This is one of the two goals of our algorithm. The second one deals with relations between and among clauses. These are basically dependency and coordination relations. Since clauses are not atomic objects and there are cases where a token belongs to more than one clause, we need to introduce a new, more general term: *sections*. The reason why we cannot use the notion of segment instead of a section is the different nature of both components of a clause. Segments are units distinguishable on the surface, they are defined for sentences in the form of sequences

of word forms and punctuation marks. Sections, on the other hand, are defined as units distinguishable from the surface syntactic (analytic) representation of a sentence. Both terms refer to units which are similar but not identical.

A section of a sentence is defined by its *representative* and its *component*. The representative of a section is a token of the sentence or its technical root (every sentence in PDT has a technical root). Each token as well as the technical root represents no more than one section. The component of a section is a subset of the tokens of the sentence. The components of the sentence's sections constitute a perfect coverage of the sentence's tokens.

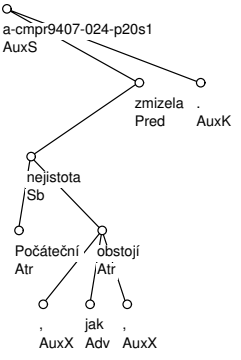


Fig. 4. Analytic tree of the sentence "Počáteční nejistota, jak obstojí, zmizela." [Initial uncertainty, how it-will-do, vanished.]

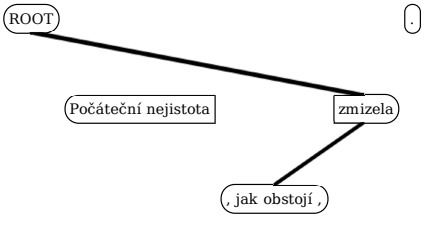


Fig. 5. Sections of the sentence from Figure 4. Each bordered shape marks a section. Each horizontal level contains one clause. The lines mark bloodline relations of clauses.

Typically, the representative of a section belongs also to its component. The exception is the technical root, which can represent a section but can never be in its component.

The component of a section forms a tree on the analytic level. The only exception is the section represented by the technical root, the component of which can be a forest.

Sections are of three types:

1. clause,
2. coordination and
3. adjunct.

Each clause constitutes a section of the type *clause*. Its representative is the finite verb that governs the clause and its component consists of the tokens that belong to the clause.

Coordination of clauses. Whenever two or more clauses are coordinated, the coordination itself constitutes an extra section of the type *coordination*. Its representative is the coordinating conjunction (or punctuation token) that holds the coordination (i.e. it has the analytic function of "Coord" or "Apos" in case of appositions, which we treat equally to coordinations). The component of a coordination section is its representative and leaf tokens dependent on it that are not related to the coordinated clauses. That is:

1. other conjunctions, commas and other separators of the coordinated clauses,
2. other words of the conjunction in case of multi-word conjunctions,
3. auxiliary leaf tokens (those with analytic function beginning with "Aux").

The third case emerges when a coordination of clauses governs a phrase that effectively depends on all the coordinated clauses. Take the English example: *"John loves Mary but won't marry."* There are two finite verbs present: "loves" and "won't". So our above stated definition would recognize two clauses plus one coordination.

Clause 1 would certainly contain tokens "loves Mary", Clause 2 would contain "won't marry" and the coordination would only contain "but". So, where would "John" go? It's him who loves Mary and it's also him who won't marry the poor girl. We could see this sentence as a coordination of clauses with the subject distributed: "John loves Mary" + "John won't marry".

To denote this type of relation, we give "John" (with his whole (empty) subtree) his own section of the type *adjunct*. Sections of this type are always formed by subtrees (dependent clauses excluding) of tokens that are not clauses and depend on a coordination of clauses but are not coordinated in it.

Tokens that do not fall into any section by the above criteria belong to the section represented by the technical root. Its type is set to *clause*, but that is a purely technical decision.

3.2 Inter-clausal Relations

The sections were defined with the intention to obtain a help when capturing relations among clauses. Notice that every section's component has a tree-like structure. The only exception again being the clause whose representative is the technical root. This means that every section has one root token. We can therefore define bloodline relations between sections like this:

Definition 1 (Parent section.) Let D be a section whose representative is not the technical root. Let r be the root token of D. Let p be the analytic parent token of r. We call the section to which p belongs or which p represents the *parent* section of D. The root section is its own parent.

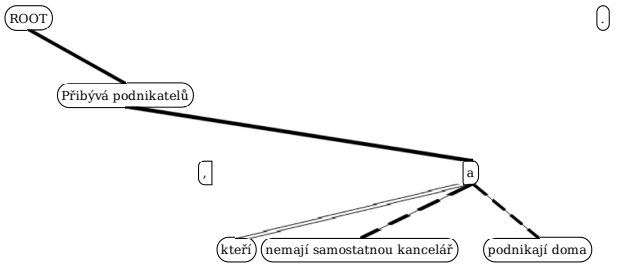


Fig. 6. Sections and their relations of the sentence "Přibývá podnikatelů, kteří nemají samostatnou kancelář a podnikají doma." [The-number-grows of-businessmen, who don't-have separate office and work at-home.] (The number of businessmen who have no separate office and work at home grows.) The main clause "Přibývá podnikatelů" (the number of businessmen grows) governs the coordination formed by the conjunction and the comma. There are two dependent clauses: "kteří nemají samostatnou kancelář" (who have no separate office) and "kteří podnikají doma" (who work at home). Their disjoint parts are marked as coordinated sections (*section type: clause, child type: member*) and their common word "kteří" (who) is marked as another section (*section type: adjunct, child type: part*).

As in the real life, one child is sometimes quite unlike another, that is an experience of many human parents. It's the same here, so we differentiate several *types of children*. These are:

1. dependants,
2. members and
3. parts.

Every clause and every adjunct can only have child sections of the *dependant* type. Coordinations, on the other hand, can have children of any type.
Whenever a coordination has a child of the *member* type, it means that the child section is coordinated in the coordination.
Whenever a coordination has a child of the *dependant* type, it means that the child section is effectively dependent on all the sections coordinated in the parent coordination.
Whenever a coordination has a child of the *part* type, it means that the child section belongs to all the sections coordinated in the parent coordination. Children of the *part* type are exactly the sections of the type *adjunct*.

This approach allows to capture virtually any clause structure from the PDT, keeping information about tokens belonging to clauses, their dependencies and coordinations. The grammatical roles of clauses are easily extracted from analytic functions of their representatives.

Since the definitions mentioned above are all based on information available on the analytic and lower levels of PDT, the algorithm for extracting clause structure from the analytic annotation is a straight-forward rewrite of those definitions into a programming language.

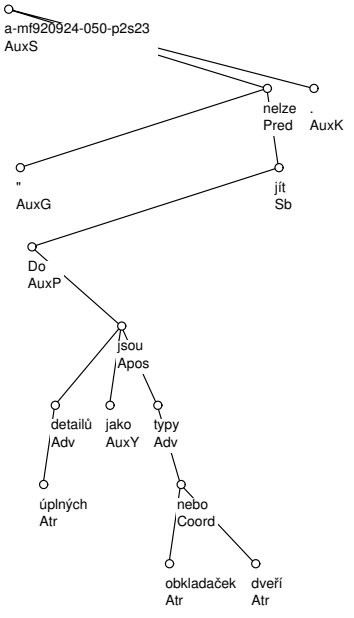


Fig. 7. Analytic tree of a sentence containing an apposition held by a finite verb

3.3 Appositions Held by a Finite Verb

The only phenomenon we know that our algorithm is not handling correctly concerns finite verbs that bear an apposition (or potentially coordination), that is, they have the analytic function of "Coord" or "Apos". Take the sentence "Do úplných detailů jako jsou typy obkladaček nelze jít." [Into sheer details like are types of-tiles is-not-possible to-go.] (We can't go into sheer details like the types of tiles.) Figure 7 shows its analytic tree. The clause that should apparently be recognized is formed by tokens "jako jsou typy obkladaček" [like are types of-tiles]. Notice that the tokens "úplných detailů" [sheer details], which do not belong to the inner clause, are in the subtree of the inner clause's founding verb "jsou" [are].

Here, the most profound rule our definition is based upon – that a clause is a subtree of its predicate – breaks the factual distribution of clauses. Even if we only tore off the clause itself (which is a well-formed tree), the parent clause would stop being connected. Our way of dealing with this is to simply ignore the presence of the inner clause and keep it as a part of the parent clause. This seems to be the best way to go, as it has virtually no negative consequences, it is very easy to detect and implement, and the phenomenon is not frequent.

4 Evaluation

Applying the algorithm described above on the PDT, we get clauses marked up in the sentences. The process is deterministic, it reflects the annotation of individual nodes of an analytic tree of the PDT. It is also an application of a *definition*,

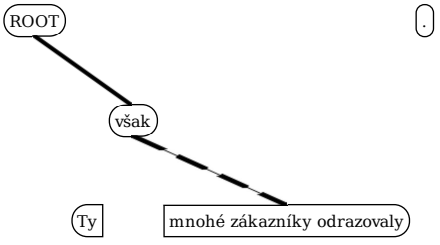


Fig. 8. Sections of the sentence "Ty však mnohé zákazníky odrazovaly." [Those however many clients$_{obj}$ discouraged.] (Those have, however, discouraged many clients.) Here, the sentence is marked up as one coordinated clause, governed by the coordination formed by the "však" (however) conjunction. The reason for this maybe surprising annotation is that the sentence is de facto coordinated with the previous one.

not an attempt to model a given linguistic phenomenon. The data should then be used as gold standard for clause detection from the lower levels of annotation (e.g. morphological). It is clear that standard precision/recall evaluation is not useful in this case.

Instead, we have decided to try to count the sentences where the algorithm provides clauses in a different manner than we think a human would. The difference between automatic and man-made annotation is based upon the fact that our algorithm keeps clauses syntactically compact, while humans prefer to keep them linearly compact. These requirements go against each other mostly in the case where a coordination section has tokens inside some of the coordinated clauses. See Figure 8. Other cases include erroneously annotated trees in the corpus (garbage in – garbage out) and the presence of adjunct segments, which humans tend to connect to the adjacent clause only.

Table 1. Evaluation of the extraction of clauses from analytic trees.

	Count	Ratio
Linearly incompact	7124	8.59%
Appositions	114	0.14%
incomp&appos	7225	8.71%
All	82944	100.00%

Table 1 presents the evaluation done on a large subset of the PDT. First row shows the number of sentences where a clause has alien tokens inside (precisely, where a clause is not bordered by conjunctions or punctuation). Second row shows the number of the problematic appositions whose governing token is a verb. Row three shows the number of sentences manifesting both phenomena. Evidently, the number of sentences where the intuitive and the definition-conforming splits of tokens to clauses differ is significant. However, the number of sentences where the algorithm fails to *do the right thing* (row 2) is almost negligible.

5 Conclusions

The most valuable result of the experiment described in the paper is actually the treebank of Czech with automatically added annotation of mutual relationships among clauses in complex sentences. Our experiment shows that it is possible to reconstruct this information from the syntactic relationships among individual words. The data obtained as a result of application of our algorithm may serve for future experiments with complex sentences and clauses. Our algorithm provides enough data not only for testing the theories, but also for the application of stochastic or machine–learning methods.

By ensuring that every section (and thus clause) is a tree on its own, our algorithm allows to develop a sort of a two-step parsing algorithm, where the first step will be the analysis of the structure of complex sentences and the second step the parsing of single clauses. The results of many existing parsers (cf. [10] etc.) show a dependency of parsing accuracy wrt. the length of input sentences, therefore dividing the complex sentence into shorter units and parsing them separately might help to overcome this natural behavior of existing parsers. This experiment is yet to be done, though.

Acknowledgments

This work was supported by the Grant Agency of the Czech Republic, grant No. 405/08/0681 and by the programme Information Society of the GAAV CR, grant No. 1ET100300517.

References

1. Fernández, M., de la Clergerie, E., Vilares, M.: Mining Conceptual Graphs for Knowledge Acquisition. In: Proc. of ACM CIKM Workshop on Improving Non-English Web Searching (inews 2008), Napa Valley, USA, pp. 25–32 (2008) ISBN 978-1-60558-253-5
2. Hajič, J., Hladká, B.: Probabilistic and Rule-Based Tagger of an Inflective Language – A Comparison. In: Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington, DC, pp. 111–118 (1997)
3. Hajič, J., Hajičová, E., Pajas, P., Panevová, J., Sgall, P., Vidová-Hladká, B.: Prague Dependency Treebank 1.0 (Final Production Label). In: CD-ROM, Linguistic Data Consortium (2001)
4. Hajič, J., Panevová, J., Buráňová, E., Urešová, Z., Bémová, A.: Anotace Pražského závislostního korpusu na analytické rovině: pokyny pro anotátory (in Czech) Technical Report No. 28, ÚFAL MFF UK, Prague, Czech Republic (1999)
5. Hajič, J.: Building a Syntactically Annotated Corpus: The Prague Dependency Treebank. In: Issues of Valency and Meaning, Karolinum, Praha, pp. 106–132 (1998)
6. Hajič, J., Panevová, J., Buráňová, E., Urešová, Z., Bémová, A.: A Manual for Analytic Layer Tagging of the Prague Dependency Treebank (2001) ISBN 1-58563-212-0

7. Hajièová, E.: Prague Dependency Treebank: From Analytic to Tectogrammatical Annotations. In: Sojka, P., Matoušek, V., Kopeèek, I. (eds.) Text, Speech, Dialogue, Brno, Masaryk University, pp. 45–50 (1998)
8. Hajièová, E.: The Prague Dependency Treebank: Crossing the Sentence Boundary. In: Matoušek, V., Mautner, P., Ocelíková, J., Sojka, P. (eds.) Text, Speech and Dialogue, pp. 20–27. Springer, Berlin (1999)
9. Kuboň, V., Lopatková, M., Plátek, M., Pognan, P.: Segmentation of complex sentences. In: Sojka, P., Kopeček, I., Pala, K. (eds.) TSD 2006. LNCS, vol. 4188, pp. 151–158. Springer, Heidelberg (2006)
10. Zeman, D.: Parsing with a Statistical Dependency, PhD. Thesis, Charles university, Prague (2004)

A General Method for Transforming Standard Parsers into Error-Repair Parsers*

Carlos Gómez-Rodríguez[1], Miguel A. Alonso[1], and Manuel Vilares[2]

[1] Departamento de Computación, Universidade da Coruña (Spain)
{cgomezr,alonso}@udc.es
[2] Escuela Superior de Ingeniería Informática, Universidade de Vigo (Spain)
vilares@uvigo.es

Abstract. A desirable property for any system dealing with unrestricted natural language text is robustness, the ability to analyze any input regardless of its grammaticality. In this paper we present a novel, general transformation technique to automatically obtain robust, error-repair parsers from standard non-robust parsers. The resulting error-repair parsing schema is guaranteed to be correct when our method is applied to a correct parsing schema verifying certain conditions that are weak enough to be fulfilled by a wide variety of parsers used in natural language processing.

1 Introduction

In real-life domains, it is common to find natural language sentences that cannot be parsed by grammar-driven parsers, due to insufficient coverage (the input is well-formed, but the grammar cannot recognize it) or ill-formedness of the input (errors in the sentence or errors caused by input methods). A standard parser will fail to return an analysis in these cases. A *robust parser* is one that can provide useful results for such extragrammatical sentences.

The methods that have been proposed to achieve robustness in parsing fall mainly into two broad categories: those that try to parse well-formed fragments of the input when a parse for the complete sentence cannot be found (partial parsers, such as that described in [6]) and those which try to assign a complete parse to the input sentence by relaxing grammatical constraints, such as *error-repair parsers*, which can find a complete parse tree for sentences not covered by the grammar by supposing that ungrammatical strings are corrupted versions of valid strings.

The problem of repairing and recovering from syntax errors during parsing has received much attention in the past (see for example the list of references provided in the annotated bibliography of [5, section 18.2.7]) and recent years (see for example [15,17,2,7,1,11]). In this paper, we try to fill the gap between standard and error-repair parsing by proposing a transformation for automatically obtaining error-repair

* Partially supported by Ministerio de Educación y Ciencia and FEDER (HUM2007-66607-C04) and Xunta de Galicia (PGIDIT07SIN005206PR, INCITE08E1R104022ES, INCITE08ENA305025ES, INCITE08PXIB302179PR and Rede Galega de Procesamento da Linguaxe e Recuperación de Información).

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 207–219, 2009.

parsers, in the form of *error-repair parsing schemata*, from standard parsers defined as *parsing schemata*.[1]

2 Standard Parsing Schemata

Parsing schemata [13] provide a formal, simple and uniform way to describe, analyze and compare different parsing algorithms. The notion of a parsing schema comes from considering parsing as a deduction process which generates intermediate results called *items*. An initial set of items is obtaineddirectly from the input sentence, and the parsing process consists of the application of inference rules (*deduction steps*) which produce new items from existing ones. Each item contains a piece of information about the sentence's structure, and a successful parsing process will produce at least one *final item* containing a full parse tree for the sentence or guaranteeing its existence.

When working with a context-free grammar[2] $G = (N, \Sigma, P, S)$, items are sets of trees from a set denoted $Trees(G)$, defined as the set of finitely branching finite trees in which children of a node have a left-to-right ordering, every node is labelled with a symbol from $N \cup \Sigma \cup (\Sigma \times \mathbb{N}) \cup \{\epsilon\}$, and every node u satisfies one of the following conditions:

- u is a leaf,
- u is labelled A, the children of u are labelled $X_1, \ldots, X_n$ and there is a production $A \rightarrow X_1 \ldots X_n \in P$,
- u is labelled A, u has one child labelled ϵ and there is a production $A \rightarrow \epsilon \in P$,
- u is labelled a and u has a single child labelled (a, j) for some j.

The pairs (a, j) will be referred to as *marked terminals*, and when we deal with a string $a_1 \ldots a_n$, we will usually write $\underline{a}_j$ as an abbreviated notation for (a_j, j) in the remainder of this paper. The natural number j is used to indicate the position of the word a in the input.

Valid parses for a string are represented by items containing complete *marked parse trees* for that string. Given a grammar G, a marked parse tree for a string $a_1 \ldots a_n$ is any tree $\tau \in Trees(G)$ such that $root(\tau) = S$ and $yield(\tau) = \underline{a}_1 \ldots \underline{a}_n$, where $root(\tau)$ refers to the root node of τ and $yield(\tau)$ refers to the frontier nodes of τ. An item containing such a tree for some arbitrary string is called a *final item*. An item containing such a tree for a particular string $a_1 \ldots a_n$ is called a *correct final item* for that string.

For each input string, a parsing schema's deduction steps allow us to infer a set of items, called *valid items* for that string. A parsing schema is said to be *sound* if all valid final items it produces for any arbitrary string are correct for that string. A parsing schema is said to be *complete* if all correct final items are valid. A parsing schema which is both sound and complete is said to be *correct*. A correct parsing schema can be used

[1] *Schemata* is the plural form of the singular noun *schema*.

[2] Although in this paper we will focus on context-free grammars, both standard and error-repair parsing schemata can be defined analogously for other grammatical formalisms.

to obtain a working implementation of a parser by using deductive parsing engines as the ones described in [12,4] to obtain all valid final items.[3]

3 Error-Repair Parsing Schemata

The parsing schemata formalism introduced in the previous section does not suffice to define error-repair parsers that can show a robust behaviour in the presence of errors. In these parsers, we should obtain items containing "approximate parses" if an exact parse for the sentence does not exist. Approximate parses need not be members of $Trees(G)$, since they may correspond to ungrammatical sentences, but they should be *similar* to a member of $Trees(G)$. Formalizing the notion of "similarity" as a distance function, we can obtain a definition of items allowing approximate parses to be generated.

3.1 Defining Error-Repair Parsing Schemata

Given a context-free grammar $G = (N, \Sigma, P, S)$, we shall denote by $Trees'(G)$ the set of finitely branching finite trees in which children of a node have a left-to-right ordering and every node is labelled with a symbol from $N \cup \Sigma \cup (\Sigma \times \mathbb{N}) \cup \{\epsilon\}$. Note that $Trees(G) \subset Trees'(G)$.

Let $d : Trees'(G) \times Trees'(G) \rightarrow \mathbb{N} \cup \{\infty\}$ be a function verifying the usual distance axioms (strict positiveness, symmetry and triangle inequality).

We shall denote by $Trees_e(G)$ the set $\{t \in Trees'(G) \mid \exists t' \in Trees(G) : d(t, t') \leq e\}$, i.e., $Trees_e(G)$ is the set of trees that have distance e or less to some valid tree in the grammar. Note that, by the strict positiveness axiom, $Trees_0(G) = Trees(G)$.

Definition 1. *(approximate trees)*
We define the set of *approximate trees* for a grammar G and a tree distance function d as $ApTrees(G) = \{(t, e) \in (Trees'(G) \times \mathbb{N}) \mid t \in Trees_e(G)\}$. Therefore, an approximate tree is the pair formed by a tree and its distance to some tree in $Trees(G)$. □

This concept of approximate trees allows us to precisely define the problems that we want to solve with error-repair parsers. Given a grammar G, a distance function d and a sentence $a_1 \ldots a_n$, the *approximate recognition problem* is to determine the minimal $e \in \mathbb{N}$ such that there exists an approximate tree $(t, e) \in ApTrees(G)$ where t is a marked parse tree for the sentence. We will call such an approximate tree an *approximate marked parse tree* for $a_1 \ldots a_n$.

Similarly, the *approximate parsing* problem consists of finding the minimal $e \in \mathbb{N}$ such that there exists an approximate marked parse tree $(t, e) \in ApTrees(G)$ for the sentence, and finding all approximate marked parse trees for the sentence.

Definition 2. *(approximate item set)*
Given a grammar G and a distance function d, we define an *approximate item set* as a set $\mathcal{I}'$ such that

[3] An example of a correct parsing schema is the Earley parsing schema, which defines the parser described by [3]. A full definition and proof of correctness for this schema can be found at [14].

$\mathcal{I}' \subseteq ((\bigcup_{i=0}^{\infty} \Pi_i) \cup \{\emptyset\})$
where each Π_i is a partition of the set $\{(t, e) \in ApTrees(G) \mid e = i\}$. □

Each element of an approximate item set is a set of approximate trees, and will be called an *approximate item*. Note that the concept is defined in such a way that each approximate item contains approximate trees with a single value of the distance e. This concrete value of e is what we will call *parsing distance* of an item ι, or $dist(\iota)$:

Definition 3. *(parsing distance of an item)*
Let $\mathcal{I}' \subseteq ((\bigcup_{i=0}^{\infty} \Pi_i) \cup \{\emptyset\})$ be an approximate item set as defined above, and $\iota \in \mathcal{I}'$. The *parsing distance* associated to the nonempty approximate item ι, $dist(\iota)$, is defined by the (trivially unique) value $i \in \mathbb{N} \mid \iota \in \Pi_i$. In the case of the empty approximate item $\emptyset$, we will say that $dist(\emptyset) = \infty$. □

Definition 4. *(error-repair parsing schema)*
Let G be a context-free grammar, d a distance function, and $a_1 \ldots a_n \in \Sigma^*$ a string. An *error-repair instantiated parsing system* is a triple $(\mathcal{I}', H, D)$ such that $\mathcal{I}'$ is an approximate item set with distance function d, H is a set of hypotheses such that $\{a_i(\underline{a_i})\} \in H$ for each $a_i, 1 \leq i \leq n$, and D is a set of deduction steps such that $D \subseteq \mathcal{P}_{fin}(H \cup \mathcal{I}') \times \mathcal{I}'$. An *error-repair uninstantiated parsing system* is a triple $(\mathcal{I}', \mathcal{K}, D)$ where $\mathcal{K}$ is a function such that $(\mathcal{I}', \mathcal{K}(a_1 \ldots a_n), D)$ is an error-repair instantiated parsing system for each $a_1 \ldots a_n \in \Sigma^*$ (in practice, we will always define this function as $\mathcal{K}(a_1 \ldots a_n) = \{\{a_i(\underline{a_i})\} \mid 1 \leq i \leq n\} \cup \{\epsilon(\epsilon)\}$). Finally, an *error-repair parsing schema* for a class of grammars $\mathcal{CG}$ and a distance function d is a function that assigns an error-repair uninstantiated parsing system to each grammar $G \in \mathcal{CG}$. □

Definition 5. *(final items)*
The set of *final items* for a string of length n in an approximate item set is defined by

$\mathcal{F}(\mathcal{I}', n) = \{\iota \in \mathcal{I} \mid \exists (t, e) \in \iota : t$ is a marked parse tree for some string $a_1 \ldots a_n \in \Sigma^\star\}$.

The set of *correct final items* for a string $a_1 \ldots a_n$ in an approximate item set is defined by

$\mathcal{CF}(\mathcal{I}', a_1 \ldots a_n) = \{\iota \in \mathcal{I} \mid \exists (t, e) \in \iota : t$ is a marked parse tree for $a_1 \ldots a_n\}$. □

The concepts of *valid items*, *soundness*, *completeness* and *correctness* are analogous to the standard parsing schemata case. Note that the *approximate recognition* and *approximate parsing* problems defined earlier for any string and grammar can be solved by obtaining the set of correct final items for that string whose associated distance is minimal. These items can be deduced by any correct error-repair parsing schema, since they are a subset of correct final items.

3.2 A Distance Function for Repairs Based on the Edit Distance

Let us suppose a generic scenario where we would like to repair errors according to edit distance. The edit distance or Levenshtein distance [8] between two strings is the minimum number of insertions, deletions or substitutions of a single terminal needed to transform eother of the strings into the other one. Given a string $a_1 \ldots a_n$ containing

errors, we would like our parsers to return an approximate parse based on the exact parse tree of one of the strings in $L(G)$ whose Levenshtein distance to $a_1 \ldots a_n$ is minimal.

A suitable distance function d for this case is given by the number of tree transformations that we need to transform one tree into another, if the transformations that we allow are inserting, deleting or changing the label of marked terminal nodes in the frontier. Therefore, $d(t_1, t_2) = e$ if t_2 can be obtained from t_1 by performing e transformations on marked terminal nodes in t_1, and $d(t_1, t_2) = \infty$ otherwise.

4 An Error-Repair Transformation

The error-repair parsing schemata formalism allows us to define a transformation to map correct parsing schemata to correct error-repair parsing schemata that can successfully obtain approximate parses minimizing the Levenshtein distance. This transformation has been formally defined, but for space reasons we cannot include here all the definitions required for a rigorous, formal description. Therefore, we will instead provide an informal explanation of how the technique works, and a sketch of the proof that correctness is preserved.

4.1 From Standard Parsers to Error-Repair Parsers

Most standard, non-robust parsers work by using grammar rules to build trees and link them together to form larger trees, until a complete parse can be found. Our transformation will be based on generalising parser deduction steps to enable them to link approximate trees and still obtain correct results, and adding some standard steps that introduce error hypotheses into the item set, which will be élegantly integrated into parse trees by the generalized steps.

The particular strategy used by parsers to build and link trees obviously varies between algorithms but, in spite of this, we can usually find two kinds of deduction steps in parsing schemata: those which introduce a new tree into the parse from scratch, and those which link a set of trees to form a larger tree. We will call the former *predictive steps* and the latter *yield union steps*.

Predictive steps can be identified because the yield of the trees in their consequent item does not contain any marked terminal symbol, that is, they generate trees which are not linked to the input string. Examples of predictive steps are the Earley *Initter* and *Predictor* steps. Yield union steps can be identified because the sequence of marked terminals in the yield of the trees of their consequent item (which we call the *marked yield* of these items[4]) is the concatenation of the marked yields of one or more of their antecedents[5], and the trees in the consequent item are formed by linking trees in

[4] In the sequel, we will use the notation $yield_m(t)$ to refer to the marked yield of a tree t, and $yield_m(\iota)$ to refer to the common marked yield of the trees in an item ι, which we will call marked yield of the item.

[5] Actually, predictive steps can also be seen as yield union steps where the marked yield of the consequent item is the concatenation of the marked yield of *zero* of their antecedents. From this perspective it is not necessary to define predictive steps, but the concept has been introduced for clarity.

antecedent items. Examples of yield union steps are Earley *Completer* and *Scanner*, and all the steps in the CYK parsing schema.

If all the steps in a parsing schema are predictive steps or yield union steps, we will call it a *prediction-completion parsing schema*. Most of the parsing schemata which can be found in the literature for widely-used parsers are prediction-completion parsing schemata, which allows us to generalize their steps to deal with an approximate item set in an uniform way. First of all, we must define this approximate item set. In order to do this, we take into account that each item in the item set of a prediction-completion parsing schemata can be represented by a triplet (p, i, j), where p is some entity from a set E (the form of the elements in E is not relevant for the transformation), and i, j are the leftmost and rightmost limits of the marked yields of the trees in the item. More formally, there exists a surjective *representation function* $r : E \times \mathbb{N} \times \mathbb{N} \rightarrow \mathcal{I}$, such that $yield_m(t) = \underline{a}_{i+1} \dots \underline{a}_j$ for every $t \in r(p, i, j)$[6]. We will denote the item $r(p, i, j)$ by $[p, i, j]$, which is the notation commonly used to represent items in the literature. Taking this into account, we can define an approximate item set $\mathcal{I}'$ as the set of approximate items of the form $[p, i, j, x]$, where an approximate tree $(t, x) \in ApTrees(G)$ belongs to $[p, i, j, x]$ iff $yield_m(t) = \underline{a}_{i+1} \dots \underline{a}_j$ and there exists a tree t' in an item of the form $[p, i', j']$ such that $d(t, t') = x$[7]. With this approximate item set defined, we can generalize the steps in the following way:

- Sets of predictive steps taking as antecedents items of the form $[p_1, i_1, j_1], [p_2, i_2, j_2], \dots [p_a, i_a, j_a]$ and producing as consequent an item of the form $[p_c, i_c, i_c]$ are generalized to sets of steps taking antecedents $[p_1, i_1, j_1, x_1], [p_2, i_2, j_2, x_2], \dots [p_a, i_a, j_a, x_a]$ and producing a consequent $[p_c, i_c, i_c, 0]$. This means that we let the distances associated to antecedents take any value, and we always generate consequents with 0 as associated distance (since items generated by predictive steps are not linked in any way to the input, they need not consider error hypotheses).
- Sets of yield union steps taking as antecedents items of the form $[p_1, i_0, i_1], [p_2, i_1, i_2], \dots [p_y, i_{y-1}, i_y], [p'_1, k_1, l_1], [p'_2, k_2, l_2] \dots [p'_a, k_a, l_a]$ and producing as consequent an item of the form $[p_c, i_0, i_y]$ are generalized to sets of steps taking antecedents $[p_1, i_0, i_1, x_1], [p_2, i_1, i_2, x_2], \dots [p_y, i_{y-1}, i_y, x_y], [p'_1, k_1, l_1, x'_1], [p'_2, k_2, l_2, x'_2] \dots [p'_a, k_a, l_a, x'_a]$ and producing as consequent an item of the form $[p_c, i_0, i_y, x_1 + x_2 + \dots + x_y]$. This means that we let the distances associated to antecedents take any value, and the distance associated to the consequent is the sum of the distances associated to the items used for building it. Therefore, we

[6] Formally speaking, the necessary condition for the existence of such a representation function is that the trees in each item all have the same marked yield, of the form $yield_m(t) = \underline{a}_{i+1} \dots \underline{a}_j$. In practice, this condition should always hold in well-formed prediction-completion parsing schemata.

[7] Note that $\mathcal{I}'$ verifies the definition of an approximate item set if, and only if, $d(t1, t2) = \infty$ for every $t_1 \in [p_1, i_1, j_1], t_2 \in [p_2, i_2, j_2]$ such that $p1 \neq p2$. That is, items associated to different entities should be at infinite distance. This is the case for known item sets (such as the Earley, CYK or Left-Corner item sets) and our distance function.

are propagating and adding the errors coming from all the trees used to build the consequent tree.

Now we know how to adapt deduction steps in standard parsing schemata to adequately handle distances between trees, but this is not enough to obtain a correct error-repair parsing schema: the generalized steps will be able to link approximate trees and propagate the errors associated to each of them, but they will not detect any errors in the string by themselves. In order to do this, we must add some steps to the schema to introduce error hypotheses, i.e., elementary approximate trees representing the presence of an error in the input. This can be done in a way totally independent from the starting parser, by adding always the same *error hypothesis steps*, which are the following:

1. $SubstitutionHyp : \frac{[a, i, i+1]}{[b, i, i+1, 1]}\ b \in \Sigma$
2. $DeletionHyp : \frac{[\epsilon, i, i]}{[b, i, i, 1]}\ b \in \Sigma$
3. $InsertionHyp : \frac{[a, i, i+1]}{[\epsilon, i, i+1, 1]}$
4. $CorrectHyp : \frac{[a, i, i+1]}{[a, i, i+1, 0]}$
5. $InsCombiner1 : \frac{\begin{matrix}[\epsilon, 0, j, e_1] \\ [(a|\epsilon), j, k, e_2]\end{matrix}}{[(a|\epsilon), 0, k, e_1 + e_2]}$
6. $InsCombiner2 : \frac{\begin{matrix}[(a|\epsilon), i, j, e_1] \\ [\epsilon, j, k, e_2]\end{matrix}}{[(a|\epsilon), i, k, e_1 + e_2]}$
7. $DistanceIncreaser : \frac{[p, i, j, e]}{[p, i, i, e+1]}\ p \in E$

The first three steps generate the basic error hypotheses: $SubstitutionHyp$ produces trees of the form $b \rightarrow \underline{a}_{i+1}$ for each symbol a_{i+1} in the input string (input symbol) and each $b \in \Sigma$ (expected symbol), which correspond to the hypothesis that the symbol a_{i+1} that we find in the input string is the product of a substitution error, and should appear as b instead in order for the string to be grammatical. The $DeletionHyp$ step generates deletion hypothesis trees of the form $b \rightarrow \epsilon$, corresponding to assuming that the symbol b, which should be the $i+1$th symbol in the input, has been deleted. Finally, the $InsertionHyp$ infers trees of the form $\epsilon \rightarrow \underline{a}_{i+1}$ for each symbol a_{i+1} in the input string, corresponding to the hypothesis that the input symbol a_{i+1} is the product of an insertion error, and therefore should not be taken into account in the parse.

The $CorrectHyp$ step corresponds to the "no error" hypothesis, producing a tree representing the fact that there is no error in a given input symbol a_{i+1}.

The two $Combiner$ steps produce trees of the form $a_2(\epsilon(\underline{a}_1)a_2(\underline{a}_2))$ and $a_1(\underline{a}_1\epsilon(\underline{a}_2))$. If the first symbol in the input is the product of an insertion error, the corresponding hypothesis is combined with the hypothesis immediately to its right. Insertion hypotheses corresponding to symbols other than the first one are combined to the hypothesis immediately to their left.

The necessity of these two combiner steps comes from the fact that standard parsing schemata are not prepared to handle trees rooted at ϵ, as generated by the *Insertion-Hyp* step. The combiner steps allow trees rooted at ϵ to be attached to neighbouring trees rooted at a terminal. In this way, the steps obtained from generalising those in the standard schema can handle insertion hypotheses. An alternative is not using combiner steps, and instead imposing extra constraints on the schemata to be transformed so that they can handle trees rooted at ϵ. Any prediction-completion parsing schema

can be adapted to met these extra constraints, but this adaptation makes the conversion somewhat more complex.

Finally, the $DistanceIncreaser$ step is useless from a practical standpoint, but we have to include it in our error-repair parser if we wish to guarantee its completeness. The reason is that completeness requires the parser to be able to generate every possible correct final item, not just those with minimal associated distance. In practice, only minimal final items are needed to solve the approximate parsing and recognition problems. As this step is never necessary to generate a minimal item, it can be omitted in practical implementations of parsers.

4.2 The Transformation

Putting it all together, we can define the *error-repair transformation* of a prediction-completion parsing system $\mathbb{S}$ as the error-repair parsing system $\mathcal{R}(\mathbb{S})$ obtained by applying the following changes to it:

1. Add the $SubstitutionHyp$, $DeletionHyp$, $InsertionHyp$, $InsCombiner1$, $InsCombiner2$, $CorrectHyp$ and $DistanceIncreaser$ steps, as defined above, to the schema.
2. For every predictive step in the schema (steps producing an item with an empty yield), change the step to its generalization obtained (in practice) by setting the distance associated to each antecedent item A_i to an unbound variable e_i, and set the distance for the consequent item to zero. For example, the Earley step
   $$EarleyPredictor : \frac{[A \to \alpha.B\beta, i, j]}{[B \to .\gamma, j, j]}\ B \to \gamma \in P$$
   produces the step
   $$TransformedEarleyPredictor : \frac{[A \to \alpha.B\beta, i, j, e]}{[B \to .\gamma, j, j, 0]}\ B \to \gamma \in P\ .$$
3. For every yield union step in the schema (steps using items with yield limits (i_0, i_1), (i_1, i_2), ..., (i_{a-1}, i_a) to produce an item with yield $(i_1 \ldots i_a)$:
   - If the step requires an hypothesis $[a, i, i+1]$, then change all appearances of the index $i+1$ to a new unbound index j[8].
   - Set the distance for each antecedent item A_k with yield (i_{k-1}, i_k) to an unbound variable e_k, and set the distance for the consequent to $e_1 + e_2 + \ldots + e_a$.
   - Set the distance for the rest of antecedent items, if there is any, to unbound variables d_j.

   For example, the Earley step
   $$EarleyCompleter : \frac{[A \to \alpha.B\beta, i, j] \qquad [B \to \gamma., j, k]}{[A \to \alpha B.\beta, i, k]}$$
   produces the step

[8] Steps including hypotheses as antecedents are not strictly yield union steps according to the formal definition of yield union step that we have omitted due to lack of space. However, these steps can always be easily transformed to yield union steps by applying this transformation. Note that this change does not alter any of the significant properties of the original (standard) parsing schema, since items $[a, i, j]$ with $j \neq i+1$ can never appear in the deduction process.

$$TransformedEarleyCompleter : \frac{[A \rightarrow \alpha.B\beta, i, j, e_1] \qquad [B \rightarrow \gamma., j, k, e_2]}{[A \rightarrow \alpha B.\beta, i, k, e_1 + e_2]}$$

The Earley step

$$EarleyScanner : \frac{[A \rightarrow \alpha.a\beta, i, j] \qquad [a, j, j+1]}{[A \rightarrow \alpha a.\beta, i, j+1]}$$

produces the step:

$$TransformedEarleyScanner : \frac{[A \rightarrow \alpha.a\beta, i, j, e_1] \qquad [a, j, k, e_2]}{[A \rightarrow \alpha a.\beta, i, k, e_1 + e_2]}$$

4.3 Correctness of the Obtained Parsers

The robust transformation function we have just described maps prediction-completion parsing systems to error-repair parsing systems. However, in order for this transformation to be useful, we need it to guarantee that the generated robust parsers will be correct under certain conditions. This is done by the following two theorems:

Let $d : Trees'(G) \times Trees'(G) \rightarrow \mathbb{N}$ be a distance function, and let $\mathbb{S} = (\mathcal{I}, \mathcal{K}, D)$ be a prediction-completion parsing system.

Theorem 1. If $(\mathcal{I}, \mathcal{K}, D)$ is sound, every deduction step d in a predictive step set $D_i \subseteq D$ has a nonempty consequent, and for every deduction step d in a yield union step set $D_i \subseteq D$ with antecedents $[p_1, i_0, i_1], [p_2, i_1, i_2], \ldots, [p_m, i_{m-1}, i_m], [p'_1, j_1, k_1]$, $[p'_2, j_2, k_2], \ldots, [p'_n, j_n, k_n]$ and consequent $[p_c, i_0, i_m]$ there exists a function $C_d : Trees'(G)^n \rightarrow Trees'(G)^n$ (tree combination function) such that:

- If $(t_1, \ldots, t_m)$ is a tuple of trees in $Trees(G)$ such that $t_w \in [p_w, i_{w-1}, i_w](1 \leq w \leq m)$, then $C_d(t_1, \ldots, t_m) \in [p_c, i_0, i_m]$.
- If $(t_1, \ldots, t_m)$ is a tuple of trees in $Trees(G)$ such that $t_w \in [p_w, i_{w-1}, i_w](1 \leq w \leq m)$, and $(t'_1, \ldots, t'_m)$ is a tuple of contiguous yield trees such that $d(t'_w, t_w) = e_w(1 \leq i \leq m)$, then $d(C_d(t_1, \ldots, t_m), C_d(t'_1, \ldots, t'_m)) = \Sigma_{w=1}^{m} e_w$, and $C_d(t'_1, \ldots, t'_m)$ is a contiguous yield tree with $yield_m(C_d(t'_1, \ldots, t'_m)) = yield_m(t'_1)yield_m(t'_2) \ldots yield_m(t'_m)$.

Then $\mathcal{R}(\mathcal{I}, \mathcal{K}, D)$ is sound.

Theorem 2. If $(\mathcal{I}, \mathcal{K}, D)$ is complete, then $\mathcal{R}(\mathcal{I}, \mathcal{K}, D)$ is complete.

Note that the condition about the existence of tree combination functions in theorem 1 is usually straightforward to verify. A yield union step set normally combines two partial parse trees in $Trees(G)$ in some way, producing a new partial parse tree in $Trees(G)$ covering a larger portion of the input string. In practice, the existence of a tree combination function just means that we can also combine in the same way trees that are not in $Trees(G)$, and that the obtained tree's minimal distance to a tree in $Trees(G)$ is the sum of those of the original trees. For example, in the case of the Earley $Completer$ step, it is easy to see the function that maps a pair of trees of the form $A(\alpha(...)B\beta)$ and $B(\gamma(...))$ to the combined tree $A(\alpha(...)B(\gamma(...))\beta)$ obtained by adding the children of B in the second tree as children of B in the first tree is a valid combination function.

4.4 Proving Correctness

For space reasons, we cannot show the full proofs for the theorems that ensure that our error-repair transformation preserves correctness. Thus, we will just provide a brief outline on how the proofs are done.

Proof of Theorem 1 (Preservation of the soundness). We define a correct item in an error-repair parsing system for a particular string $a_1 \dots a_n$ as an approximate item $[p, i, j, e]$ containing an approximate tree (t, e) such that t is a tree with $yield_m(t) = \underline{a}_{i+1} \dots \underline{a}_j$; and we prove that the robust transformation of a given schema $\mathbb{S}$ is sound (all valid final items are correct) by proving the stronger claim that all valid items are correct. To prove this, we show that if the antecedents of a deduction step are correct, then the consequent is also correct. If we call D' the set of deduction steps in $\mathcal{R}(\mathbb{S})$, this is proven by writing D' as an union of step sets, and proving it separately for each step set. In the particular case of the steps D'_i coming from yield union step sets D_i in the original schema, the combination function is used to obtain a tree allowing us to prove that the consequent is correct. In the rest of the cases, this tree is obtained directly.

Proof of Theorem 2 (Preservation of the completeness). The proof for this theorem uses the fact that any final item in the approximate item set associated to $\mathcal{R}(\mathbb{S})$ can be written as $[p, i, j, e]$ (formally, there is a surjective *error-repair representation function* r' such that any approximate item in the set can be written as $r'(p, i, j, e)$; we use this function to formally define the approximate item set associated to $\mathcal{R}(\mathbb{S})$). Therefore, proving completeness is equivalent to proving that every correct final item of the form $[p, i, j, e]$ is valid. This is proven by induction on the distance e.

The base case of this induction ($e = 0$) is proven by mapping items with distance 0 to items from the item set $\mathcal{I}$ corresponding to the original non-error-repair parser, and using the fact that this original parser is complete.

For the induction step, we suppose that the proposition holds for a distance value e, and prove it for $e + 1$. In order to do this, we take an arbitrary correct final item with associated distance $e+1$ and prove that it is valid. This is done by taking an approximate tree $(t, e+1)$ from this item and using it to build an approximate tree (t', d) whose yield only differs from $yield(t)$ in a single substitution, insertion or deletion operation. For each of the three operations, we build an instantiated parsing system where the item containing (t', e) is correct. By the induction hypothesis, this item is also valid in that system, so each case of the induction step is reduced to proving that validity of the item containing (t', e) in the constructed system implies the validity of $(t, e + 1)$ in the original system. This is proven by building suitable mappings between the items of both systems so that deductions are preserved and the item associated to (t', e) is mapped to the one associated to $(t, e+1)$. These mappings are different for each case (substitution, insertion, deletion) of the proof.

4.5 Simplifying the Resulting Parsers

The error-repair parsers obtained by using our transformation are guaranteed to be correct if the original standard parsers meet some simple conditions, but the extra steps

added by the transformation can make the semantics of the obtained parsers somewhat hard to understand. Moreover, the $SubstitutionHyp$ and $DeletionHyp$ steps would negatively affect performance if implemented directly in a deductive engine.

However, once we have the error-repair transformation of a parser, we can apply some standard simplifications to it in order to obtain a simpler, more efficient one which will generate the same items except for the error hypotheses. That is, we can bypass items of the form $[p, i, j, e]$. In order to do this, we remove the steps that generate this kind of items and, for each step requiring $[a, i, j, e]$ as an antecedent, we change this requirement to the set of hypotheses of the form $[b, i', j']$ needed to generate such an item from the error hypothesis steps.

With this simplification, the error-repair transformation for an Earley parser is as follows:

$$TransformedEarleyInitter: \frac{}{[S \to .\alpha, 0, 0, 0]}\ S \to \alpha \in P$$

$$TransformedEarleyPredictor: \frac{[A \to \alpha.B\beta, i, j, e]}{[B \to .\gamma, j, j, 0]}\ B \to \gamma \in P$$

$$TransformedEarleyCompleter: \frac{[A \to \alpha.B\beta, i, j, e_1] \qquad [B \to \gamma., j, k, e_2]}{[A \to \alpha B.\beta, i, k, e_1 + e_2]}$$

$$GeneralSubstitutionEarleyScanner: \frac{[A \to \alpha.a\beta, i, j, e]}{[A \to \alpha a.\beta, i, k, e + k - j]}\ k \geq j + 1$$

$$GeneralDeletionEarleyScanner: \frac{[A \to \alpha.a\beta, i, j, e]}{[A \to \alpha a.\beta, i, k, e + k - j + 1]}\ k \geq j$$

$$GeneralEarleyScanner1: \frac{[A \to \alpha.a\beta, 0, 0, e] \qquad [a, w - 1, w]}{[A \to \alpha a.\beta, 0, k, e + k - 1]}\ 0 < w \leq k$$

$$GeneralEarleyScanner2: \frac{[A \to \alpha.a\beta, i, j, e] \qquad [a, j, j + 1]}{[A \to \alpha a.\beta, i, k, e + k - j - 1]}\ k \geq j + 1$$

where the steps named "scanner" are obtained from the Earley $Scanner$ step after applying the simplification in order to bypass items of the form $[p, i, j, e]$.

Note that, if we choose the alternative transformation that does not use $Combiner$ steps (see section 4.1), the obtained parser would be the one described in [9], which is a *step refinement* ([13]) of the parsing schema described in this section.

5 Empirical Results

In order to test the results of our transformation in practice, we have used the system described in [4] to execute the error-repair version of the Earley parser explained above. We have executed the schema in two different modes: obtaining all the valid final items with minimal distance (*global error repair*); and restricting repair steps on errors

Table 1. Performance results for the global and regional error-repair parsers when parsing sentences from the ATIS test set. Each row corresponds to a value of the minimal parsing distance (or error count).

Minimal Distance	# Sentences	Avg. Length	Avg. Items (Global)	Avg. Items (Regional)	Improvement
0	70	11.04	37,558	37,558	0%
1	24	11.63	194,249	63,751	65.33%
2	2	18.50	739,705	574,534	22.33%
3	2	14.50	1,117,123	965,137	13.61%
>3	none	n/a	n/a	n/a	n/a

to regions surrounding the errors in order to obtain a single optimal solution, as explained in [16] (*regional error repair*).

The grammar and sentences used for testing are from the DARPA ATIS3 system. Particularly, we have used the same test sentences that were used by [10]. This test set is suitable for error-repair parsing, since it comes from a real-life application and it contains ungrammatical sentences. In particular, 28 of the 98 sentences in the set are ungrammatical. By running our error-repair parsers, we find that the minimal edit distance to a grammatical sentence is 1 for 24 of them (i.e., these 24 sentences have a possible repair with a single error), 2 for two of them, and 3 for the remaining two.

Table 1 shows the average number of items generated by our parsers with respect to the minimal parsing distance of the inputs. As we can see, regional parsing reduces item generation by a factor of three in sentences with a single error. In sentences with more than one error the improvements are smaller. However, we should note that parsing time grows faster than the number of items generated, so these relative improvements in items translate to larger relative improvements in runtime. Moreover, in practical settings we can expect sentences with several errors to be less frequent than sentences with a single error, as in this case. Thus, regional error-repair parsers are a good practical alternative to global ones.

6 Conclusions and Future Work

We have presented a method to transform context-free grammar parsers (expressed as parsing schemata) into error-repair parsers. Our transformation guarantees that the resulting error-repair parsers are correct as long as the original parsers verify certain conditions. It is easy to see that popular grammar parsers such as CYK, $Earley$ or $Left-Corner$ verify these conditions, so this method can be applied to create a wide range of error-repair parsers. The transformation can be applied automatically, and its results are error-repair parsing schemata that can be executed by means of a deductive engine. Therefore, a system as the one described in [4] can be extended to automatically generate implementations of robust parsers from standard parsing schemata.

The method is general enough to be applied to different grammatical formalisms. Currently, we are applying it to parsers for tree adjoining grammars.

References

1. Cerecke, C.: Repairing syntax errors in LR-based parsers. Australian Computer Science Communications 24(1), 17–22 (2002)
2. Corchuelo, R., Pérez, J.A., Ruiz, A., Toro, M.: Repairing Syntax Errors in LR Parsers. ACM Transactions on Programming Languages and Systems 24(6), 698–710 (2002)
3. Earley, J.: An efficient context-free parsing algorithm. Communications of the ACM 13(2), 94–102 (1970)
4. Gómez-Rodríguez, C., Vilares, J., Alonso, M.A.: A compiler for parsing schemata. Software: Practice and Experience, doi:10.1002/spe.904 (forthcoming)
5. Grune, D., Jacobs, C.J.H.: Parsing Techniques. A Practical Guide, 2nd edn. Springer Science+Business Media, Heidelberg (2008)
6. Kasper, W., Kiefer, B., Krieger, H.U., Rupp, C.J., Worm, K.L.: Charting the depths of robust speech parsing. In: Proc. of ACL 1999, Morristown, NJ, USA, pp. 405–412 (1999)
7. Kim, I.-S., Choe, K.-M.: Error Repair with Validation in LR-based Parsing. ACM Transactions on Programming Languages and Systems 23(4), 451–471 (2001)
8. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10(8), 707–710 (1966)
9. Lyon, G.: Syntax-directed least-errors analysis for context-free languages: a practical approach. Commun. ACM 17(1), 3–14 (1974)
10. Moore, R.C.: Improved left-corner chart parsing for large context-free grammars. In: Proc. of the 6th IWPT, Trento, Italy, pp. 171–182 (2000)
11. Perez-Cortes, J.C., Amengual, J.C., Arlandis, J., Llobet, R.: Stochastic error-correcting parsing for OCR post-processing. In: ICPR 2000: Proceedings of the International Conference on Pattern Recognition, Washington, DC, USA, p. 4405. IEEE Computer Society, Los Alamitos (2000)
12. Shieber, S.M., Schabes, Y., Pereira, F.C.N.: Principles and implementation of deductive parsing. Journal of Logic Programming 24(1–2), 3–36 (1995)
13. Sikkel, K.: Parsing Schemata — A Framework for Specification and Analysis of Parsing Algorithms. Springer, Heidelberg (1997)
14. Sikkel, K.: Parsing schemata and correctness of parsing algorithms. Theoretical Computer Science 199(1–2), 87–103 (1998)
15. van der Spek, P., Plat, N., Pronk, N.: Syntax Error Repair for a Java-based Parser Generator. ACM SIGPLAN Notices 40(4), 47–50 (2005)
16. Vilares, M., Darriba, V.M., Ribadas, F.J.: Regional least-cost error repair. In: Yu, S., Păun, A. (eds.) CIAA 2000. LNCS, vol. 2088, pp. 293–301. Springer, Heidelberg (2001)
17. Vilares, M., Darriba, V.M., Vilares, J., Ribadas, F.J.: A formal frame for robust parsing. Theoretical Computer Science 328, 171–186 (2004)

Topic-Focus Articulation from the Semantic Point of View

Marie Duží

VSB-Technical University Ostrava, Czech Republic
marie.duzi@vsb.cz

Abstract. In the paper we show that sentences differing only in topic-focus articulation have different logical structures, and thus they also have different truth-conditions. Our analysis is based on the procedural semantics of Transparent Intensional Logic (TIL) assigning to sentences hyperpropositions as their structured meanings. We analyse the phenomena of presupposition connected with a topic and allegation triggered by a focus of a sentence so that relevant consequences can be formally derived.

1 Introduction

In the invited talk [4], presented at CICLing 2008, Eva Hajičová argued that the problem of topic-focus articulation is a semantic, rather than pragmatic problem. We agree, and to put her arguments still on a more solid ground, we are going to demonstrate the semantic nature of the topic-focus difference by its *logical* analysis. To this end we apply procedural semantics of Transparent Intensional Logic (TIL) and assign (algorithmically structured) procedures to expressions as their meaning. As a result, we furnish sentences differing only in the topic-focus articulation with different structured meanings producing different propositions.

As sample sentences we analyse (slightly modified) examples adduced in [4]. Moreover, we present general schemata of a logical structure of arguments according whether the phenomena of presupposition or allegation are the case. These relations are defined in [4], where Hajičová shows that the clause standing in the topic often induces the case of presupposition, whereas a focus-clause is connected with allegation. If a presupposition Q of a given proposition P is not true, then P as well as negated P have no truth-value. In other words, Q is entailed both by P and non-P. On the other hand, if Q is an allegation of P, then P entails, but does not presuppose, Q. If non-P is the case, we cannot deduce anything about the truth of Q. Since our logic is a hyper-intensional logic of *partial functions*, we analyse sentences with presuppositions in a natural way. We furnish them with hyper-propositions that produce propositions with truth-value gaps. Having a rigorous, fine-grained analysis at our disposal, we can easily infer the relevant consequences. Thus our logic meets the philosophical and linguistic desiderata formulated in [4].

The paper is organised as follows. After briefly introducing TIL philosophy and its basic notions in Section 2, the main Section 3 describes the method of analysing sentences with topic-focus articulation. Concluding Section 4 presents the direction of

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 220–232, 2009.

future research and a few notes on TIL implementation via the *TIL-Script* functional programming language.

2 TIL in Brief

Transparent Intensional Logic (TIL) is a system with procedural semantics primarily designed for the logical analysis of natural language.[1] Traditional non-procedural theories of formal semantics are less or more powerful logical languages, from the extensional languages based on the first-order predicate logic paradigm, through some hybrid systems up to intensional (modal or epistemic) logics. Particular systems are well suited for analysing restricted sublanguages. Yet some special hard cases like attitudes, anaphora, or the topic-focus articulation are stumbling blocks for all of them. On the other hand, TIL, due to its strong typing and procedural semantics, operates smoothly with the three levels of granularity: the extensional level of truth-functional connectives, the intensional level of modalities and finally the hyper-intensional level of attitudes.[2] The sense of a sentence is an algorithmically structured *construction* of a proposition denoted by the sentence. The denoted proposition is a flat mapping with the domain of possible worlds. Our motive for working 'top-down' has to do with anti-contextualism: any given unambiguous term or expression (even one involving indexicals or anaphoric pronouns) expresses the same construction as its sense (meaning) in whatever sort of context the term or expression is embedded within. And the meaning of an expression determines the respective denoted entity (if any), but not vice versa. Thus we strictly distinguish between a procedure and its product, and between a function and its value.

TIL *constructions* are uniquely assigned to expressions as their algorithmically structured senses. When assigning a construction to an expression as its meaning, we specify *procedural know-how*, which must not be confused with the respective *performatory know-how*.[3] Understanding a sentence *S* involves procedural know-how; one can spell out instructions for evaluating the truth-conditions of *S* in any state-of-affairs *w* at any time *t*. But, of course, one can know how to evaluate *S* without actually being able to do so—that is, without having the performatory skills that enable him to determine the truth-value of *S* in a particular possible world *W* at time *T*. Intuitively, construction *C* is a *procedure* (a generalised algorithm), that consists of sub-instructions (constituents) that must be executed in order to execute *C*. It is an instruction on how to proceed in order to obtain the output entity given some input entities.

There are two kinds of constructions, atomic and compound (molecular). Atomic constructions (*Variables* and *Trivializations*) do not contain any other constituent but themselves; they supply objects (of any type) on which compound constructions operate. *Variables x, y, p, q, …,* construct objects dependently on a valuation; they *v*-construct. *Trivialisation* of an object *X* (of any type, even a construction), in symbols ${}^{0}\boldsymbol{X}$, constructs simply *X* without the mediation of any other construction. *Compound* constructions, which consist of other constituents, are *Composition* and *Closure.*

[1] See, for instance, [1], [6], [7], [11] and [12].
[2] For TIL analysis of anaphoric references, see, e.g., [2], and for attitudes [3].
[3] See [8], pp.6-7.

Composition **[*F A1…An*]** is the instruction to apply a function *f* (*v*-constructed by *F*) to a tuple argument A (*v*-constructed by $A_1…A_n$).[4] Thus it *v*-constructs the value of *f* at A, if the function *f* is defined at A, otherwise the Composition is *v-improper*, i.e., it *fails* to *v*-construct anything. *Closure* **[λ*x1…xn X*]** is the instruction to *v*-construct a function by abstracting over variables $x_1,…,x_n$ in the ordinary manner of λ-calculi. Finally, higher-order constructions can be used twice over as constituents of composed constructions. This is achieved by a fifth construction called *Double Execution*, ^{2}***X***, that behaves as follows: If *X* *v*-constructs a construction *X*', and *X'* *v*-constructs an entity *Y*, then ^2X *v*-constructs *Y*; otherwise ^2X is *v-improper*, it *fails* to *v*-construct anything.

TIL constructions, as well as the entities they construct, all receive a type. The formal ontology of TIL is bi-dimensional; one dimension is made up of constructions, the other dimension encompasses non-constructions. On the ground level of the type-hierarchy, there are non-constructional entities unstructured from the algorithmic point of view belonging to a *type of order 1*. Given a so-called *epistemic (or 'objectual') base* of *atomic types* (**o**-truth values, **ι**-individuals, **τ**-time moments / real numbers, **ω**-possible worlds), the induction rule for forming functional types is applied: where α, $\beta_1,…,\beta_n$ are types of order 1, the set of partial mappings from $\beta_1 \times…\times \beta_n$ to α, denoted *(α $\beta_1…\beta_n$)*, is a type of order 1 as well.[5] Constructions that construct entities of order 1 are *constructions of order 1*. They belong to a *type of order 2*, denoted by $*_1$. This type $*_1$ together with atomic types of order 1 serves as a base for the induction rule: any collection of partial mappings, type *(α $\beta_1…\beta_n$)*, *involving* $*_1$ in their domain or range is a *type of order 2*. Constructions belonging to a type $*_2$ that identify entities of order 1 or 2, and partial mappings involving such constructions, belong to a *type of order 3*. And so on *ad infinitum.*

The sense of an empirical expression is a *hyper-intension*, i.e., a construction that produces a PWS intension.

(α-)intensions are members of type (αω), i.e., functions from possible worlds to an arbitrary type α.

(α-)extensions are members of a type α, where α is not equal to (βω) for any β, i.e., extensions are not functions with the domain of possible worlds.

Intensions are frequently functions of a type ((ατ)ω), i.e., functions from possible worlds to *chronologies* of the type α (in symbols: $\alpha_{\tau\omega}$), where a chronology is a function of type (ατ).

Some important kinds of intensions are:

Propositions, type $o_{\tau\omega}$. They are denoted by empirical sentences.

Properties of members of a type α, or simply *α-properties,* type $(o\alpha)_{\tau\omega}$.[6] General terms, some substantives, intransitive verbs ('student', 'walking') denote properties, mostly of individuals.

---

[4] We treat functions as mappings, i.e., set-theoretical objects, unlike the *constructions* of functions.

[5] TIL is an open-ended system. The above epistemic base {o, ι, τ, ω} was chosen, because it is apt for natural-language analysis, but the choice of base depends on the area to be analysed.

[6] We model α-sets and $(\alpha_1…\alpha_n)$-relations by their characteristic functions of type (oα), $(o\alpha_1…\alpha_n)$, respectively. Thus an α-property is an empirical function that dependently on states-of-affairs (τω) picks-up a set of α-individuals, the population of the property.

Relations-in-intension, type $(o\beta_1\ldots\beta_m)_{\tau\omega}$. For example transitive empirical verbs ('like', 'worship'), also attitudinal verbs denote these relations.
α-roles, offices, type $\alpha_{\tau\omega}$, where $\alpha \neq (o\beta)$. Frequently $\iota_{\tau\omega}$. Often denoted by concatenation of a superlative and a noun ('the highest mountain').

An object A of a type α is denoted A/α. That a construction $C/*_n$ *v*-constructs an object of type α is denoted $C \rightarrow_v \alpha$. We use variables *w, w1, …* as *v*-constructing elements of type ω (possible worlds), and *t, t1, …* as *v*-constructing elements of type τ (times). If $C \rightarrow \alpha_{\tau\omega}$ *v*-constructs an α-intension, the frequently used Composition of the form $[[Cw]t]$, the intensional descent of an α-intension, is abbreviated as $\boldsymbol{C_{wt}}$.

We invariably furnish expressions with their procedural structural meanings, which are explicated as TIL constructions. The analysis of an expression thus consists in discovering the logical construction encoded by the expression. *TIL method of analysis* consists of three steps:[7]

1) *Type-theoretical analysis*, i.e., assigning types to the objects that receive mention in the analysed sentence.
2) *Synthesis*, i.e., combining constructions of the objects *ad* (1) in order to construct the proposition of type $o_{\tau\omega}$ denoted by the whole sentence.
3) *Type-Theoretical checking.*

As an example we are going to analyse the proverbial sentence "The King of France is bald". The sentence talks about the office of the King of France (topic) ascribing to the individual (if any) that occupies this office the property of being bald (focus). Thus there is a presupposition that the King of France exists, i.e., that the office is occupied. If not, then the proposition denoted by the sentence has no truth-value.[8] This fact has to be revealed by our analysis. Here is how.

Ad (1) *King_of*/$(\iota\iota)_{\tau\omega}$; *France*/$\iota$; *King_of_France*/$\iota_{\tau\omega}$; *Bald*/$(o\iota)_{\tau\omega}$.

Ad (2) Now we have to combine *constructions* of the objects *ad (1)* in order to construct the proposition of type $o_{\tau\omega}$ denoted by the whole sentence. The simplest constructions of the above objects are their Trivialisations: ^{0}*King_of*, ^{0}*France*, ^{0}*Bald.* The attribute *King_of* has to be extensionalised first *via* Composition $^0King\_of_{wt}$, and then applied to *France*, $[^0King\_of_{wt}\ ^0France]$. Finally by abstracting over *w,t* we obtain the office: $\lambda w\lambda t\,[^0King\_of_{wt}\ ^0France]$. But the property of being bald cannot be ascribed to an individual office. Rather, it is ascribed to an individual occupying the office. Thus the office has to be subjected to the intensional descent to *v*-construct an individual occupying the office (if any): $\lambda w\lambda t\,[^0King\_of_{wt}\ ^0France]_{wt}$. The property itself has to be extensionalised as well, $^0Bald_{wt}$. Composing these two constructions, we obtain a truth-value **T**, **F**, or nothing, according whether the King of France is or is not bald, or does not exist.[9] Finally, abstracting over *w, t,* we construct the proposition:

[7] For details see, e.g., [7].
[8] On our approach this does not mean that the sentence is meaningless. The sentence has its sense, namely the instruction how to evaluate in any possible world *w* at any time *t* its truth-conditions. Just that if we evaluate this instruction in such a state-of-affairs where the King of France does not exist, the process of evaluation yields a truth-value gap.
[9] For details on predication of properties see [5].

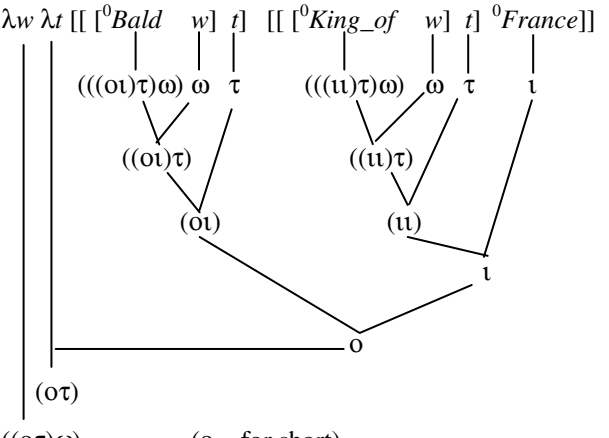
$$\lambda w\lambda t\,[{}^0Bald_{wt}\,\lambda w\lambda t\,[{}^0King\_of_{wt}\,{}^0France]_{wt}].$$

This Closure can be equivalently simplified into

$$\lambda w\lambda t\,[{}^0Bald_{wt}\,[{}^0King\_of_{wt}\,{}^0France]].$$

Gloss. In *any* world at *any* time ($\lambda w\lambda t$) do these: First, find out who is the King of France by applying the extensionalised attribute *King_of* to *France* ($[{}^0King\_of_{wt}\,{}^0France]$). If there is none, then finish with the truth-value gap because the Composition $[{}^0King\_of_{wt}\,{}^0France]$ is *v*-improper. Otherwise, check whether the so-obtained individual has the property of being bald ($[{}^0Bald_{wt}\,[{}^0King\_of_{wt}\,{}^0France]]$). If so, then **T**, otherwise **F**.

Ad (3). Drawing a type-theoretical structural tree,[10] we check whether particular constituents of the above Closure are combined in the correct way.



So much for the semantic schema of TIL logic. Now we are going to apply this formal apparatus to analyse the topic-focus distinction.

3 Topic-Focus Articulation

In this section we propose the method of *logically* analysing sentences with topic-focus articulation. The input for our analysis is the tectogrammatical representation of sentences which reflects the topic-focus articulation as described in [4, pp. 254-259]. When used in a communicative act, the sentence communicates something (the focus *F*) about something (the topic *T*). Thus the schematic structure of a sentence is *F(T).* The topic of a sentence *S* often generates a presupposition of *S* which is entailed both by *F(T)* and $\neg F(T)$. To start up, let us analyse some of the examples adduced in [4].

(1) All *John's children* are asleep.
(1') All *John's children* are not asleep.[11]

[10] See [1].
[11] In what follows we mark the topic of a sentence in italics.

According to Strawson (1) as well as (1') entail[12]

(2) John has children.

In other words, (2) is a *presupposition* of (1).[13] If each of John's children is asleep, then the sentence (1) is true and (1') false. If each of John's children is not asleep, then the sentence (1) is false and (1') is true. However, if John does not have any children, then (1) and (1') are neither true nor false.

However, applying a classical translation of (1) into the language of first-order predicate logic, we get

$$\forall x\,[JC(x) \supset S(x)].$$

But this formula is true under every interpretation assigning an empty set of individuals to the predicate *JC*. We need to apply a richer logical system in order to express the instruction how to evaluate the truth-conditions of (1) in the above described way.

Reformulating the above specification of the truth-conditions of (1) in a rather technical jargon of English, we get

If John has children *then* check whether all his children are asleep, *otherwise fail*.

Now we have to analyse particular constituents of this instruction. As always, we start with assigning types to the objects that receive mention in the sentence: *Child_of*$((o\iota)\iota)_{\tau\omega}$ – an empirical function that dependently on states-of-affairs (time/τ and possible world/ω) assigns to an individual (ι) a set of individuals ($o\iota$), its children; *John*/ι; *Sleep*/$(o\iota)_{\tau\omega}$; $\exists$/$(o(o\iota))$ – the existential quantifier that assigns to a non-empty set **T**, otherwise **F**; *All*/$((o(o\iota))(o\iota))$ – a restricted general quantifier that assigns to a given set the set of all its supersets.

The presupposition that John has children receives the analysis

$$\lambda w\lambda t\,[{}^0\exists\lambda x\,[[{}^0Child\_of_{wt}\;{}^0John]\;x]].$$

Gloss. The set $\lambda x\,[[{}^0Child\_of_{wt}\;{}^0John]\;x]$ is not empty. In what follows we will use an abbreviated notation without Trivialisation for the quantifiers $\forall$, $\exists$. Thus we have

$$\lambda w\lambda t\,\exists x\,[[{}^0Child\_of_{wt}\;{}^0John]\;x].$$

Now the literal analysis of "All John's children are asleep" is best obtained by using the restricted quantifier *All*. Composing the quantifier with the set of John's children, $[{}^0All\;[{}^0Child\_of_{wt}\;{}^0John]]$, we obtain the set of all supersets of John's children population in *w* at *t*. The sentence claims that the population of those who are asleep, ${}^0Sleep_{wt}$, is one of such supersets:

$$\lambda w\lambda t\,[[{}^0All\;[{}^0Child\_of_{wt}\;{}^0John]]\;{}^0Sleep_{wt}].$$

So far so good; yet there is a problem how to analyse the connective '*if-then-else*'. We cannot simply apply material implication '$\supset$'. For instance, it might seem that the instruction "If 5=5 then output 1 else output the result of 1 divided by 0" receives the analysis $[[[{}^05={}^05]\supset[n={}^01]]\wedge[\neg[{}^05={}^05]\supset[n=[{}^0Div\;{}^01\;{}^00]]]]$, where *n* is the outputted number. But the output of the above instruction should be the number 1 because

[12] See [10, in paticular p. 173ff.)

[13] As well as of (1'), of course.

the 'else-clause' is never executed. However, due to strict compositionality, the above analysis fails to produce anything, it is improper. The reason is this. The Composition $[{}^0Div\ {}^0 1\ {}^0 0]$ does not produce anything, it is improper because the division function has no value at <1,0>. Thus the Composition $[n = [{}^0Div\ {}^0 1\ {}^0 0]]$ is v-improper for any valuation v, because the identity relation = does not receive an argument, and so is any other Composition containing the improper Composition $[{}^0Div\ {}^0 1\ {}^0 0]$ as a constituent (partiality is strictly propagated up).

The schematic proper analysis of sentences of the form "If P then C else D" is

(*) $$ {}^2[{}^0\iota\lambda c\ [[P \supset [c={}^0C]] \wedge [\neg P \supset [c={}^0D]]]]. $$

Types: $P \to o$ – the condition of the choice between the execution of C or D, $C/*_n$, $D/*_n$; variable $c \to *_n$; $\iota/(*_n(o*_n))$ – the singulariser function that associates a singleton set of constructions with the only construction that is an element of this singleton, otherwise (i.e., if the set is empty or many-valued) it is undefined.

The analysis decomposes into a two-phase instruction (therefore *Double Execution*). First, the Composition $[[P \supset [c={}^0C]] \wedge [\neg P \supset [c={}^0D]]]$ is the instruction to make a choice between C and D. If P constructs **T** then the variable c is instantiated to the *construction C*, and if P constructs **F** then the variable c is instantiated to the *construction D*. In any case the set constructed by $\lambda c\ [[P \supset [c={}^0C]] \wedge [\neg P \supset [c={}^0D]]]$ is a singleton and the singulariser ι returns as its value either the construction C or D. Second, the chosen construction is executed.

In our case the condition P is that John has children, $[\exists x\ [[{}^0Child\_of_{wt}\ {}^0John]\ x]]$, the construction C that is to be executed if P yields **T** is $[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}]$, and if P yields **F** then no construction D is to be chosen. Thus the analysis of the sentence (1) comes down to this Closure:

(1*) $$ \lambda w \lambda t\ {}^2[{}^0\iota\lambda c\ [\exists x\ [[{}^0Child\_of_{wt}\ {}^0John]\ x] \supset [c={}^0[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}] \wedge [\neg\exists x\ [[{}^0Child\_of_{wt}\ {}^0John]\ x] \supset {}^0\mathbf{F}]]] $$

The evaluation of (1*) in any world/time $\langle w,t\rangle$ depends on whether the presupposition condition $\exists x\ [[{}^0Child\_of_{wt}\ {}^0John]\ x]$ is true in $\langle w,t\rangle$.

a) $\exists x\ [[{}^0Child\_of_{wt}\ {}^0John]\ x] \to_v \mathbf{T}$.
Then $\lambda c\ [{}^0\mathbf{T} \supset [c={}^0[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}] \wedge [{}^0\mathbf{F} \supset {}^0\mathbf{F}]] = \{{}^0[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}]\}$. Hence
${}^2[\iota\lambda c\ [{}^0\mathbf{T} \supset [c={}^0[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}] \wedge [{}^0\mathbf{F} \supset {}^0\mathbf{F}]] = {}^{20}[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}] = [[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}]$.

b) $\exists x\ [[{}^0Child\_of_{wt}\ {}^0John]\ x] \to_v \mathbf{F}$.
Then $\lambda c\ [{}^0\mathbf{F} \supset [c={}^0[[{}^0All\ [{}^0Child\_of_{wt}\ {}^0John]]\ {}^0Sleep_{wt}] \wedge [{}^0\mathbf{T} \supset {}^0\mathbf{F}]] = \lambda c\ {}^0\mathbf{F}$.
The v-constructed set is *empty*. Hence, ${}^2[\iota\lambda c\ {}^0\mathbf{F}]$ is v-improper, *fails*.

To generalise, we now present a ***general analytic schema*** of sentences with topic-focus articulation, i.e., sentences of the schematic form *F(T)*, *¬F(T)*.

Let *P(T)* be the presupposition induced by the topic *T*. Then the sentence of the form *F(T)* expresses an instruction of the respective form

If *P(T)* then *F(T)* else *Fail*.

The corresponding schematic TIL construction is

(**) $\lambda w \lambda t\, {}^2[\iota\lambda c\, [[P(T) \supset [c={}^0[F\,T]]] \wedge [\neg P(T) \supset {}^0\mathbf{F}]]]$.

The evaluation in any $\langle w,t\rangle$-pair depends on the value v-constructed by $P(T)$.

(a) $P(T) \rightarrow_v \mathbf{T}$. Then
$${}^2[\iota\lambda c\, [[{}^0\mathbf{T} \supset [c={}^0[F\,T]]] \wedge [{}^0\mathbf{F} \supset {}^0\mathbf{F}]]] = {}^2[{}^0[F\,T]] = [F\,T]$$

(b) $P(T) \rightarrow_v \mathbf{F}$. Then
${}^2[\iota\lambda c\, [[{}^0\mathbf{F} \supset [c={}^0[F\,T]]] \wedge [{}^0\mathbf{T} \supset {}^0\mathbf{F}]]]$ is v-improper.

(c) $P(T)$ is v-improper, then
${}^2[\iota\lambda c\, [[P(T) \supset [c={}^0[F\,T]]] \wedge [\neg P(T) \supset {}^0\mathbf{F}]]]$ is v-improper.

Similarly for the sentence of the form $\neg F(T)$; the respective schematic analysis is

(***) $\lambda w \lambda t\, {}^2[\iota\lambda c\, [[P(T) \supset [c={}^0[\neg[F\,T]]]] \wedge [\neg P(T) \supset {}^0\mathbf{F}]]]$.

Another phenomenon we encounter when analysing sentences with topic-focus articulation is *allegation*.[14] Consider another group of sample sentences.

(3) *The King of France* visited London yesterday.
(3') *The King of France* did not visit London yesterday.

The sentences (3) and (3') talk about the (actual and current) King of France (the topic), ascribing to him the property of having (not having) visited London yesterday (the focus). Thus both the sentences have the presupposition that the King of France actually exists *now*. If it is not so, then none of the propositions expressed by (3) and (3') have any truth-value. The situation is different in case of sentences (4) and (4'):

(4) *London* was visited by the King of France yesterday.
(4') *London* was not visited by the King of France yesterday.

Now the property (in focus) of having been visited by the King of France yesterday is predicated of London (the topic). The existence of the King of France (now) is not presupposed by (4), and thus also not by (4'), of course. The sentences can be read as "Among the visitors of London was (was not) yesterday the King of France". The existence of the King of France *yesterday* is *implied*, but *not presupposed*, by (4).

To describe the difference between the cases such as (3) and (4), Hajičová introduces the so-called *allegation.* While (i) *presupposition* is characterised as an assertion A entailed by an assertion carried by a sentence S, and also by the negation of S, (ii) an *allegation* is an assertion A entailed by an assertion carried by a sentence S, but the negative counterpart of S entails neither A nor its negation. Schematically,

(i) $S \models A$ and $\neg S \models A$ (A is a ***presupposition*** of S);
Corollary: If $\neg A$ then *neither* S *nor* $\neg S$ have any truth-value.
(ii) $S \models A$ and *neither* $(\neg S \models A)$ *nor* $(\neg S \models \neg A)$ (***allegation***).

Our analyses reflect these conditions. Let *Yesterday*/$((o\tau)\tau)$ be the function that associates a given time t with a time-interval (that is yesterday with respect to t); *Visit*/$(o\iota)_{\tau\omega}$; *King_of*/$(\iota)_{\tau\omega}$; *France*/ι.
Then the sentences (3), (3') express

(3*) $\lambda w \lambda t\, [\lambda x\, \exists t'[[[{}^0Yesterday\ t]\ t'] \wedge [{}^0Visit_{wt\cdot}\ x\ {}^0London]]\ [{}^0King\_of_{wt}\ {}^0France]]$

[14] See [4, pp. 248-249].

(3'*) $\lambda w\lambda t\,[\lambda x\,[\exists t'[[[{}^0Yesterday\ t]\ t'] \wedge \neg[{}^0Visit_{wt'}\ x\ {}^0London]]\ [{}^0King\_of_{wt}\ {}^0France]]$

In such a $\langle w,t\rangle$-pair in which the King of France does not exist both the propositions constructed by (3*) and (3'*) have no truth-value, because the Composition $[{}^0King\_of_{wt}\ {}^0France]$ is *v*-improper. On the other hand, the sentences (4), (4') express

(4*) $\lambda w\lambda t\ \exists t'[[[{}^0Yesterday\ t]\ t'] \wedge [{}^0Visit_{wt'}\ [{}^0King\_of_{wt'}\ {}^0France]\ {}^0London]]$
(4'*) $\lambda w\lambda t\ \exists t'[[[{}^0Yesterday\ t]\ t'] \wedge \neg[{}^0Visit_{wt'}\ [{}^0King\_of_{wt'}\ {}^0France]\ {}^0London]]$

Now in such a $\langle w,t\rangle$-pair in which the proposition constructed by (4*) is true, the Composition $\exists t'[[[{}^0Yesterday\ t]\ t'] \wedge [{}^0Visit_{wt'}\ [{}^0King\_of_{wt'}\ {}^0France]\ {}^0London]]$ *v*-constructs the truth-value **T**. This means that the second conjunct *v*-constructs **T** as well. Hence $[{}^0King\_of_{wt'}\ {}^0France]$ *v*-constructs **T**, which means that the King of France *existed in some time t'* belonging to *yesterday*. On the other hand, if the King of France did not exist at any time of yesterday, then the Composition $[{}^0King\_of_{wt'}\ {}^0France]$ is *v*-improper for any *t'* belonging to yesterday. Thus the time interval *v*-constructed by $\lambda t'[[[{}^0Yesterday\ t]\ t'] \wedge [{}^0Visit_{wt'}\ [{}^0King\_of_{wt'}\ {}^0France]\ {}^0London]]$, as well as by $\lambda t'[[[{}^0Yesterday\ t]\ t'] \wedge \neg[{}^0Visit_{wt'}\ [{}^0King\_of_{wt'}\ {}^0France]\ {}^0London]]$, is empty, and the existential quantifier takes this interval to the truth-value **F**. This is as it should be, because (4*) *only implies yesterday's existence* of the King of France but *does not presuppose* it.[15]

Note that here we utilised the singularity of the office of King of France, i.e., of the function of type $\iota_{\tau\omega}$. If the King of France does not exist in some world *W* at time *T*, the office is not occupied and the function does not have any value in *W* at *T*. Thus we did not have to explicitly specify the presupposition of (3) that the King exists using the schema (**). As explained above, due to partiality of the office constructed by $\lambda w\lambda t\ [{}^0King\_of_{wt}\ {}^0France]$ and compositionality, (3*) and (4*) behave as desired.

Consider now another example.

(5) *Our defeat* was caused by John.
(5') *Our defeat* was not caused by John.
(6) We were defeated.

Both (5) and (5') entail (6), because these two sentences are about our defeat. Thus (6) is a presupposition of (5) and (5'), and the schematic logical form of (5) is the instruction "If we were *Defeated* then it was *Caused by John*, else *Fail.*" Simplifying a bit by ignoring the indexical character of 'we', let the proposition that we were defeated be constructed by $[{}^0Defeat]$.[16]

[15] Using medieval terminology, we also say that the concept of the King of France occurs with *de re* supposition in (3) and (3'), and with *de dicto* supposition in (the τ-intensional context of) (4) and (4').

[16] If we want to take into account the indexical character of these sentences, we use free variable '*we*' and obtain *open* constructions that construct propositions only after a valuation of *we* is supplied by a context of utterance. Thus (6) expresses $\lambda w\lambda t\ [{}^0Defeated_{wt}\ we]$. However, this is irrelevant here, as well as the past tense used in the example.

Using the general schema,
(**) $\lambda w\lambda t\,{}^{2}[\iota\lambda c\,[[P(T) \supset [c={}^{0}[F\,T]]] \wedge [\neg P(T) \supset {}^{0}\mathbf{F}]]]$, we have $P(T)$ and $T = {}^{0}Defeat_{wt}$; $F = \lambda p\ [{}^{0}Cause_{wt}\ {}^{0}John\ p]$; $[F\ T] = [{}^{0}Cause_{wt}\ {}^{0}John\ {}^{0}Defeat]$; $p \rightarrow o_{\tau\omega}$; $Defeat/o_{\tau\omega}$; $Cause/(o\iota o_{\tau\omega})_{\tau\omega}$; $John/\iota$.

As a result, (5) expresses

(5*) $\lambda w\lambda t\,{}^{2}[\iota\lambda c\,[[{}^{0}Defeat_{wt} \supset [c = {}^{0}[{}^{0}Cause_{wt}\ {}^{0}John\ {}^{0}Defeat]]] \wedge [\neg {}^{0}Defeat_{wt} \supset {}^{0}\mathbf{F}]]]$.

The evaluation of the truth-conditions in any w, at any t thus follows these cases:

a) ${}^{0}Defeat_{wt} \rightarrow_{v} \mathbf{T}$.
Then ${}^{20}[{}^{0}Cause_{wt}\ {}^{0}John\ {}^{0}Defeat] = [{}^{0}Cause_{wt}\ {}^{0}John\ {}^{0}Defeat]$;
b) ${}^{0}Defeat_{wt} \rightarrow_{v} \mathbf{F}$.
Then ${}^{20}[\iota\lambda c\ {}^{0}\mathbf{F}] \rightarrow$ *Fails.*

On the other hand, the truth-conditions of (7) and (7') are different.

(7) *John* caused our defeat.
(7') *John* did not cause our defeat.

Now the sentence (7) is about the topic John, ascribing to him the property that he caused our defeat (focus). Thus the scenario of truly asserting (7') can be, for instance, this. Though it is true that John has a reputation of a rather bad player, Paul was in a very good shape and we won. Or, the other scenario is thinkable. We were defeated not because of John but because the whole team performed badly.

The analyses of (7) and (7') are:

(7*) $\lambda w\lambda t\ [{}^{0}Cause_{wt}\ {}^{0}John\ {}^{0}Defeat]$
(7'*) $\lambda w\lambda t\ \neg[{}^{0}Cause_{wt}\ {}^{0}John\ {}^{0}Defeat]$

Yet if (7) is true then (6) can be validly inferred. In other words, (6) is entailed by (7) but not by (7'). This indicates that (6) is an allegation of (7) rather than a presupposition. As Hajičová says, the '(be)cause-clause' in focus triggers an allegation. To capture such truth-conditions, we need to refine the analysis. A plausible explication of this phenomenon is this: x is a cause of a proposition $p/o_{\tau\omega}$ iff p is true and if so then x affected p to be true. Schematically,

$$\lambda w\lambda t\ [{}^{0}Cause_{wt}\,x\,p] = \lambda w\lambda t\ [p_{wt} \wedge [p_{wt} \supset [{}^{0}Affect_{wt}\,x\,p]]].$$

Types: $Cause$, $Affect/(o\alpha o_{\tau\omega})_{\tau\omega}$; $x \rightarrow \alpha$, α – any type; $p/o_{\tau\omega}$.
If x is not a cause of p, then either p is not true or p is true but x did not affect p so that to be true:

$$\lambda w\lambda t\ \neg[{}^{0}Cause_{wt}\,x\,p] = \lambda w\lambda t\ [\neg p_{wt} \vee [p_{wt} \wedge \neg[{}^{0}Affect_{wt}\,x\,p]]].$$

Applying such an explication to (7), we get

(7**) $\lambda w\lambda t\ [{}^{0}Defeat_{wt} \wedge [{}^{0}Defeat_{wt} \supset [{}^{0}Affect_{wt}\ {}^{0}John\ {}^{0}Defeat]]]$,

which entails that we were defeated ($\lambda w\lambda t\ [{}^{0}True_{wt}\ {}^{0}Defeat]$), as it should be.

Similar phenomenon also crops up in case of seeking and finding. Imagine that one is referring on the tragedy in Dallas, November 22, 1963, by "The police were seeking the murderer of JFK but never found him". The sentence is ambiguous due to different topic-focus articulation.

(8) The *police* were seeking the murderer of JFK but never found him.
(9) The police were seeking *the murderer of JFK* but never found him.

The existence of the murderer of JFK is not presupposed by (8) unlike (9). The sentence (8) can be true in such states-of-affairs when JFK was not murdered, unlike the sentence (9). The latter can be reformulated in a more unambiguous way as "*The murderer of JFK* was looked for by the police but never found". This sentence expresses the construction

(9*) $\lambda w\lambda t\,[[{}^0Look\_for_{wt}\;{}^0Police\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]_{wt}] \wedge$
$\neg[{}^0Find^L_{wt}\;{}^0Police\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]_{wt}]].$

Types: *Look_for*, $Find^L/(o\iota\iota)_{\tau\omega}$; *Police*/$\iota$; $Murder\_of/(\iota\iota)_{\tau\omega}$; *JFK*/$\iota$.[17]
On the other hand, the analysis of (8) relates police to the *office* of the murderer rather than to its holder. The police aim at finding who the murderer is. Thus we have *Seek*, $Find^S/(o\iota\iota_{\tau\omega})_{\tau\omega}$; and (8) expresses:

(8*) $\lambda w\lambda t\,[[{}^0Seek_{wt}\;{}^0Police\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]] \wedge$
$\neg[{}^0Find^S_{wt}\;{}^0Police\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]]].$

If the police did not find the murderer then either the murderer did not exist or the search was not successful. However, if the foregoing search was successful, then it is true that police found the murderer

$$\lambda w\lambda t\,[{}^0Find^S_{wt}\;{}^0Police\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]]$$

and the murderer exists. Hence a successful search, i.e. *finding*, also *triggers an allegation*:

$$\frac{\lambda w\lambda t\,[{}^0Find^S_{wt}\;{}^0Police\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]]}{\lambda w\lambda t\,[{}^0Exist_{wt}\;[\lambda w\lambda t\,[{}^0Murder\_of_{wt}\;{}^0JFK]]]}$$

where $Exist/(o\iota_{\tau\omega})_{\tau\omega}$ is the property of an individual office of being occupied. In order to capture this allegation, we explicate finding after a foregoing search in a similar way as the above causing ($x \to \iota$; $c \to \iota_{\tau\omega}$; $Success\_Search/(o\iota\iota_{\tau\omega})_{\tau\omega}$):

$\lambda w\lambda t\,[{}^0Find^S_{wt}\;x\;c] = \lambda w\lambda t\,[[{}^0Exist_{wt}\;c] \wedge [[{}^0Exist_{wt}\;c] \supset [{}^0Success\_Search_{wt}\;x\;c]]]$
$\lambda w\lambda t\,\neg[{}^0Find^S_{wt}\;x\;c] = \lambda w\lambda t\,[\neg[{}^0Exist_{wt}\;c] \vee [[{}^0Exist_{wt}\;c] \wedge \neg[{}^0Success\_Search_{wt}\;x\;c]]].$

The last example we are going to analyse using our method concerns again a presupposition connected with the topic taken from [4].

(10) John only introduced *Bill* to Sue.
(11) John only introduced Bill to *Sue.*

Leaving aside possible disambiguation "John introduced only *Bill* to Sue" vs. "John introduced Bill only to *Sue*", (10) can be truly affirmed only in a situation when John did not introduce other people to Sue except for Bill. This is not the case of (11). This sentence can be true in a situation when John introduced other people to Sue, but the only person Bill was introduced to by John was Sue.

[17] For the sake of simplicity, past tense and anaphora reference are ignored. For a more detailed analysis, see, for instance, [2].

Recalling the general schema of analysis of sentences with presupposition

$$\lambda w \lambda t\,^2[\iota\lambda c\ [[P(T) \supset [c={}^0C]] \wedge [\neg P(T) \supset {}^0\mathbf{F}]]],$$

we have:

ad (10). Presupposition $P(T) = \lambda w \lambda t\ [\forall x\ [[{}^0Int\_to_{wt}\ {}^0John\ x\ {}^0Sue] \supset [x={}^0Bill]]]_{wt}$

ad (11). Presupposition $P(T) = \lambda w \lambda t\ [\forall y\ [[{}^0Int\_to_{wt}\ {}^0John\ {}^0Bill\ y] \supset [y={}^0Sue]]]_{wt}$

The construction *C* that is to be executed in case the presupposition is true is here

$$\lambda w \lambda t\ [{}^0Int\_to_{wt}\ {}^0John\ {}^0Bill\ {}^0Sue].$$

Types: *Int_to*/$(o\iota\iota)_{\tau\omega}$ – *who* introduced *who* to *whom*; *John*, *Sue*, *Bill*/ι.

The resulting analyses are

(10*) $\lambda w \lambda t\,^2[\iota\lambda c\ [[\forall x\ [[{}^0Int\_to_{wt}\ {}^0John\ x\ {}^0Sue] \supset [x={}^0Bill]] \supset [c={}^0[{}^0Int\_to_{wt}\ {}^0John\ {}^0Bill\ {}^0Sue]]] \wedge [\exists x\ [[{}^0Int\_to_{wt}\ {}^0John\ x\ {}^0Sue] \wedge \neg[x={}^0Bill]] \supset {}^0\mathbf{F}]]];$

(11*) $\lambda w \lambda t\,^2[\iota\lambda c\ [[\forall y\ [[{}^0Int\_to_{wt}\ {}^0John\ {}^0Bill\ y] \supset [y={}^0Sue]] \supset [c={}^0[{}^0Int\_to_{wt}\ {}^0John\ {}^0Bill\ {}^0Sue]]]] \wedge [\exists y\ [[{}^0Int\_to_{wt}\ {}^0John\ {}^0Bill\ y] \wedge \neg[y={}^0Sue]] \supset {}^0\mathbf{F}]]].$

Using a technical jargon, the truth conditions expressed by the construction (10*) are "If the only person that was introduced by John to Sue is Bill, then it is true that John introduced only *Bill* to Sue, otherwise undefined". Similarly for (11*).

4 Concluding Remarks

We demonstrated the semantic character of topic-focus articulation. This problem is connected with the ambiguity of natural language sentences. Logical analysis cannot disambiguate any sentence, because it presupposes full linguistic competence. Thus the input for our method is the output of a linguistic annotation providing labels for the topic-focus articulation. Yet, our fine-grained method can contribute to a language disambiguation by making these hidden features explicit and logically tractable. In case there are more non-equivalent senses of a sentence we furnish the sentence with different TIL constructions. Having a formal fine-grained encoding of a sense, we can then automatically infer the relevant consequences. Thus in our opinion theoretical linguistics and logic must collaborate and work hand in hand.

Using the expressive logical system of TIL, we were able to provide rigorous analyses such that sentences differing only in the topic-focus articulation are assigned different constructions producing different propositions and implying different consequences. We also analysed the phenomena of presupposition connected with a topic and allegation triggered by a focus so that relevant consequences can be formally derived. Thus, in principle, an inference machine can be built on the basis of TIL analysis such that it neither over-infers (by inferring something that does not follow from the assumptions) nor under-infers (by not being able to infer something that does follow). Currently we develop a computational variant of TIL, the *TIL-Script* functional programming language. TIL constructions are encoded by natural-language expressions in a near-isomorphic manner and for the needs of real-world human agents *TIL-Script* messages are presented in a standardised natural language. *Vice*

versa, humans can formulate their requests, queries, etc., in the standardised natural language that is transformed into *TIL-Script* messages. Thus the provision of services to humans can be realised in a form close to human understanding. From the theoretical point of view, the inference machine for TIL has been specified. However, its full implementation is still an open problem.

The direction of further research is clear. We are going to develop the *TIL-Script* language in its full power, and examine other complex features of natural language. Yet the clarity of this direction does not imply its triviality. The complexity of the work going into building a procedural theory of language is almost certain to guarantee that complications we are currently unaware of will crop up. Yet we are convinced that if any logic can serve to solve such problems, then it must be a logic with hyperintensional (most probably procedural) semantics, such as TIL.

Acknowledgements. This research has been supported by the Grant Agency of the Czech Republic, project "Semantization of pragmatics", No. GACR 401/07/0451, and by the program 'Information Society' of the Czech Academy of Sciences within the project "Logic and Artificial Intelligence for multi-agent systems", No. 1ET101940420.

References

1. Duží, M., Materna, P.: 'Logical form'. In: Sica, G. (ed.) Essays on the Foundations of Mathematics and Logic, vol. 1, pp. 115–153. Polimetrica International Scientific Publisher, Monza (2005)
2. Duží, M.: TIL as the Logic of Communication in a Multi-Agent System. Research in Computing Science 33, 27–40 (2008)
3. Duží, M., Jespersen, B., Müller, J.: Epistemic closure and inferable knowledge. In: Běhounek, L., Bílková, M. (eds.) The Logica Yearbook 2004, pp. 125–140. Filosofia, Czech Academy of Science (2005)
4. Hajičová, E.: What we are talking about and what we are saying about it. In: Gelbukh, A. (ed.) CICLing 2008. LNCS, vol. 4919, pp. 241–262. Springer, Heidelberg (2008)
5. Jespersen, B.: Predication and extensionalization. Journal of Philosophical Logic 37(5), 479–499 (2008)
6. Materna, P.: Conceptual Systems. Logos Verlag, Berlin (2004)
7. Materna, P., Duží, M.: The Parmenides principle. Philosophia, philosophical quarterly Israel 32, 155–180 (2005)
8. Rescher, N.: Epistemic Logic. University of Pittsburgh Press, Pittsburgh (2005)
9. Sandu, G., Hintikka, J.: Aspects of compositionality. Journal of Logic, Language, Information 10, 49–61 (2001)
10. Strawson, P.: Introduction to Logical Theory. Methuen, London (1952)
11. Tichý, P.: The Foundations of Frege's Logic. De Gruyter, Berlin (1988)
12. Tichý, P.: Collected Papers in Logic and Philosophy. In: Svoboda, V., Jespersen, B., Cheyne, C. (eds.). University of Otago Press, Filosofia, Czech Academy of Sciences, Prague (2004)

The Value of Weights in Automatically Generated Text Structures

Dana Dannélls

NLP Research Unit, Department of Swedish Language,
University of Gothenburg, Sweden
dana.dannells@svenska.gu.se

Abstract. One question that arises if we want to evolve generation techniques to accommodate Web ontologies is how to capture and expose the relevant ontology content to the user. This paper presents an attempt to answer the question about how to select the ontology statements that are significant for the user and present those statements in a way that helps the user to learn. Our generation approach combines bottom-up and top-down techniques with enhanced comparison methods to tailor descriptions about a concept described in an ontology. A preliminary evaluation indicates that the process of computing preferable property weights in addition to enhanced generation methods has a positive effect on the text structure and its content. Future work aims to assign grammar rules and lexical entries in order to produce coherent texts that follow on from the generated text structures in several languages.

Keywords: NLG, Ontology, Semantic Web.

1 Introduction

The ability to generate natural language text from web ontology languages and more generally knowledge bases that are encoded in RDF (Resource Description Framework) imposes new demands on natural language generators that aim to produce written text either for textual presentation or for eventual use by text-to-speech system. One of these demands concerns the process of text planning. Text planning, also referred to *Document Planning* [20], is the process responsible for producing a specification of the text's content and structure. The fact that aspects such as the user characteristics, e.g., cognitive state, desires, the background domain knowledge, and linguistic properties must be taken into account and computed simultaneously during planning makes this process computationally hard and so far there has been little success in computing a general model with a suitable structure for generating from ontologies in general and from web ontologies in particular. This brings a need to find alternative strategies to generate knowledge from ontology languages, or alternatively to adapt previously presented ideas to the new emerging technology standards.

Recent attempts to develop natural language generators that support the Web Ontology Language (OWL) and similar Semantic Web languages,[1] treat the

[1] http://www.w3.org/TR/

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 233–244, 2009.

class hierarchy as kind of directed graph that is utilised to produce a coherent text [3, 4] with the most common algorithms including top-down approaches. To enhance personalisation and improve the clarity of the text content describing an object in a hierarchy, these approaches have been combined with comparison methods whose goal is to facilitate learning by relating new concepts to a user's existing knowledge [12, 17]. Yet, one of the main questions that arises in this context is how to capture and expose the relevant ontology content to the reader.

In this paper we present a text planning technique that has been developed to explore the value of assigning weights to ontology properties in addition to comparison methods. The generation technique is optimised to tailor descriptions about a concept from the rich logical structure of Web ontologies and was implemented as a part of a question-answering system. It combines top-down and bottom-up algorithms with enhanced comparison methods to produce a personalised text structure. To test the method performance we run the system on a range of user queries with different user preferences. The generation results indicate that the process of computing preferable property weights in addition to known generation techniques has a positive effect on the text structure and its content. An experiment was conducted to evaluate the generation results using human subjects. The evaluation results show the benefits of manipulating the ontology knowledge on the basis of pre-assigned property weights.

The remainder of this paper is structured as follows. In section 2 we describe the prior approaches in more detail. In section 3 we present the methodology of the generation machinery and the motivation behind the implementation. In section 4 we describe the implementation and the text planning approach. In section 5 we report on the experimental setup and present the evaluation results. In section 6 we discuss their implications and we conclude with section 7.

2 Background

2.1 Semantic Web Ontologies

An *Ontology* is defined as a representation of a shared conceptualisation of a specific domain and plays a vital role in the Semantic Web [2] as it provides a shared and common understanding of a domain that can be communicated between people and heterogeneous, distributed application systems.

Web ontology languages are built upon the RDF and RDF Schema.[2,3] The basic RDF data model contains the concepts of resource in terms of named properties and their values. It is an object-property-value mechanism, which can be seen as forming a graph where each edge represents a *statement* that the resource at the starting end of the edge, called the *Subject* of the statement has a property called the *Predicate* of the statement with a value called the *Object* of the statement. This is shown in Figure 1. Every elliptical node with a label corresponds to a resource and every edge in the graph represents the property of the resource. Formally:

[2] http://www.w3.org/RDF/
[3] http://www.w3.org/TR/rdf-schema/

Definition 1. An ontology O=(G,R) where G is a labeled graph and R is a set of rules. The graph G=(V,E) comprises a finite set of nodes V, and a finite set of edges E. An edge e belonging to the set of edges E is written as (n_1,α, n_2) where n_1 (the subject) and n_2 (the object) are labels of two nodes belonging to a set of nodes V and α is the label (the predicate) of the edge between them.

2.2 Planning the Text Structure from Web Ontologies

The fact that the RDF's abstract syntax can be represented as a directed graph which corresponds to the structure of a coherent text was exploited by various authors who utilise top-down approaches to generate natural languages [4, 18, 22]. As pointed out by these authors, selection methods which follow the ontology graph structure pose several difficulties on the task of planning the text content. One of those is the fact that web ontologies are described as resources and are identified with URIs. This means that they can act as fields not just in the local store but anywhere they appear; when generating natural languages from ontologies it is not always clear where to begin to acquire knowledge about the concept that will be described. Recently, a new approach to content planning has been suggested by [16] who impose a bottom-up method to identify appropriate text contents. They follow an approach that is associated with conversational maxims to select and plan consistent and informative contents [16, 23]. Our approach is most closely in line with [16, 23], however our goals are different. While Mellish and Pan [16] aim to find optimal axioms that are language motivated by inducing new inferences, we aim to improve the text content and structure by combining different generation approaches with preferred property weights. Here we describe an attempt to enhance the input of an NLG system with some domain specific preferences in a way that is adaptive to the task at hand and test whether the generation results actually improve.

2.3 Tailoring the Content and Form of the Text

Bontcheva [3] extends the approach presented in [4] towards portability and personalisation. She presents an approach for producing tailored summaries by accounting for the user preferences that are imposed during the last generation phase, mostly to adapt the length of the generated text. No weights are computed to distinguish what should be included in the text content, and thus there is no adaptation in terms of the contextual information. In M-PIRO [1], it is the user himself who chooses the information that should be included in the generated text and specifies his/her preferred language. This is accomplished through an authoring tool that makes the properties of the object visible to the user. The specified preferences are stored in a user model that is consulted during generation. Similarly to ILEX [18] their user model contains scores indicating the educational value of the chosen information as well as how likely it is for him/her to find a particular type of information interesting. Our approach adds an addition feature to those as it allows to define a set of properties with higher

weights which can be interleaved with the user model and computed during the comparison process.

3 Methodology

3.1 Conveying Semantic Information

To make certain predictions that will help us to convey an ontology content and will allow the system to generate certain continuities in the text structure, there are several questions that are asked, these are: what statements must occur; where can they occur; how often must they occur. Answers to these questions which guide our generation approach depend on the statement's property weight, the ontology content, the user preferences, the context, etc. Let us introduce the following text.

Text T1
U: What is Ghm156?
S: Ghm156 is titled "Vid Rya Strand". Ghm156 was painted by Ekholm Gideon.
U: Who is Ekholm Gideon?
S: Ekholm Gideon is a painter. Ekholm Gideon was born in Sweden.

Text T1 is an example of a successful interaction sequence with the user, the user model in this context was: UM={a18,eN,g2,lS}, following the UM attributes described in section 3.2. The four statements that were generated by the system have received the highest property weights, given the ontology content. A fragment of the ontology from which the ontology statements were generated is shown is Figure 1, in this ontology four domain ontologies are emerged.

We consider this interaction to be a successful one since in the text sequence produced by the system the generated ontology statements that are relevant to the topic of the conversation are presented. Thereby following the Grice's conversational maxims of quantity, i.e., the contribution to the conversation is informative. There is no abundance of information and the generated statements allow the user to ask back on one of the new concepts given in the generated description, e.g., the title, the painter place of birth, etc., from which the system can generate new descriptions relevant to natural language presentation.

To find an adequate sequence of statements about a concept described in the ontology and be able to present the related statements that are relevant in the context, are relevant to the user and eases the user understanding about it, we implemented a stepwise text planning (described in section 4). The planning procedure combines top-down and bottom-up algorithms with comparison techniques to generate relevant content about a concept described in an ontology. In addition it is possible to specify a set of properties with higher weights that can be computed during the comparison process.

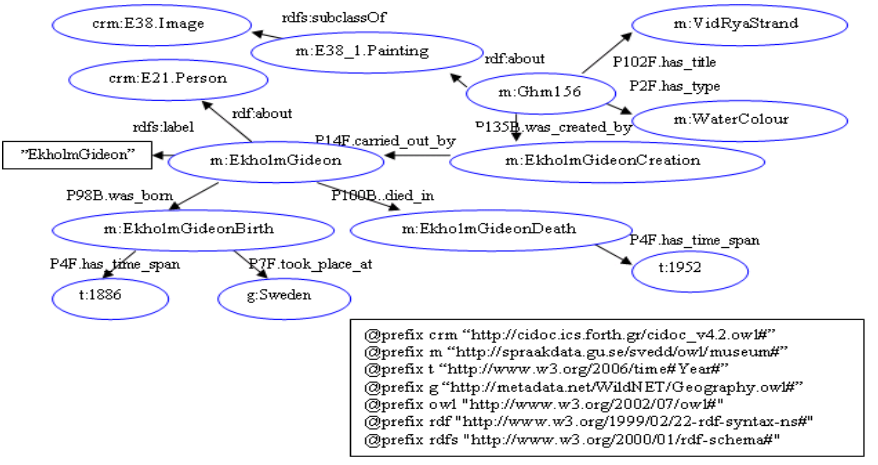


Fig. 1. Classes and properties represented in RDF syntax

3.2 Tailoring the Ontology Content

We aim to establish the rhetorical text content that supports reader and listener preferences. This is accomplished with the help of two modules: (1) the *User Module (UM)*, holds metadata information about the user's: age $a \in \{$ 7-16, $\geq 17\}$; expertise $e \in \{$ expert, non-expert $\}$; generated facts per sentence $g \in \{1, 3, \geq 4\}$ preferred textual complexity $l \in \{simple, complex\}$. (2) the *Memory Module (MM)*, represents the user knowledge, filters out repetitive RDF statements and ranks the selected statements. As the discourse evolves the memory increases; depending on the user module, statements in the memory might receive higher selection priority (section 4.2). This information characterise the user specific part of the input to a single invocation of the generation system.

Similarly to [3, 21], we utilise the names of the ontology concepts and properties to generate the lexicon and produce the text content. Our point of departure is the English language in which the ontology information is given. However, we intend to map each concept and property to its appropriate lexical entry in languages other than English and implement a grammar that makes use of those entries to generate natural language contents.

4 Implementation

4.1 The Generation Machinery

Our approach was implemented within a question-answering system where users can access a knowledge base of information in natural language [13]. The system architecture is introduced in [9]. The initial input data to the process are an ontology file and a user profile file. The user profile holds the user preferences that are stored in the UM. The ontology knowledge is held in a Jena store from

which information is retrieved.[4] The system generates a description about the concept described in the ontology that was chosen by the user. The output is a set of content elements describing the input concept for the given case. It is a subset of verbalised statements describing the input concept.

4.2 Stepwise Text Planning

The text planning module is decomposed into two distinct phases [5], it is a flexible approach that allows to exploit text possibilities [15].[5] (1) *rhetorical representation*, deciding on how to select and organise the data (see below); (2) *document representation* (also called surface realisation) distributing the available data among sentences, paragraphs and perhaps vertical lists in the hope that it will permit a coherent realization as text. Here we take a simple approach to complete the generation process, i.e., concepts are assumed to be lexicalised as nouns and properties as verbs.[6]

The rhetorical representation module acquisition problem is decomposed in two main steps: Content selection and Content organisation.

Content Selection. Content selection operates over a relevant data that has been identified within the generator, see (1a), Table 1. Given the user query, the user model, the memory model, the ontology knowledge-base and a set of scored properties (edges) the task is to select the informative statements that meet the user request and that eases the user understanding about it.

First, all the edges in which the concept n appears in are selected. Second, every concept, i.e.,n_new other than the input one that has a path from n in G is selected. The selected edges are added to a subgraph G', the prim sign ' indicating a subset.

Scoring Equation. Scores are computed for every selected edge according to the equation presented in (1b) Table 1 that was partially inspired by [12].
W_α: the edge property weight;

$Hier_n$: hierarchical distance between the selected concept and the compared resource (i.e., the subject node of the edge in focus);
$Hist_n$: historical distance, i.e., the amount of generated edges after the edge in focus was presented to the user, 0 if it was never presented.

Content Organisation. In this phase we assume there is no useful organisation to the taxonomic information of the selected subgraphs, or alternatively that

[4] http://jena.sourceforge.net/

[5] This process of text planning is equivalent to the two processing modules: *Content Determination* and *Content Planning* that were proposed by [19].

[6] Although there appears to be similarities between lexical entries and concepts, in linguistics and philosophy the term *concept* is defined as a nonlinguistic psychological representation of a class of entities in the ontology, where verbs distinguish what properties it has.

Table 1. Content selection algorithm, following the formal ontology Definition 1, section 2.1.

(1a) Statement selection:
function SELECT(n, G)
Input a node n, and an ontology graph G
For $n \in$ V
Add (V_n,E_n) to G'
For $n\_new \in$ V
Add (V_{n_new},E_{n_new}) to G'
return G'

(1b) Score selected statement:
procedure SCORE(E,p)
Input a set of edges E, and a set of properties p
For $e \in$ E
Score(e)= W_α + $Hier_n$ + $Hist_n$

such organisation as there is, follows the ontology structure. Given a set of scored edges that cover the input query, the task is to look for the relevant ones and organise them accordingly to generate the final output. This step is carried out by a stochastic search method [14, 15]. The stochastic search method is a form of heuristic search that executes the following generic algorithm:

1. Randomly pick one or more edges from the set, in such a way as to prefer items with the highest scores.
2. Use these to generate one or more new random variations.
3. Add these to the set, possibly removing less preferred edges in order to adapt the size to the user requirements.

5 Evaluation

5.1 The Domain Ontology

Our domain ontology follows the CIDOC Conceptual Reference Model (CRM) thesaurus standard.[7] It is a conceptual model that subscribes an object-centred view of the CH domain. The model comprises 81 relations and 244 concepts and covers the semantic field of hundreds of schemata [10].

The domain ontology was created from the Carlotta database,[8] which is designed to be equally applicable to the CIDOC-CRM and covers objects from cultural history, photos, literature, archaeology, theatre, etc. It was enhanced with about 150 new concepts and properties, each of which was assigned with a *rdfs:label* that links its lexical string-name. Figure 1 illustrates a fragment of the data represented as RDF graph (in this graph only one label is made visible).

[7] http://cidoc.ics.forth.gr/
[8] http://carlotta.gotlib.goteborg.se/pls/carlotta/welcome

Table 2. Property list for scoring edges (w1-less valuable, w2-valuable, w3-most valuable).

View	Property name	Weight
Artist name	carried_out_by	w3
Artwork titles	has_title	w3
Artist's birth date	was_born	w2
Artist's death date	died_in	w1
Artist birth place	took_place_at	w2

5.2 Adjusting the Domain Properties

The experiment presented here was carried out within the cultural heritage domain and the heuristics for selecting and ordering properties are based on the results from pilot studies that were conducted in order to gauge user needs and preferences with regards to browsing and exploring material related to artists and artwork [6, 7].

As a result of these studies four main information views that correspond to user needs were identified. We learned that the four views that were found most helpful and interesting for the variety of users correspond to five properties in our domain ontology. The identified views, their corresponding properties and their weights are specified in Table 2.

5.3 Experiment and Result

We run the system on a range of user queries with different user preferences and generated two outputs for each query, one that applies the method described here (Our method), and another that does compute user preferences but does not utilise the scores and the ordering component, i.e., the results follow the ontology structure (Simple method). Below follows one example of the generation results.

Our method

Text T2

(1) U: What is Ghm156?
(2) S: Ghm156 is a painting. Ghm156 has title "Vid Rya Strand". Ghm156 was carried out by Ekholm Gideon.
(3) U: Who is Ekholm Gideon?
(4) S: Ekholm Gideon is a painter. Ekholm Gideon's birth took place in Sweden. Ekholm Gideon birth date has time span 1886.

Simple method

Text T3

(1) U: What is Ghm156?
(2) S: Ghm156 is a painting. Ghm156 has type water colour. Ghm156 has title "Vid Rya Strand".

Table 3. Evaluation results. Quality (Q), Coherency (C), Interest (I)

	Our method	Simple method
Q	3.6	1.8
C	2.3	2.8
I	3.8	1.9

(3) U: What is Ghm156?
(4) S: Ghm156 has time span 1924. Ghm156 was created by Ekholm Gideon's creation. Ghm156 was carried out by Ekholm Gideon.

In text T2 that was produced using our approach, three of the most important properties (according to our property set) are presented already after the first enumeration question, which enables the user to precede with the next question about the new concept, e.g., "Ekholm Gideon". When we employed the simple method approach, text T3, the user needs to repeat on the query about the same concept, e.g., "Ghm156" since the information provided after the first enumeration question does not contribute with informative knowledge. In this case the generated statements are not consist and violet Grice's maxim.

Fourteen interaction sequences, similar to the above examples, were generated and presented to non-experts human subjects, in total eleven subjects participated in the evaluation. Each participant was asked to evaluate the usefulness of each interaction sequence in terms of: (a) Quality (Q), whether the content of the generated statements were relevant and helpful in describing the required object; (b) Coherency (C), whether the generated text structures were coherent and made sense; (c) Interest (I), whether the presented statements (facts) invoked the user interest. For this evaluation a five-point scale (0-poor, 5-excellent) was used. We calculated the mean value of results, these are summarised in Table 3.

A closer look at the generated text structures that were presented in different points of the interaction sequences showed there were cases where the generated content contained a mixture of statements describing different concepts, yet that are all related to the required concept. This may explain why the simple method is superior in "coherency". On the other hand in "quality" and "interest", our method outperforms over the simple approach, which is encouraging.

6 Discussion

Though the idea of exploiting properties of the ontology concepts for generation tasks is not new, the approach here is new in regards to testing in practice how the choice of the property weights effects the text structure and its content with the aim to promote insights into generating knowledge from web ontologies. The fact that content determination is not bounded to the ontology structure makes it possible to gradually present information that accommodates to different contextual degrees and user needs.

The choice of employing a generation approach such as the one presented here that is compatible with employing a domain-specific ontology is based on the relative ease by which such knowledge might provide solutions for building domain-independent generators. Currently it is assumed that a task-specific approach such as the one presented here is tied to the domain ontology and operates at the object level, however, when merged with other ontologies it may operate on meta-level [22].

The approach presented here was only tested on a small ontology with, where only a few subjects participated in the evaluation, a question that comes to mind is how well does it scale [11]. From our observation we anticipate that operating on larger ontologies may give raise to several modifications, for example the selection strategy may result in a large content when retrieving all knowledge about an object, this might be limited by putting an exceeds threshold on the depth and length of the required graph.

The growing body of research that generates from structured databases has directed different methods towards comparisons to enhance comprehension and improve the clarity of texts for the end-user [8, 17, 18]. Comparison methods can reveal the patterns of contrast and similarity [12] and have proven to be useful to remove redundant information, a problem that is exhibited in RDF's [4]. The specific selection strategy adopted here accommodates to these approaches and has proven these methods feasibility for generating from a web ontology.

7 Conclusion and Future Work

In this paper we presented a generation approach to text planning that has been developed to explore the value of assigning weights to domain specific properties. The generation method combines bottom-up and top-down approaches with enhanced comparison techniques to accommodate for the complex structure of Web ontologies. It was implemented within a question-answering framework where the primary goal was to tailor descriptions about a concept described in an ontology to a variety of users. The generated results show the benefits of assigning preferred property weights to enhance the quality and relevance of the generated content elements. A preliminary evaluation indicates that when several factors are enforced during planning, users' interest about the content describing an ontological concept seems to increase.

Although this study focused on a domain specific ontology, and conclusions were drawn based on a small amount of generation results, the findings and technical principles behind the presented methodology could likely to be generalised to other domains. Furthermore, an evaluation can potentially be repeated to confirm the generation results and to test how well does the method scales. Future work aims to assign grammar rules and lexical entries in order to produce coherent texts from the generated text structure elements. In this paper we emphasised mainly the text structure and rhetorical content, but it is necessary to cover linguistic aspects to motivate the chosen text structures for producing grammatically correct texts.

References

1. Androutsopoulos, I., Kokkinaki, V., Dimitromanolaki, A., Calder, J., Oberl, J., Not, E.: Generating multilingual personalized descriptions of museum exhibits: the m-piro project. In: Proceedings of the International Conference on Computer Applications and Quantitative Methods in Archaeology (2001)
2. Berners-lee, T.: Semantic Web Road Map. W3C Design Issues (October 1998), http://www.w3.org/DesignIssues/Semantic.html
3. Bontcheva, K.: Generating tailored textual summaries from ontologies. In: Gómez-Pérez, A., Euzenat, J. (eds.) ESWC 2005. LNCS, vol. 3532, pp. 531–545. Springer, Heidelberg (2005)
4. Bontcheva, K., Wilks, Y.: Automatic report generation from ontologies: The MI-AKT approach. In: Meziane, F., Métais, E. (eds.) NLDB 2004. LNCS, vol. 3136, pp. 324–335. Springer, Heidelberg (2004)
5. Bouayad-Agha, N., Power, R., Scott, D.: Can text structure be incompatible with rhetorical structure? In: Proceedings of First International Natural Language Generation Conference (INLG), pp. 194–200 (2000)
6. Capra, R., Marchionini, G., Oh, J.S., Stutzman, F., Zhang, Y.: Effects of structure and interaction style on distinct search tasks. In: JCDL 2007: Proceedings of the 2007 conference on Digital libraries, pp. 442–451. ACM Press, New York (2007)
7. Clough, P., Marlow, J., Ireson, N.: Enabling semantic access to cultural heritage: A case study of tate online. In: Proceedings of the ECDL. Workshop on Information Access to Cultural Heritage, Aarhus, Denmark (2008) ISBN 978-90-813489-1-1
8. Dale, R., Reiter, E.: Computational interpretations of the gricean maxims in the generation of referring expressions. Cognitive Science 19, 233–263 (1994)
9. Dannélls, D.: A system architecture for conveying historical knowledge to museum visitors. In: Proceedings of the 12th European Conference On Research And Advanced Technology For Digital Libraries (ECDL) Workshop on Information Access to Cultural Heritage, Aarhus, Denmark (2008) ISBN 978-90-813489-1-1
10. Doerr, M., Ore, C.E., Stead, S.: The cidoc conceptual reference model: a new standard for knowledge sharing. In: ER 2007: Tutorials, posters, panels and industrial contributions at the 26th International Conference on Conceptual Modeling (2007)
11. Hardcastle, D., Scott, D.: Can we evaluate the quality of generated text? In: The 6th edition of the Language Resources and Evaluation Conference (LREC 2008), Marrakech, Morocco (2008)
12. Isard, A.: Choosing the best comparison under the circumstances. In: Proceedings of the International Workshop on Personalization Enhanced Access to Cultural Heritage (PATCH 2007) (2007)
13. Johansson, P., Degerstedt, L., Jönsson, A.: Iterative development of an information-providing dialogue system. In: Proceedings of 7th European Research Consortium for Informatics and Mathematics (ERCIM) Workshop (2002)
14. Mellish, C., Knott, A., Oberlander, J.: Experiments using stochastic search for text planning. In: International Conference on Natural Language Generation (1998)
15. Mellish, C., Oberlander, J., Knott, A.: An architecture for opportunistic text generation. In: The Ninth International Workshop on Natural Language Generation (1998)
16. Mellish, C., Pan, J.Z.: Natural language directed inference from ontologies. Artificial Intelligence 172, 1285–1315 (2008)
17. Milosavljevic, M.: Content selection in comparison generation. In: The 6th European Workshop on Natural Language Generation (6th EWNLG) (1997)

18. O'Donnell, M.J., Mellish, C., Oberlander, J., Knott, A.: ILex: An architecture for a dynamic hypertext generation system. Natural Language Engineering 7, 225–250 (2001)
19. Reiter, E.: Has a consensus NLG architecture appeared and is it psycholinguistically plausible? In: Proceedings of the Seventh International Workshop on Natural Language Generation, Kennebunkport, Maine, pp. 163–170 (1994)
20. Reiter, E., Dale, R.: Building Natural Language Generation Systems. MIT Press and The McGraw-Hill Companies,Inc., Singapore (2000)
21. Wilcock, G.: Talking owls: Towards an ontology verbalizer. In: Human Language Technology for the Semantic Web and Web Services, Sanibel Island, Florida, pp. 109–112 (2003)
22. Wilcock, G., Jokinen, K.: Generating responses and explanations from rdf/xml and daml+oil. In: Knowledge and Reasoning in Practical Dialogue Systems IJCAI, Acapulco, pp. 58–63 (2003)
23. Young, M.R.: Using grice's maxim of quantity to select the content of plan descriptions. Artificial Intelligence 115, 215–256 (1999)

AORTE for Recognizing Textual Entailment

Reda Siblini and Leila Kosseim

CLaC laboratory
Department of Computer Science and Software Engineering
1400 de Maisonneuve Blvd. West
Montreal, Quebec, Canada H3G 1M8
r_sibl@cse.concordia.ca, kosseim@cse.concordia.ca

Abstract. In this paper we present the use of the AORTE system in recognizing textual entailment. AORTE allows the automatic acquisition and alignment of ontologies from text. The information resulted from aligning ontologies created from text fragments is used in classifying textual entailment. We further introduce the set of features used in classifying textual entailment. At the TAC RTE4 challenge the system evaluation yielded an accuracy of 68% on the two-way task, and 61% on the three way task using a simple decision tree classifier.

1 Introduction

In this paper we present a novel method of recognizing textual entailment. Textual entailment is defined as *"a relationship between a coherent text T and a language expression, which is considered as a hypothesis, H. We say that T entails H (H is a consequent of T), if the meaning of H, as interpreted in the context of T, can be inferred from the meaning of T"* [1].
For example, the text:

(T): *Jurassic Park is a novel written by Michael Crichton.*
Entails the following hypothesis (among others):
(H1): *Michael Crichton is an author.*
(H2): *Jurassic Park is a book.*
(H3): *Michael Crichton is the author of the book Jurassic Park.*

Recognizing textual entailment is a fundamental task to many applications in natural language processing, such as in *Information Retrieval* where retrieving relevant documents could be seen as finding documents containing the text that entails the information we are looking for, in *Information Extraction* where the extraction of information is based on a set of templates that entail the information that we would like to extract, in *Question Answering* where candidate answers are snippets that entail the question we want to answer, and in *Summarization* where redundancy can be avoided by detecting textual entailment.

The remainder of the paper is organized as follows. Section 2 presents related work in recognizing textual entailment. In Section 3 we give an overview of our

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 245–255, 2009.

approach, focusing on the main features that were used to classify textual entailment. Section 4 presents a performance evaluation of our system at the recent TAC RTE-4 2008 challenge, and finally in Section 5, we present our conclusion.

2 Related Work

Current methods for recognizing textual entailment are based on a measure of similarity between the text T and the hypothesis H. These methods can be categorized into three main approaches: The first and most popular approach is based on a different set of similarity matching techniques that usually differ by the assumption they make for measuring the similarity. For example, some calculate a similarity measure assuming word independence such as the system of [2], others assume some sort of relationship between words such as the use of parse trees as in the system of [3]. In addition to defining a similarity measure, these methods usually rely heavily on the use of machine learning techniques to classify textual entailment.

The second type of approach is more of a traditional one that relies on a logic based semantic representation of the text and a theorem prover to prove the hypothesis, such as the COGEX system of [4].

The last type of approach is a combination of the two first categories, such as the system of [5] that also relies greatly on world knowledge. This system has achieved the best results on the Recognizing Textual Entailment challenge for the last three years.

The method we describe here can be categorized in the last group. It is a novel approach that relies on knowledge representation, use of extracted world knowledge from the web, and machine learning. But what is unique about it is its use of available techniques in acquiring and aligning ontologies, in addition to machine learning in order to classify textual entailment. The next section will explain our approach in more details.

3 The AORTE Approach

Our approach for recognizing textual entailment is based on the automatic acquisition of an ontology from the text T, and another ontology from the hypothesis H, and the alignment of the acquired ontologies. The textual entailment problem is then reduced to a classification one based on the resulted aligned ontology. In this paper we will not present the details of the creation and alignment of the ontologies, but rather will focus on the classification of textual entailment based on the aligned ontologies.

3.1 Ontology Acquisition

Figure 1 shows a diagram of AORTE's system architecture. As shown in the Figure, given a Text and a Hypothesis, the system automatically acquires an ontology from each, namely ontology-T and ontology-H.

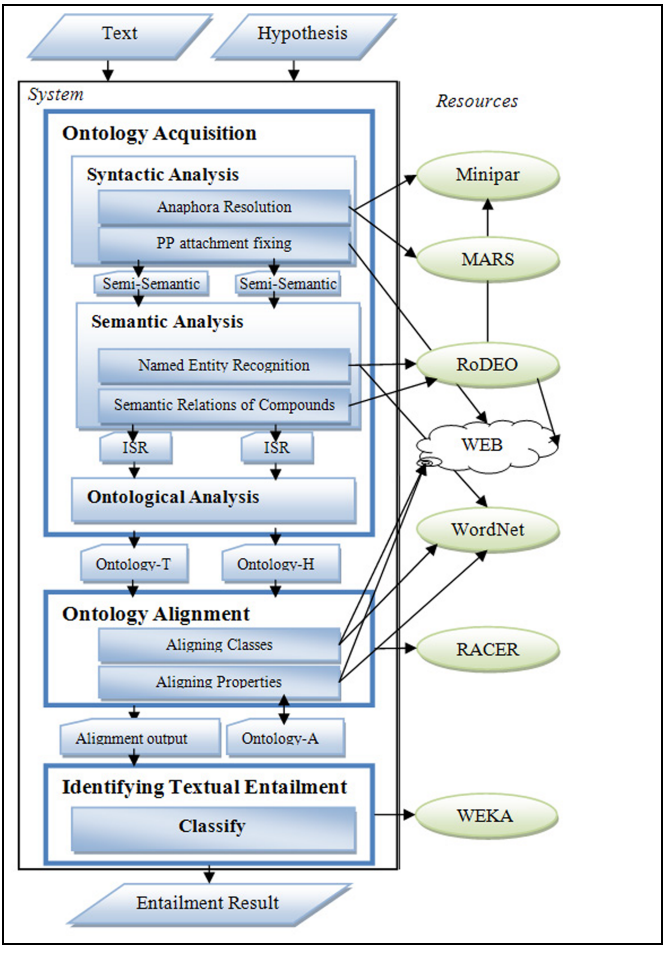


Fig. 1. AORTE System Architecture

The ontology acquisition phase includes different steps that support the acquisition of classes, properties, and instances of an ontology. Briefly, the first step is based on syntactic analysis, which uses the Minipar dependency parser [6], the MARS anaphora resolution system [7], and a set of transformation rules based on part of speech tags to create a structure that is referred to as a semi-semantic structure. The second step to ontology acquisition uses the semi-semantic structure to create a semantic one based on a set of transformation rules and restrictions, in addition to RoDEO's named entities and noun compounds semantic relation extractor [8]. The semantic structure is a semantic underspecified representation of a sentence described by a set of ternary predicate argument relations, characterized by having a predicate as a property that is always relating a governing verb to a content word. For example, the sentence *"Michael Crichton is the author of Jurassic Park"* could be represented as a set of

"(Predicate(Governing Verb, Content Word[Type])" as follows: *"writer(write, Michael Crichton[author]) & writable(write, Jurassic Park[book])"*. RoDEO is responsible for adding relevant world knowledge that is extracted from the web and added to the ontologies, such as the type of content word as in *"book"* for the type of *"Jurassic Park"* in our example, or the verb relating compound nouns as in *"write"* relating *"author"* to *"book"*. The last step in ontology acquisition is the ontological analysis which transforms the created semantic structures into a formal semantic representation expressed in the Ontology Web Language (OWL) and more specifically in OWL-DL, a subset of OWL supporting a decidable (SHOIN(D)) description logic. OWL is a semantic markup language for defining and instantiating web ontologies, and it is based on description logic. It is a vocabulary extension of RDF (the Resource Description Framework), derived from the DAML+OIL (DARPA Agent Markup Language and Ontology Interchange Language), and based on XML (Extensible Markup Language). An OWL ontology may include descriptions of classes, properties (relations between two classes), the classes instances, and a set of axioms. The main reason for selecting OWL is the existence of well studied reasoners that can be used to reason over the created knowledge base schema and instances, the availability of well studied powerful query formalism, the possibility of applying rules to the created knowledge base using backward or forward chaining, and the ability to integrate multiple knowledge bases. The reasoner that we are using is RACER [9]. The RACER system (an acronym for Renamed ABox and Concept Expression Reasoner) is a reasoner that implements tableau calculus for description logic (DL) and supports the web ontology languages DAML+OIL, RDF, and OWL.

In order to illustrate the system's output at each stage, let us take the following example from the Recognizing Textual Entailment (RTE3) challenge development set [10].

(T): *Jurassic Park is a novel written by Michael Crichton.*
(H): *Michael Crichton is the author of the book Jurassic park.*

Figure 2 shows a graphical representation of the automatically created ontology from T (ontology-T); while Figure 3 shows the ontology created from H (ontology-H). In these graphs, rectangles represent classes, solid arrows represent subclasses, ovals represent properties, dotted lines represent property domain, and dotted arrows represent property range. What is particular about these automatically created ontologies is that they are fine-grained in the sense that almost every occurrence of a content word in the text results in the creation of a class, every verb semantic role results in the creation of a property, and additional world knowledge extracted from the web is also added.

As we can see in Figure 2, the created properties are the arguments of the verb *Write*, where it has two properties: WRITER having as range the class *Michael Crichton*, and a WRITABLE having as range the class *Jurassic Park*. In addition the ontology includes a taxonomy created from the extracted types of named entities, and the syntactic structure of the sentence, where *Jurassic Park* is a subclass of *Novel* and *Michael Crichton* is a subclass of *Writer, Director, and*

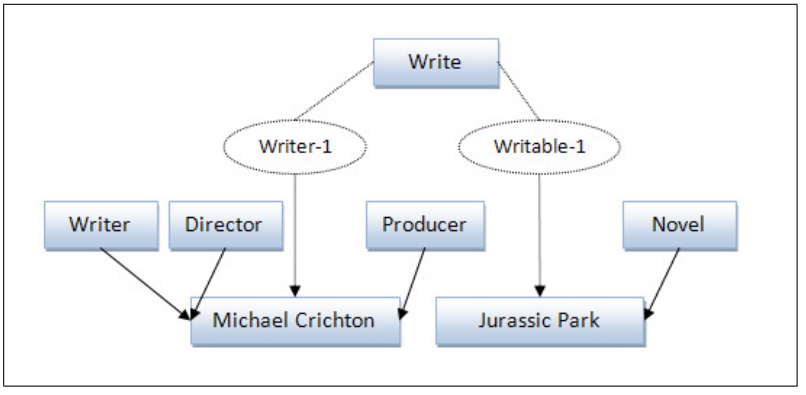


Fig. 2. Example of ontology-T for the sentence *Jurassic Park is a novel written by Michael Crichton*

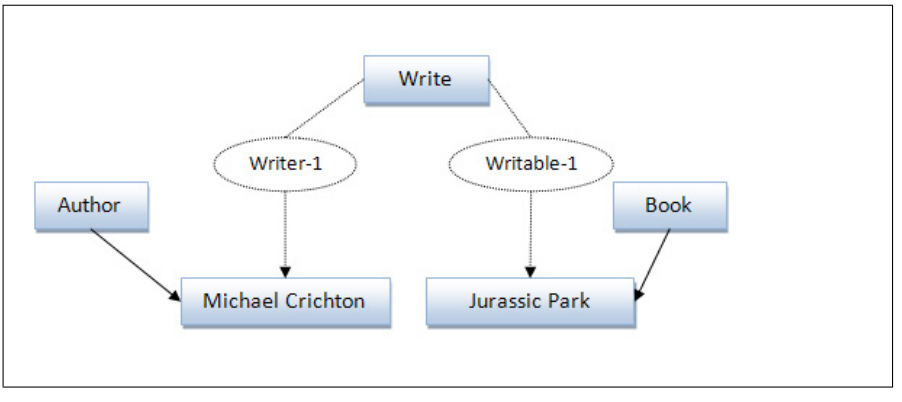


Fig. 3. Example of ontology-H for the sentence *Michael Crichton is the author of the book Jurassic park*

Producer. It should be noted that our definition of an instance as a representation of a snapshot of the world at a certain time, would force us to consider a named entity, such as *Michael Crichton*, as a class containing many instances that each describes a different situation of the class at a different time.

In Figure 3 the class *Write*, which was not actually explicitly mentioned in the hypothesis text, was added in the ontology using the RoDEO system that extracted it from the web as a verb that characterizes the relationship between an *Author* and a *Book*. The arguments of the verb *Write* are then added as properties of this class, were the WRITER property takes as range *Michael Crichton*, and the WRITABLE property takes as range *Jurassic park*.

You can notice from these two graphs that these ontologies are quite similar to each other. Only the two classes: *Author* and *Book*, which are available in the ontology-H, are missing from the ontology-T.

3.2 Ontology Alignment

The next stage of the AORTE system as shown in Figure 1, is the alignment of the created ontologies to form a single ontology, namely ontology-A. The alignment phase aligns the classes and properties of the two created ontologies. The alignment takes as its base the ontology created from T and adds to it the classes and properties that align from the hypothesis ontology. This stage is based on our implementation of the S-match algorithm [11] that uses the verbOcean lexical patterns [12] in addition to WordNet [13] to perform the semantic alignment of classes and properties.

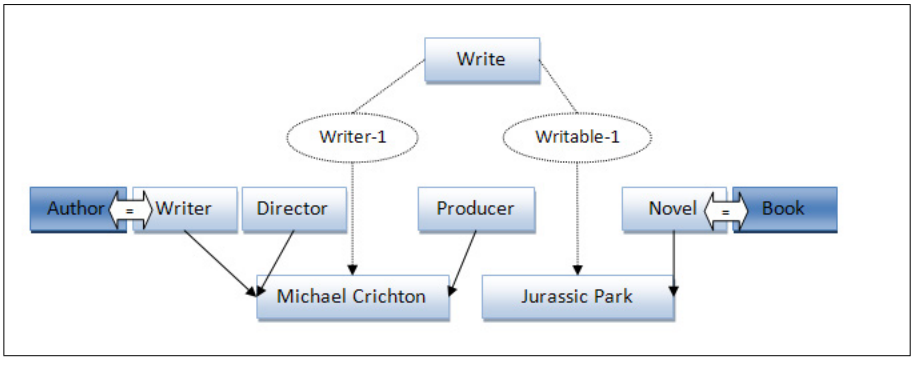


Fig. 4. Example of ontology-A, the alignment of ontology-T of Fig. 2 and ontology-H of Fig. 3

Figure 4 shows a graphical representation of the aligned ontology example, where the arrows represent an equivalence axiom between classes or properties. Note that in this specific example, all the classes and properties have been aligned in the resulted ontology. This is the basis of our hypothesis for recognizing textual entailment, where we take the resulted alignments as features for classifying textual entailment.

3.3 Identifying Textual Entailment

Our hypothesis for recognizing textual entailment is that if a high proportion of classes and properties can be aligned between the two created ontologies, then most probably we have an entailment. Consequently, we created a set of features based on the aligned ontology that we believe may be helpful in classifying textual entailment.

The features are:

F1: Available Classes. This feature represents the percentage of the classes in ontology-H that were available in ontology-T.

F2: Available Properties. This feature represents the percentage of the properties in ontology-H that were available in ontology-T.

F3: Available Sub-Classes. The percentage of subclass relationships between classes in ontology-H that are available in the ontology-A.

F4: Equivalent Classes. The percentage of equivalent classes available in ontology-A from the number of classes in ontology-H; where an equivalent class is a class having a synonym relation discovered using Wordnet.

F5: Possible Equivalent Classes. The percentage of possible equivalent classes available in ontology-A from the number of classes in ontology-H; where a possible equivalent class is an equivalent class that has been labeled by AORTE as being a synonym extracted from the web using the verbOcean lexical patterns.

F6: Equivalent Properties. The percentage of equivalent properties available in ontology-A from the number of properties in ontology-H; where an equivalent property is a property having a synonym relation discovered using Wordnet.

F7: Possible Equivalent Properties. The percentage of possible equivalent properties available in ontology-A from the number of properties in ontology-H; where a possible equivalent properties is an equivalent property that has been labeled by AORTE as being a synonym extracted from the web using the verbOcean lexical patterns.

F8: Disjoint Classes. The percentage of disjoint classes available in ontology-A from the number of classes in ontology-H. Disjoint classes in OWL are two classes that do not have members in common, in our case it means that the two classes representing content words are antonyms, these antonym relations are discovered using Wordnet.

F9: Possible Disjoint Classes. The percentage of possible disjoint classes available in ontology-A from number of classes in ontology-H; where a possible disjoint class is a disjoint class that has been labeled by AORTE as being an antonym extracted from the web using the verbOcean lexical patterns.

F10: Disjoint Properties. The percentage of disjoint properties available in ontology-A from the number of properties in ontology-H. Similar to disjoint classes, that represents content words that are antonyms, and these antonym relations are discovered using Wordnet.

To better illustrate how these features are computed, let us take our examples of ontology-T, ontology-H and ontology-A as shown in Figures 2, 3, and 4. Table 1 shows the necessary parameters needed to compute our feature values. These parameters are retrieved by querying ontology-T, ontology-H, and ontology-A. The querying should not be seen as a simple matching of classes in a repository but as a logical inference that is performed using an inference engine. for example, if we queried the ontology-A for if *Michael Crichton is an author*, the reasoner will return *True* and this will be added to the number of subclass relations in ontology-A parameter. And with these parameters, we can now compute our features:

Table 1. Parameters from ontology-T, ontology-H, and ontology-A used to compute the feature values

Parameter	Value	Parameter	Value
Classes in ontology-T	7	Properties in ontology-T	2
Subclass relationships in ontology-T	4	Classes in ontology-H	5
Properties in ontology-H	2	Subclass relationships in ontology-T	2
Classes in ontology-A	9	Properties in ontology-A	2
Subclass relationships in ontology-A	6	Equivalent classes in ontology-A	2
Equivalent properties in ontology-A	0	Possible equivalent classes in ontology-A	0
Possible equivalent properties in ontology-A	0	Disjoint classes in ontology-A	0
Possible disjoint classes in ontology-A	0	Possible disjoint classes in ontology-A	0
Disjoint properties in ontology-A	0		

Table 2. Sample of annotated Text-Hypothesis pairs from RTE3

Text	Hypothesis	F1	F2	...	F10	Entailment
The sale was made to pay Yukos' US$ 27.5 billion tax bill, Yuganskneftegaz was originally sold for US$ 9.4 billion to a little known company Baikalfinansgroup which was later bought by the Russian state-owned oil company Rosneft.	Baikalfinansgroup was sold to Rosneft.	1	0		0	YES
A decision to allow the exiled Italian royal family to return to Italy may be granted amid the discovery that the head of the family, Prince Vittorio Emmanuele, addressed the president of Italy properly. He has called President Ciampi "our president, the president of all Italians".	Italian royal family returns home.	0.56	0.6		0	NO
Amsterdam police said Wednesday that they have recovered stolen lithographs by the late U.S. pop artist Andy Warhol worth more than $1 million.	Police recovered 81 Andy Warhol lithographs.	0.77	0.2		0	UNKNOWN

$F1 = (7 + 5 - 9)/5 = 0.6$ $F2 = (2 + 2 - 2)/2 = 1$
$F3 = (4 + 2)/6 = 1$ $F4 = 2/5 = 0.4$
$F5 = 0$ $F6 = 0$
$F7 = 0$ $F8 = 0$
$F9 = 0$ $F10 = 0$

We used three classifying algorithms: the B40 decision tree classifier based on ID3, a k-Nearest neighbor classifier with k=1, and a Naïve Bayes classifier. We used as training set the 800 text-hypothesis pairs from the RTE3 pilot task dataset [10]. The RTE3 pilot task is the task of recognizing textual entailment; where the dataset is annotated into three decisions: *yes* for entailment, *no* for no entailment, and *unknown*. We ran the AORTE ontology acquisition and alignment to create ontology-T, ontology-H and ontology-A for all 800 pairs. For each pair, we then extracted the 10 features described above, then trained the classifiers. Table 2 shows a sample from the training set.

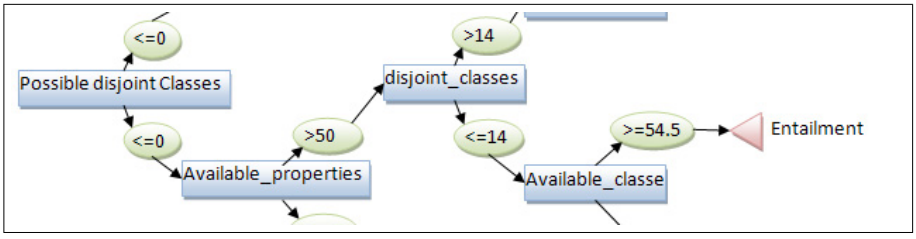


Fig. 5. Part of the decision tree learned from the ontology alignment of RTE3 data set showing the features decision nodes represented by rectangles, each followed by the chance nodes represented by ovals, and ending with a triangle followed by the related decision.

Figure 5) shows part of the decision tree learned from our training set that is relevant to our T-H example. An analysis of the full decision tree indicates that the most discriminating features are the following:

1. F9: Possible disjoint classes (root of the tree).
2. F5, F2, and F3: Possible equivalent classes, available properties, and available sub-classes.
3. F1, F4 and F8: Available classes, equivalent classes and disjoint classes.
4. F6 and F7: Equivalent properties and possible equivalent properties.
5. F10: Disjoint properties.

Traversing the relevant part of the learned decision tree (shown in Figure 5) shows that for our specific example, the decision tree would classify the relation as entailment.

4 Performance Evaluation

To evaluate our system, we participated in the recent TAC-RTE challenge. The Text Analysis Conference (TAC) is a conference that comprises a collection of tracks concerned with the evaluation of different NLP related technologies. One of these tracks is the Recognizing Textual Entailment (RTE) Track, which has the goal of providing a framework to systems that recognize when text entails another. The RTE challenge define textual entailment as *"a directional relationship between two text fragments", where a Text (T) entails a Hypothesis (H) if the truth of H can be inferred from T within the context induced by T"*. The RTE challenge is also divided into two tasks: A three way task in which a system must classify textual entailment into: *"Entailment"* if T entails H, a *"Contradiction"* if T contradicts H and *"Unknown"* if the system cannot determine the truth of H based on T. A two way task in which a system must classify textual entailment into: *"Entailment"* if T entails H, and *"No entailment"* if T does not entail H. In order to evaluate our AORTE system, we used the RTE4 challenge, and classified the given 1000 T-H pair into the three way task (*Entailment, Contradiction,*

and Unknown). The evaluation is done automatically, where the classifications returned by a system are compared to human annotated golden standard, and the returned score is the accuracy or the percentage of matching judgments. As the RTE4 task did not provide a development set, we used the RTE3 pilot dataset introduced in the previous section for training.

We submitted three different runs, each for a different classification algorithm. In the first run we used the B40 decision tree classifier, and compared to human annotated answers, this run resulted in an accuracy of 61.6%. For the second run we used the nearest neighbor classifier (k=1) and resulted in an accuracy of 52%. In the last run we employed a Naïve Bayes classifier that yielded a score of 43.2%. The best run that scored 61.6% was ranked 2nd when compared to other system that participated in the same challenge.

The RTE challenge automatically converts the three way submitted runs of each system into two way runs by automatically conflating *"Contradiction"* and *"Unknown"* to *"No Entailment"*. The B40 decision tree classifier on the two way run scored a 68.8%, the nearest neighbor classifier achieved 54%, and the Naive Bayes classifier marked a 54.7%. The best run of 68.8% was ranked 2nd when compared to other system that participated in the 3-way challenge and had their answers automatically converted to the two-way format.

In addition to the accuracy measure, the challenge provided the possibility of ranking the textual entailment pairs by confidence. Where the more confident the system is in an entailment the higher it is placed or ranked in the evaluated set. This score is labeled an average precision score, where it is computed as the average of the systems precision values at all points in the ranked list in which the gold standard annotation is *"Entailment"*. As average precision is relevant only for a binary annotation, in the case of three-way judgment submissions the pairs tagged as *"Contradiction"* and *"Unknown"* will be conflated and re-tagged as *"No entailment"*.

We ranked the resulted classification in our system simply by highest number of available classes and properties. So the highest is the percentage of available classes + available properties in the aligned ontologies of a T-H pair having an *"Entailment"* result, the higher it is placed on the result set. As such, the system had an average precision of 58.1% for all runs.

5 Conclusion

This paper proposed a textual entailment recognizing approach based on the alignment of ontologies that are automatically extracted from text. The approach classifies textual entailment by learning from the overlap of classes and properties between the text ontologies and the hypothesis ontology, the percentage of equivalent and disjoint classes and properties in the aligned ontology, and other ontology-alignment related features. The system performance was evaluated using the Recognizing Textual Entailment (RTE-4) challenge, resulting in a accuracy of 68% on the two-way task, and 61% on the three way task for the best run that uses a simple decision tree classifier, which ranked 2nd when compared

to the other systems that participated in the challange. By carefully studying our results we have realized that the system performance had significantly been affected by the text length, as a result our future work will focus on resolving this issue and mainly on improving the association of relevant knowledge. In addition, we will work on providing a detailed analyze of the type of textual entailment the system can handle and specifically the contribution of each of the system's component to the overall performance.

References

[1] Dagan, I., Glickman, O.: Probabilistic textual entailment: Generic applied modeling of language variability. Learning Methods for Text Understanding and Mining (2004)
[2] Corley, C., Csomai, A., Mihalcea, R.: A Knowledge-based Approach to Text-to-Text Similarity. In: Recent Advances in Natural Language Processing IV: Selected Papers from RANLP 2005 (2007)
[3] Kouylekov, M., Magnini, B.: Tree Edit Distance for Textual Entailment. In: Recent Advances in Natural Language Processing IV: Selected Papers from RANLP 2005 (2007)
[4] Tatu, M., Iles, B., Moldovan, D.: Automatic Answer Validation Using COGEX. In: Peters, C., Clough, P., Gey, F.C., Karlgren, J., Magnini, B., Oard, D.W., de Rijke, M., Stempfhuber, M. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 494–501. Springer, Heidelberg (2007)
[5] Hickl, A., Bensley, J.: A Discourse Commitment-Based Framework for Recognizing Textual Entailment. In: ACL (2007)
[6] Lin, D.: Dependency-based evaluation of Minipar. In: Workshop on the Evaluation of Parsing Systems, Granada, Spain (1998)
[7] Mitkov, R., Evans, R., Orăsan, C.: A new, fully automatic version of mitkov's knowledge-poor pronoun resolution method. In: Gelbukh, A. (ed.) CICLing 2002. LNCS, vol. 2276, p. 168. Springer, Heidelberg (2002)
[8] Sibini, R., Kosseim, L.: Rodeo: Reasoning over dependencies extracted online. In: Proceedings of the The 4th Web as Corpus: Can we do better than Google?, a workshop of the Sixth International Language Resources and Evaluation (LREC 2008), Marrakech, Morocco (2008)
[9] Haarslev, V., Moller, R.: Racer: A Core Inference Engine for the Semantic Web. In: Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools, Sanibel Island, Fl, USA (2003)
[10] 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)
[11] Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-match: an algorithm and an implementation of semantic matching. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 61–75. Springer, Heidelberg (2004)
[12] Chklovski, T., Pantel, P.: VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. In: Proc. of EMNLP 2004, Barcelona, Spain (2004)
[13] Fellbaum, C.: WordNet: an electronic lexical database. MIT Press, Cambridge (1998)

Semi-supervised Word Sense Disambiguation Using the Web as Corpus

Rafael Guzmán-Cabrera[1,2], Paolo Rosso[2], Manuel Montes-y-Gómez[3], Luis Villaseñor-Pineda[3], and David Pinto-Avendaño[4]

[1] FIMEE, Universidad de Guanajuato, Mexico
guzmanc@salamanca.ugto.mx
[2] NLE Lab, DSIC, Universidad Politécnica de Valencia, Spain
prosso@dsic.upv.es
[3] LabTL, Instituto Nacional de Astrofísica, Óptica y Electrónica, Mexico
{mmontesg,villasen}@inaoep.mx
[4] FCC, Benemérita Universidad Autónoma de Puebla, Mexico
dpinto@cs.buap.mx

Abstract. As any other classification task, Word Sense Disambiguation requires a large number of training examples. These examples, which are easily obtained for most of the tasks, are particularly difficult to obtain for this case. Based on this fact, in this paper we investigate the possibility of using a Web-based approach for determining the correct sense of an ambiguous word based only in its surrounding context. In particular, we propose a semi-supervised method that is specially suited to work with just a few training examples. The method considers the automatic extraction of unlabeled examples from the Web and their iterative integration into the training data set. The experimental results, obtained over a subset of ten nouns from the SemEval lexical sample task, are encouraging. They showed that it is possible to improve the baseline accuracy of classifiers such as Naïve Bayes and SVM using some unlabeled examples extracted from the Web.

1 Introduction

It is well known that, in all languages, some words may have several different meanings or senses. For example, in English, the word "bank" can either mean a financial institution or a sloping raised land. Related to this language phenomenon, the task of *Word Sense Disambiguation* (WSD) considers the assignment of the correct sense to such ambiguous words based on their surrounding context [6].

There are two main kinds of methods to carry out the task of WSD. On the one hand, the knowledge-based methods, which disambiguate words by comparing their context against information from a predefined lexical resource such as Wordnet [1, 3]. On the other hand, *corpus-based methods*, which achieve the sense disambiguation by applying rules that were automatically learned from a sense tagged corpus [14]. Recent reports [8] indicate that corpus-based methods tend to be more precise than knowledge-based ones. Nevertheless, due to the lack of large sense tagged corpora (as well as to the difficulty of manually creating them), the use of these kind of methods is still very limited.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 256–265, 2009.

In order to tackle the above mentioned problem, many researches have recently been working on *semi-supervised learning methods* [2, 4], which consider the usage of large amount of unlabeled data together with a few labeled examples. In particular, the idea of learning classifiers from a combination of labeled and unlabeled data has been successfully applied in WSD [9, 10, 13, 15, 16].

In line with these current works, we have proposed a new semi-supervised method for general text classification tasks [5]. This method differs from previous approaches in two main issues. First, it does not require a predefined set of unlabelled training examples, instead it considers their automatic extraction from the Web. Second, it applies a self-training approach that selects instances not only considering their labelling confidence by a base classifier, but also their correspondence with a web-based labelling[1]. This method has been applied with success in thematic and non-thematic text classification tasks, indicating that it is possible to automatically extract discriminative information from the Web.

In this paper, we move forward to investigate the application of the proposed *web-based self-training method* in the task of WSD. This task confronts our method with new challenges since (i) ambiguous words tend to have several "slightly" different meanings, and (ii) their classification typically rely only on a very small context. This way, the task of WSD can be considered as a narrow-domain and short-text classification problem.

The rest of the paper is organized as follows. Section 2 describes our web-based self-training approach. Section 3 presents the evaluation results of the method in a subset of ten words from the last SemEval English lexical sample exercise. Finally, Section 4 depicts our conclusions.

2 Our Semi-supervised Classification Method

Figure 1 shows the general scheme of our semi-supervised text classification method. It consists of two main processes. The first one deals with the corpora acquisition

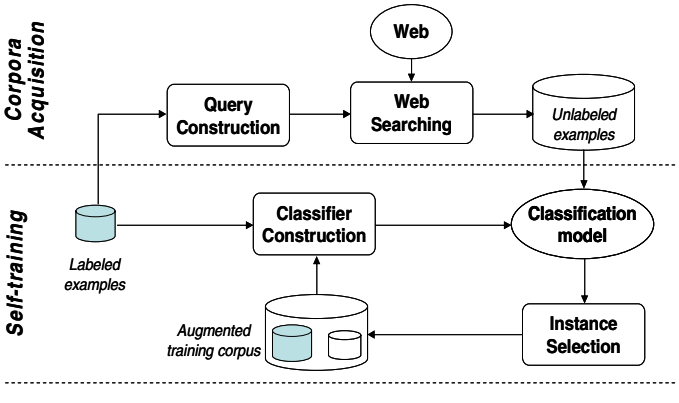


Fig. 1. General overview of our text classification method

[1] Given that each unlabeled example is downloaded from the Web using a set of automatically defined class queries, each of them has a default category or web-based label.

from the Web, whereas the second focuses on the self-training learning approach. The following sections describe in detail these two processes.

It is important to notice that this method can be directly applied to the task of WSD since it is, in essence, a text classification problem, where word senses correspond to classes and word contexts represent the documents.

2.1 Corpora Acquisition

This process considers the automatic extraction of unlabeled examples from the Web. In order to do this, it first constructs a number of *queries* by combining the most significant words for each sense of a polysemous word; then, using these queries, it looks at the Web for some additional training examples related to the given senses.

At this point, it is important to comment that even though the idea of using the Web as corpus, it may not initially sound intuitive; there are already a number of successful efforts concerning different natural language tasks [7]. In particular, in [17], the authors proposed a method for mining the Web to improve text classification by creating a background text set. Our method is similar to this approach in the sense that it also mines the Web for additional information (extra-unlabeled examples). Nevertheless, as we will describe below, our method applies finer procedures to construct the set of queries related to each sense and to combine the downloaded information.

Query Construction

To construct the set of queries for searching the Web, it is necessary to previously determine the set of relevant words from each sense in the training corpus. The criterion used for this purpose is based on a combination of two characteristics of the given words: on the one hand, their frequency of occurrence, and on the other hand, their information gain. Explicitly, we consider that a word w_i is relevant for a sense S if:

1. The frequency of occurrence of w_i in S is greater than the average occurrence of all words (happening more than once) in that sense. That is:

$$f_{w_i}^S > \frac{1}{|S'|} \sum_{\forall w \in S'} f_w^S \text{, where } S' = \left\{ w \in S \middle| f_w^S > 1 \right\}$$

2. The information gain of w_i in the given training set is positive ($IG_{w_i} > 0$). The idea of this condition is to select those words that help reducing the uncertainty of the value of the sense from the given set of examples.

Having obtained the set of relevant words per each sense it is possible to construct their corresponding set of queries. We decided to construct queries of three words[2]. This way, we created as many queries per sense as all three-word combinations of its relevant words. We measure the significance of a query $q = \{w_1, w_2, w_3\}$ to the sense S as indicated below:

[2] Queries formed by more than three words tend to produce very few results; on the other hand, queries of one or two words are very general and, consequently, tend to retrieve a lot of irrelevant results.

$$\Gamma_S(q) = \sum_{i=1}^{3} f_{w_i}^{S} \times IG_{w_i}$$

Because the selection of relevant words relies on a criterion based on their frequency of occurrence and their information gain, the number of queries per sense is not the same even though they include the same number of training examples. In addition, an increment in the number of examples does not necessarily represent a growth in the number of built queries.

Web Searching

The next action is using the defined queries to extract from the Web a set of additional unlabeled text examples from the Web. Based on the observation that most significant queries tend to retrieve the most relevant Web pages, our method for searching the Web determines the number of downloaded examples per query in a direct proportion to its Γ-value. Therefore, given a set of *M* queries $\{q_1,\ldots, q_M\}$ for sense *S,* and considering that we want to download a total of *N* additional examples per sense, the number of examples to be extracted by a query q_i is determined as follows:

$$\Psi_S(q_i) = \frac{N}{\sum_{k=1}^{M} \Gamma_S(q_k)} \times \Gamma_S(q_i)$$

It is important to notice that, because each downloaded example corresponds exactly to one particular query; it is possible to consider that these examples belong to a particular sense (the same sense of the query that was used to retrieved them). This information, which we previously mentioned as Web-based labeling, represents a kind of prior category for the unlabeled examples, and thus it can be of great help in improving the performance of the semi-supervised learning approach.

2.2 Semi-supervised Learning

The objective of this second process is to increase the classification accuracy by gradually enlarging the originally small training set with the unlabeled examples downloaded from the Web. In particular, we designed this process based on the *self-training approach* described in [12]. In this approach, a classifier is initially trained using the small amount of labeled data; then, this classifier is used to classify the unlabeled data, and the most confident examples -in conjunction with their predicted label- are added to the training set; finally, the classifier is re-trained and the procedure is repeated.

In our case, as we previously explained, the selection of the most confident examples not only considers their labeling confidence by a base classifier, but also their correspondence with the Web-based labeling. Following, we detail our new self-training algorithm:

1. Build a weak classifier (C_l) using a specified learning method (l) and the training set available (T).
2. Classify the unlabeled Web examples (E) using the constructed classifier (C_l). In other words, estimate the sense for all downloaded examples.

3. Select the best m examples per sense ($E_m \subseteq E$; in this case E_m represent the union of the best m examples from all senses) based on the following two conditions:
   a) The estimated sense of the example corresponds to the sense of the query used to download it. In some way, this filter works as an ensemble of two classifiers: C_l and the Web (expressed by the set of queries).
   b) The example has one of the m-highest confidence predictions for the given sense.
4. Combine the selected examples with the original training set ($T \leftarrow T \cup E_m$) in order to form a new training collection. At the same time, eliminate these examples from the set of downloaded instances ($E \leftarrow E - E_m$).
5. Iterate σ times over steps 1 to 4 or repeat until $E_m = \varnothing$. In this case σ is a user specified threshold.
6. Construct the final classifier using the enriched training set.

3 Experimental Evaluation

3.1 Evaluation Data Set

The evaluation of the method was carried out on a subset of the lexical sample task from the SemEval forum[3]. In particular, we consider only *nine nouns* which have training instances for all their senses. Table 1 shows some numbers about these nouns. It is interesting to notice that there is an important imbalance problem for some nouns indicated by the standard deviation value. For instance, for the first sense of "bill" there are 685 training instances, whereas for the second there are only 54, producing an average standard deviation of 446.18.

3.2 Evaluation Measure and Baseline Results

The effectiveness of the method was measured by the *classification accuracy*, which indicates the percentage of instances of a polysemous word that were correctly classified from the entire test set.

Table 1. Data set statistics

Noun	Number of senses	Training instances	Test instances	Standard deviation (*of training instances per sense*)
Source	5	151	35	20.64
Bill	2	739	114	446.18
President	3	872	176	401.24
Management	2	277	44	40.30
Condition	2	130	33	59.40
Policy	2	329	39	129.4
Rate	2	1003	145	490.02
Drug	2	205	46	28.99
State	3	609	70	263.03

[3] http://nlp.cs.swarthmore.edu/semeval/tasks/task17/description.shtml

Table 2. Baseline results using Naïve Bayes and SVM

Noun	Classification accuracy	
	Naïve Bayes	SVM
Source	77.14	74.29
Bill	92.08	95.05
President	89.20	89.20
State	78.57	78.57
Management	77.27	85.82
Condition	66.66	72.72
Policy	74.36	87.18
Rate	86.90	87.59
Drug	78.26	71.74

Table 2 shows the baseline results for two different classifiers, namely, Naïve Bayes and SVM. In all cases, we determined the context of the words using a window of five words to the left and five words to the right. In all cases, we also removed all punctuation marks and numerical symbols, as well as all stopwords.

As it can be seen, there is a relationship between the number of training instances and their degree of imbalance (refer to Table 1) with the baseline accuracy (refer to Table 2). Therefore, this result evidences the need for increasing the size of the training sets by incorporating new unlabeled examples.

3.3 Results of the Method

This section describes the application of the proposed semi-supervised method to the task of WSD. The method, as depicted in Section 2, includes two main processes: the corpora acquisition from the Web and the self-training learning approach. Following, we detail some data from both of them.

The central task for corpora acquisition is the automatic construction of a set of queries that expresses the relevant content of each sense. For this experiment we considered the ten words with the greatest weight. Then, using these queries, we collected from the Web a set of 1,000 additional examples per sense for each polysemous word. Table 3 shows some example queries corresponding to the two different senses of the word "drug".

Regarding the learning phase, it is important to point out that there is not a clear criterion to determine the parameters m and σ of our self-training method. For this experiment, we determined the number of unlabeled examples that must be incorporated into the training set at each iteration based on the following condition: the added

Table 3. Example queries for the two senses of the word "drug"

Sense	Queries
Drug-1	drug new used drug said company drug sales companies
Drug-2	drug trafficking charges drug charges major drug major use

information –expressed in number of words– must be proportionally small with respect to the original training data. This last condition is very important because of the small size of word contexts. In particular, we decided to incorporate five examples per sense at each iteration. However, it is necessary to perform further experiments in order to determine the best value of *m* for this task.

Table 4 shows the results of this experiment. They indicate that our method slightly outperformed all baseline results especially when using the Naïve Bayes classifier. These results confirm our intuition that in scenarios having very few training instances it is better to include a small group of unlabeled examples that considerably augments the dissimilarities among senses than to include a lot of doubtable-quality information.

Table 4. Results of our method for the first three iterations (using Bayes and SVM)

	Bayes				SVM			
Noun	**Baseline Result**	**It. 1**	**It. 2**	**It. 3**	**Baseline Result**	**It. 1**	**It. 2**	**It. 3**
Source	77.1	80.0	80.0	80.0	74.3	77.1	80.0	68.6
Bill	92.0	92.0	92.1	91.1	95.1	95.1	95.1	93.1
President	89.2	87.5	88.1	88.1	89.2	89.8	89.8	87.5
State	78.5	80.0	78.6	80.0	78.6	78.6	78.6	78.6
Managment	77.2	79.5	79.5	79.5	85.8	81.8	81.8	81.8
Condition	66.6	66.6	66.7	63.6	72.7	72.7	75.8	75.8
Policy	74.3	76.9	76.9	74.4	87.2	87.2	87.2	74.8
Rate	86.9	86.9	89.0	89.0	87.6	87.6	86.9	74.4
Drug	78.2	80.4	80.4	-	71.7	71.7	69.6	86.9

3.4 Discussion of Results

In order to have a deep understanding of achieved results, we carried out a statistical analysis of the used corpus. The purpose of this analysis was to explain the complementary performance of the Naïve Bayes and SVM classifiers. It is necessary to remark that Naïve Bayes is a probabilistic classifier that apply the Bayes theorem under the assumption (naïvely) that exist independence on the features of the items to be classified. From this viewpoint, we suggest to use a statistical measure that takes into account the relationship among the words that made up each text (the features used in this experiment). In particular, we applied a measure called SLMB[4] (supervised language modeling based measure) [11]. This measure uses a set of language models (based on bigrams and trigrams) to compute the entropy among the different meanings of each ambiguous word. Formally, given a corpus *D* (of one ambiguous word), with a gold standard consisting of *k* classes (or meanings) $C = \{C_1, C_2,\ldots,C_k\}$, the SLMB measure is defined as follows:

$$SMLB(D) = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(Perplexity(C_i \mid \overline{C_i^*}) - \mu(Perplexity(C)))^2}$$

[4] http://nlp.dsic.upv.es:8080/watermaker

$$\mu(Perplexity(C)) = \frac{\sum_{i=1}^{k} Perplexity(C_i \mid \overline{C_i^*})}{k}$$

In these formulas, $\overline{C_i^*}$ indicates the language model obtained by using all the classes except C_i, and $Perplexity(C_i \mid \overline{C_i^*})$ denotes the perplexity of the class C_i language model with respect to the $\overline{C_i^*}$ language model. The latter formula calculates the mean of the perplexity among the different ambiguous word meanings.

Table 5 shows the results obtained by the SLMB measure and the perplexity mean for all word corpora. In all cases, we evaluate the original and the enriched corpus. On the one hand, we may observe that words *state*, *management*, *policy* and *rate* have not changed significantly their language model from the original to the enriched version of the corpus. Therefore, there were not significant changes over the dependency relationships among the words (features), which leads to obtain a similar behavior of the Naïve Bayes classifier with both corpora (original and enriched). On the other hand, we may see that the ambiguous words *president*, *source*, *condition* and *drug* have obtained important changes on the values obtained with the SLMB and perplexity mean, which means that their language models have been modified sufficiently avoiding to preserve the same or similar results with both corpora (original and enriched). However, the SVM classifier may have been benefited from this last fact. We consider that their support vectors have been enriched, which could helped the SVM classifier to have obtained better results than the Naïve Bayes did on these last ambiguous words.

Table 5. SLMB and perplexity mean results over both, the original and enriched corpus

Noun	SLMB			Perplexity Mean		
	Original value	Final value	Change (%)	Original value	Final value	Change (%)
Bill	23.8	30.5	28.1	140.2	166.5	18.7
State	24.4	29.0	19.0	106.3	143.5	34.9
Management	5.7	5.1	10.5	114.3	122.4	7.1
Policy	41.4	38.2	7.7	116.1	135.9	17.1
Rate	21.6	22.8	5.5	124.7	136.8	9.7
President	61.2	164.9	169.4	150.8	264.4	75.3
Source	52.3	81.7	56.2	68.8	145.4	111.1
Condition	25.8	31.5	22.4	76.8	111.5	45.1
Drug	5.8	0.4	93.4	81.9	87.4	6.8

4 Conclusions

This paper describes a novel web-based self-training method for text classification. This method differs from other semi-supervised classification approaches in that: (i) it is specially suited to work with very few training examples, and (ii) it considers the automatic extraction of additional training knowledge from the Web.

The described method was already evaluated on two different classification tasks (classification of news reports and contemporary poem authorship attribution, respectively), obtaining good results in both cases. In this paper, we went a step forward and investigated the possibility of applying this method in the task of WSD, which can be considered a narrow-domain and short-text classification problem.

The results obtained in a subset of ten nouns from the SemEval lexical sample task were not as successful as those achieved in previous tasks. Nevertheless, they evidence that unlabeled data may improve performance of potentially any corpus-based WSD system.

Acknowledgments

This work was done under partial support of CONACYT-Mexico, PROMEP-Mexico (UGTO-121), and MCyT-Spain (TIN2006-15265-C06-04).

References

1. Aguirre, E., Rigau, G.: A Proposal for Word Sense Disambiguation using Conceptual Distance. In: Proc. of the Int. Conf. on Recent Advances in NLP. RANLP 1995 (1995)
2. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proc. COLT, pp. 92–100 (1998)
3. Buscaldi, D., Rosso, P.: A conceptual density-based approach for the disambiguation of toponyms. International Journal of Geographical Information Science 22(3), 143–153 (2008)
4. Goldman, S., Zhou, Y.: Enhancing supervised learning with unlabeled data. In: Proc. ICML, pp. 327–334 (2000)
5. Guzmán-Cabrera, R., Montes-y-Gómez, M., Rosso, P., Villaseñor-Pineda, L.: Using the Web as Corpus for Self-training Text Categorization. Journal of Information Retrieval (forthcoming, 2009) ISSN 1386-4564
6. Ide, N., Veronis, J.: Introduction to the special Issue on word sense disambiguation: the state of the art, Computational Linguistics. Special Issue on word sense Disambiguation 24(1), 1–40 (1998)
7. Kilgarriff, A., Greffenstette, G.: Introduction to the Special Issue on Web as Corpus. Computational Linguistics 29(3), 1–15 (2003)
8. Lee, Y.K., Ng, H.T.: An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In: Proc. EMNLP, pp. 41–48 (2002)
9. Mihalcea, R.: Co-training and Self-training for Word Sense Disambiguation. In: Proc. CoNLL, pp. 33–40 (2004)
10. Pham, T.P., Ng, H.T., Lee, W.S.: Word Sense Disambiguation with Semi-Supervised Learning. In: Proc. AAAI, pp. 1093–1098 (2005)
11. Pinto, D.: On Clustering and Evaluation of Narrow Domain Short-Text Corpora. PhD thesis, Universidad Politécnica de Valencia, Spain (2008)
12. Solorio, T.: Using unlabeled data to improve classifier accuracy. M.Sc. thesis, Computer Science Department, INAOE, Mexico (2002)
13. Su, W., Carpuat, M., Wu, D.: Semi-Supervised Training of a Kernel PCA-Based Model for Word Sense Disambiguation. In: Proc. COLING, pp. 1298–1304 (2004)

14. Tratz, S., Sanfilippo, A., Gregory, M., Chappell, A., Posse, C., Paul, W.: PNNL: A Supervised Maximum Entropy Approach to Word Sense Disambiguation. In: Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval. 2007), pp. 264–267 (2007)
15. Yarowsky, D.: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In: Proc. ACL, pp. 189–196 (1995)
16. Yu, N.Z., Hong, J.D., Lim, T.C.: Word Sense Disambiguation Using Label Propagation Based Semi-supervised Learning Method. In: Proc. ACL, pp. 395–402 (2005)
17. Zelikovitz, S., Kogan, M.: Using Web Searches on Important Words to Create Background Sets for LSI Classification. In: 19th Int. FLAIRS Conf., Melbourne Beach, Florida (2006)

Semi-supervised Clustering for Word Instances and Its Effect on Word Sense Disambiguation

Kazunari Sugiyama and Manabu Okumura

Precision and Intelligence Laboratory, Tokyo Institute of Technology,
4259 Nagatsuta, Midori, Yokohama, Kanagawa 226-8503, Japan
sugiyama@lr.pi.titech.ac.jp, oku@pi.titech.ac.jp

Abstract. We propose a supervised word sense disambiguation (WSD) system that uses features obtained from clustering results of word instances. Our approach is novel in that we employ semi-supervised clustering that controls the fluctuation of the centroid of a cluster, and we select seed instances by considering the frequency distribution of word senses and exclude outliers when we introduce "must-link" constraints between seed instances. In addition, we improve the supervised WSD accuracy by using features computed from word instances in clusters generated by the semi-supervised clustering. Experimental results show that these features are effective in improving WSD accuracy.

1 Introduction

Many words have multiple meanings depending on the context in which they are used. For example, among the possible senses of the verb "run" are "to move fast by using one's feet" and "to direct or control." Word sense disambiguation (WSD) is the task of determining the meaning of such an ambiguous word in its context. In this paper, we apply semi-supervised clustering by introducing sense-tagged instances (we refer to them as "seed instances" in the following) to the supervised WSD process. Our approach is based on the following intuitions: (1) in the case of word instances, we can use sense-tagged word instances from various sources as supervised instances, and (2) the features computed from word instances in clusters generated by our semi-supervised clustering are effective in supervised WSD since word instances clustered around sense-tagged instances may have the same sense. Existing semi-supervised clustering approaches solely focus on introducing constraints and learning distances and overlook control of the fluctuation of the cluster's centroid. In addition, to enable highly accurate semi-supervised clustering, it is important to consider how to select seed instances and how to introduce constraints between the seed instances. Regarding seed instances, we have to pay attention to the frequency distribution of word senses when selecting seed instances as well as the way of introducing "must-link" constraints, since outlier instances may exist when we select seed instances with the same sense.

In this paper, we describe our semi-supervised clustering approach that controls the fluctuation of the centroid of a cluster and propose a way of introducing appropriate seed instances and constraints. In addition, we explain our WSD approach using features computed from word instances that belong to clusters generated by the semi-supervised

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 266–279, 2009.

clustering. Our approach is novel in that we employ semi-supervised clustering that controls the fluctuation of the centroid of a cluster, and we select seed instances by considering the frequency distribution of word senses and exclude outliers when we introduce "must-link" constraints between seed instances.

2 Related Work

2.1 Semi-supervised Clustering

The semi-supervised clustering methods can be classified into *constraint-based* and *distance-based*. Constraint-based methods rely on user-provided labels or constraints to guide the algorithm toward a more appropriate data partitioning. For example, Wagstaff et al. [12,13] introduced two types of constraint – "must-link" (two instances have to be together in the same cluster) and "cannot-link" (two instances have to be in different clusters) – and their semi-supervised K-means algorithm generates data partitions by ensuring that none of the user-specified constraints are violated. Basu et al. [18] also developed a semi-supervised K-means algorithm that makes use of labeled data to generate initial seed clusters and to guide the clustering process. In distance-based approaches, an existing clustering algorithm that uses a particular clustering measure is employed; however, it is trained to satisfy the labels or constraints in the supervised data [5,9,1].

2.2 Word Sense Disambiguation

In order to improve WSD accuracy, several works add features to original features such as POS tags, local collocations, bag-of-words, syntactic relations. For example, Agirre et al. [7] proposed the idea of "topic signatures." They first submit synonyms, gloss, hypernyms, hyponyms, meronums, holonyms and attributes in WordNet as well as the target word as a query to a search engine and then compute χ^2 values (topic signatures) using the extracted words from the searched documents. Finally, they apply these topic signatures to WSD. Specia et al. [19] presented a WSD system employing inductive logic programming [16] that can represent substantial knowledge to overcome the problem of relying on a limited knowledge representation and generate a disambiguation model by applying machine learning algorithms to attribute-value vectors. Cai et al. [3] constructed topic features on an unlabeled corpus by using the latent dirichlet allocation (LDA) algorithm [6], then used the resulting topic model to tag the bag-of-words in the labeled corpus with topic distributions. Finally, to create the supervised WSD system, they constructed a classifier applying features such as POS tags, local collocations, bag-of-words, syntactic relations as well as topic models to a support vector machine.

Generally, in the case of using context as features for WSD, the feature space tends to be sparse. Niu et al. [23] proposed a semi-supervised feature clustering algorithm to conduct dimensionality reduction for WSD with maintaining its accuracy.

Other recent WSD studies include nominal relationship classification where pattern clusters are used as the source of machine learning features to learn a model [4], and WSD system using OntoNotes project [8] that has a coarse-grained sense inventory [24].

3 Proposed Method

The existing semi-supervised clustering approaches solely focus on introducing constraints and learning distances. However, when we apply semi-supervised clustering to word instances, we have to pay attention to introduce "must-link" constraints since word instances might be distant from each other in the feature space even if they have the same sense. In addition, semi-supervised clustering method used in [23] is based on label propagation algorithm. Unlike this method, our proposed semi-supervised clustering approach is constraint-based with controlling the fluctuation of the centroid of a cluster. We could verify that this approach is effective in personal name disambiguation in Web search results [20]. Therefore, we refine this semi-supervised clustering approach suitable for word instances. Moreover, the recent supervised WSD systems described in Section 2.2 do not use information obtained from word instances clustered to seed instances although they add a lot of features to improve WSD accuracy. We believe that the accuracy of WSD can be improved by directly computing features from word instances clustered to seed instances. In this section, we give an overview of our system, describe our semi-supervised clustering approach for word instances, and explain how to compute features obtained from the clustering results.

3.1 System Architecture

Figure 1 illustrates our WSD system. This system extracts features for clustering and WSD, and performs semi-supervised clustering by introducing seed instances, as described in Section 3.2. After that, it computes features for WSD from the word instances in the generated clusters (Section 3.3). Using these features, a classifier can be constructed on the basis of three machine learning approaches: support vector machine (SVM), naïve Bayes (NB), and maximum entropy (ME).

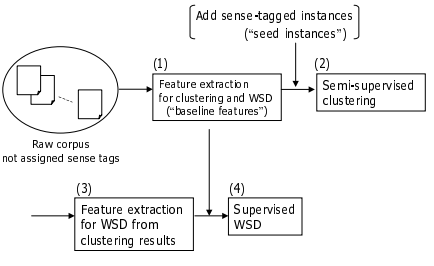


Fig. 1. Proposed WSD system

3.2 Semi-supervised Clustering

3.2.1 Features for Clustering

We use the following features:

- Morphological features
  - Bag-of-words (BOW), Part-of-speech (POS), and detailed POS classification. We extract these features from the target word itself and the two words to its right and left.

– Syntactic features
  - If the POS of a target word is a noun, extract the verb in a grammatical dependency relation with the noun.
  - If the POS of a target word is a verb, extract the noun in a grammatical dependency relation with the verb.
– Figures in Bunrui-Goi-Hyou (BGH)[1] [21]
  - 4 and 5 digits regarding the content word to the right and left of the target word. For example, when the target word is "syakai" ("society") and its left content word is "chiiki" ("community"), the figures of "chiiki" in thesaurus is "1.1720,4,1,3." We use 1172 and 11720 as 4 and 5 digits, respectively.
– 5 topics inferred on the basis of LDA [6]
  - We compute the log-likelihood of an instance using the "soft-tag"approach [3] where the topics are estimated from training data set (Fig.4) by regarding this set as unlabeled set using LDA.

We chose ChaSen[2] as the morphological analyzer and CaboCha[3] as the syntactic parser. We denote the feature vector $\boldsymbol{f}^x$ of word instance x as follows:

$$\boldsymbol{f}^x = (f_1^x, f_2^x, \cdots, f_n^x).$$

We refer to these features as "baseline features." We also use them in our WSD system (Section 3.3).

3.2.2 Semi-supervised Clustering

If the similarity between cluster C_{s_j} that contains seed instances and cluster C_i that does not contain seed instances is large, these two clusters are to be merged. However, when the distance between the centroids of these two clusters is large, the fluctuation of the centroid tends to be large. Therefore, when we merge a certain cluster C_i (its centroid vector $\boldsymbol{G}^{C_i}$) into C_{s_j} (its centroid vector $\boldsymbol{G}^{C_{s_j}}$) that contains seed instances, we first weight the feature vector $\boldsymbol{f}^x \in C_i$ relative to the distance between the centroids of the clusters. After that, we control the fluctuation of the centroid of a cluster by recomputing it with the weighted feature vectors. The details of the procedure are as follows:

We assume a cluster $C_{s_j}^{(k_j)}$ (number of elements: n_{s_j}) in which k_j clusters are merged and that contains a seed instance. We also assume that C_i (number of elements: n_i) is a cluster merged $(k_j + 1)$ times into $C_{s_j}^{(k_j)}$. We define the elements of cluster $C_{s_j}^{(0)}$ as the initial seed instances.

(1) Regarding each element contained in C_i that is merged into $C_{s_j}^{(k_j)}$, by using the distance $D(\boldsymbol{G}^{C_i}, \boldsymbol{G}^{C_{s_j}^{(k_j)}})$ between the centroid $\boldsymbol{G}^{C_{s_j}^{(k_j)}}$ of cluster $C_{s_j}^{(k_j)}$ and the centroid $\boldsymbol{G}^{C_i}$ of cluster C_i, we weight the feature vector $\boldsymbol{f}_{C_i}^{x_l}$ $(l = 1, \cdots, n_i)$ of word instances belonging to cluster C_i and define the generated cluster as $C_{i'}$ (number of elements: $n_{i'}$). The feature vector $\boldsymbol{f}_{C_{i'}}^{x_l}$ after weighting that belongs to cluster $C_{i'}$ is

[1] BGH is "Word List by Semantic Principles." In BGH, each word has a number called *a category number.*
[2] http://sourceforge.net/projects/masayu-a/
[3] http://sourceforge.net/projects/cabocha/

$$\boldsymbol{f}_{C_{i'}}^{x_l} = \frac{\boldsymbol{f}_{C_i}^{x_l}}{D(\boldsymbol{G}^{C_i}, \boldsymbol{G}^{C_{s_j}^{(k_j)}}) + c}, \tag{1}$$

where c is a constant to prevent the elements of $\boldsymbol{f}_{C_i}^{x_l}$ from being extremely large when $D(\boldsymbol{G}^{C_i}, \boldsymbol{G}^{C_{s_j}^{(k_j)}})$ is very close to 0. This value of c is set to 0.92 based on our preliminary experiments. We introduce adaptive Mahalanobis distance $D(\boldsymbol{G}^{C_i}, \boldsymbol{G}^{C_{s_j}^{(k_j)}})$, to overcome the drawback of the ordinary Mahalanobis distance whereby the covariance tends to be large when the number of elements in a cluster is small.

(2) We add the elements of $C_{i'}$ (number of elements: $n_{i'}$) to cluster $C_{s_j}^{(k_j)}$ (number of elements: n_{s_j}) that contain a seed instance and generate cluster $C_{s_j}^{(k_j+1)}$ (number of elements: $n_{s_j} + n_{i'}$) as follows:

$$C_{s_j}^{(k_j+1)} = \{\boldsymbol{f}_{C_{s_j}^{(k_j)}}^{x_1}, \cdots, \boldsymbol{f}_{C_{s_j}^{(k_j)}}^{x_{n_{s_j}}}, \boldsymbol{f}_{C_{i'}}^{x_1}, \cdots, \boldsymbol{f}_{C_{i'}}^{x_{n_{i'}}}\},$$

(3) The centroid $\boldsymbol{G}^{C_{s_j}^{(k_j+1)}}$ of cluster $C_{s_j}^{(k_j+1)}$ that merged with the $(k_j+1)^{th}$ cluster is defined as

$$\boldsymbol{G}^{C_{s_j}^{(k_j+1)}} = \frac{\sum_{\boldsymbol{f}^x \in C_{s_j}^{(k_j+1)}} \boldsymbol{f}^x}{n_{s_j} + n_{i'} \times \frac{1}{D(\boldsymbol{G}^{C_i}, \boldsymbol{G}^{C_{s_j}^{(k_j)}}) + c}}. \tag{2}$$

We weight the feature vector of the cluster to be merged in Equation (1), thus we also weight $n_{i'}$ as we can compute weighted average in Equation (2). If the cluster does not contain seed instances, the new centroid $\boldsymbol{G}^{new}$ of the cluster is computed using the following equation:

$$\boldsymbol{G}^{new} = \frac{\sum_{\boldsymbol{f}^x \in C_i} \boldsymbol{f}^x + \sum_{\boldsymbol{f}^x \in C_j} \boldsymbol{f}^x}{n_i + n_j}. \tag{3}$$

Figure 2 shows the semi-supervised clustering approach. Constraints between seed instances are also introduced at the beginning of the clustering in order to get accurate clustering results.

3.2.3 Seed Instances and Constraints for Clustering

To obtain higher clustering accuracy in semi-supervised clustering, it is important to introduce the initial seed instances and constraints between the seed instances properly. In this section, we describe how to introduce them in the semi-supervised clustering for word instances. We refer to a set of word instances for selecting seed instances as a "training data set." Generally, when we deal with word instances, it is important to consider the frequency of word senses in the training data set because there are some words whose instances are occupied by the small number of word senses, or other words whose instances are occupied by the large number of word senses. Thus, we consider this characteristic when we introduce seed instances for semi-supervised clustering. The number of training instances in our experiment was set to 100. The constraints between seed instances are "cannot-link" only, "must-link" only and both constraints. However, regarding "must-link" constraints, we have to exclude outlier instances. That is, if we select seed instances that contain outliers, the centroid of the initial cluster is not so accurate; inappropriate clusters tend to be generated in the subsequent clustering. If we

Algorithm: Semi-supervised clustering
Input: Set of feature vectors of word instances $\boldsymbol{f}^{x_i}$ $(i = 1, 2, \cdots, n)$ and seed instances $\boldsymbol{f}^{x_{s_j}}$ $(j = 1, 2, \cdots, u)$,
$E = \{\boldsymbol{f}^{x_1}, \boldsymbol{f}^{x_2}, \cdots, \boldsymbol{f}^{x_n}, \boldsymbol{f}^{x_{s_1}}, \boldsymbol{f}^{x_{s_2}}, \cdots, \boldsymbol{f}^{x_{s_u}}\}$.
Output: Set of clusters $\mathcal{C} = \{C_1, C_2, \cdots\}$ that contain the word instances that have the same sense.
Method:
1. Set feature vectors of each word instance $\boldsymbol{f}^{x_i}$ and each feature of seed instances $\boldsymbol{f}^{x_{s_j}}$ in E
as the initial cluster C_i and $C_{s_j}^{(k_j)}$, respectively.
$C_i = \{\boldsymbol{f}^{x_i}\}, C_{s_j}^{(k_j)} = \{\boldsymbol{f}^{x_{s_j}}\}$,
thus, the set of clusters $\mathcal{C} = \{C_1, C_2, \cdots, C_n, C_{s_1}^{(k_1)}, \cdots, C_{s_u}^{(k_u)}\}$,
where constraints are introduced between $C_{s_m}^{(k_m)}$ and $C_{s_n}^{(k_n)} (m \neq n)$.
k_h $(h = 1, \cdots, u) \leftarrow 0$,
where k_h denotes the frequency of merging other clusters into $C_{s_h}^{(k_h)}$.
2. **do**
2.1 Compute the similarity between C_i and C_j $(i \neq j)$, C_i and between $C_{s_h}^{(k_h)}$.
if the maximum similarity is obtained between C_i and $C_{s_h}^{(k_h)}$,
then compute the distance $D(\boldsymbol{G}^{C_i}, \boldsymbol{G}^{C_{s_h}^{(k_h)}})$
between the centroids $\boldsymbol{G}^{C_i}$ and $\boldsymbol{G}^{C_{s_h}^{(k_h)}}$ of C_i and $C_{s_h}^{(k_h)}$, respectively.
for $l = 1$ **to** n_{C_i} **do**
transform the feature vector $\boldsymbol{f}_{C_i}^{x_l}$ in C_i into $\boldsymbol{f}_{C_{i'}}^{x_l}$, by using Equation (1),
add $\boldsymbol{f}_{C_{i'}}^{x_l}$ to $C_{s_h}^{(k_h)}$
end
$k_h \leftarrow k_h + 1$
recompute the centroid $\boldsymbol{G}^{C_{s_h}^{(k_h)}}$ using Equation (2), and remove C_i from $\mathcal{C}$.
else if the maximum similarity is obtained between C_i and C_j,
then merge C_i and C_j to form a new cluster C^{new}, add C^{new} to $\mathcal{C}$, remove C_i and C_j from $\mathcal{C}$,
and recompute the centroid $\boldsymbol{G}^{new}$ of the cluster C^{new} by using Equation (3).
2.2 Compute similarities between C^{new} and all $C_i \in \mathcal{C}$ $(C_i \neq C^{new})$.
3. **until** All of the similarities computed in 2.2 between C_i and C_j are less than the predefined threshold.
4. **return** Set of clusters $\mathcal{C}$.

Fig. 2. Proposed semi-supervised clustering algorithm

exclude outliers, the centroid of the initial cluster becomes more accurate. We believe this idea leads to better clustering results. Figure 3 shows the algorithm and how to exclude outlier instances. We compute the new centroid generated by two clusters, then compute the distance between the new centroid and the clusters. If the distance is less than a predefined threshold, a "must-link" constraint is put between the two clusters. We compare the following two methods for selecting seed instances in semi-supervised clustering:

[Method I] Select seed instances for semi-supervised clustering from the whole training data set.

[Method II] First classify the training data set into each word sense. Then, considering the frequency of word sense, select seed instances for semi-supervised clustering from each classified word sense.

Figure 4 shows how to select seed instances from the data set of word instances.

Method I

We compared the following three ways of selecting seed instances for semi-supervised clustering:

Algorithm: Adding the "must-link" constraint that excludes outliers
Input: Set each feature vector of seed instance $\boldsymbol{f}^{x_{s_j}}$ $(j = 1, 2, \cdots, u)$,
$S = \{\boldsymbol{f}^{x_{s_1}}, \boldsymbol{f}^{x_{s_2}}, \cdots, \boldsymbol{f}^{x_{s_u}}\}$.
Output: Set of seed instances connected by "must-link" constraints
Method:
1. Set each feature of seed instance $\boldsymbol{f}^{x_{s_j}}$ as an initial cluster C_{s_j},
$C_{s_j} = \{\boldsymbol{f}^{x_{s_j}}\}$,
thus, the set of clusters $\mathcal{C} = \{C_{s_1}, \cdots, C_{s_u}\}$.
2. **do**
2.1 Find the cluster C_{s_w} that has the same sense as C_{s_v} $(v \neq w)$.
if C_{s_w} and C_{s_v} have different senses,
introduce a "cannot-link" constraint between them.
2.2 Compute the new centroid $\boldsymbol{G}^{new}$ based on clusters C_{s_v} and C_{s_w}.
2.3 Compute the distance $D(\boldsymbol{G}^{new}, \boldsymbol{G}^{C_{s_v}})$ between $\boldsymbol{G}^{new}$ and $\boldsymbol{G}^{C_{s_v}}$,
and the distance $D(\boldsymbol{G}^{new}, \boldsymbol{G}^{C_{s_w}})$ between $\boldsymbol{G}^{new}$ and $\boldsymbol{G}^{C_{s_w}}$.
2.4 **if** $D(\boldsymbol{G}^{new}, \boldsymbol{G}^{C_{s_v}}) < Th_{dis}$, and $D(\boldsymbol{G}^{new}, \boldsymbol{G}^{C_{s_w}}) < Th_{dis}$,
a "must-link" constraint is introduced between C_{s_v} and C_{s_w},
then merge them to form a new cluster C^{new}, add C^{new} to $\mathcal{C}$,
and remove C_{s_v} and C_{s_w} from $\mathcal{C}$.
else $D(\boldsymbol{G}^{new}, \boldsymbol{G}^{C_{s_v}}) > Th_{dis}$, $D(\boldsymbol{G}^{new}, \boldsymbol{G}^{C_{s_w}}) > Th_{dis}$
remove C_{s_w} from $\mathcal{C}$.
3. **until** $v = u$
4. **return** Initial seed clusters $\mathcal{C}$ with constraints.
(Outliers are excluded in clusters connected by "must-link" constraints.)

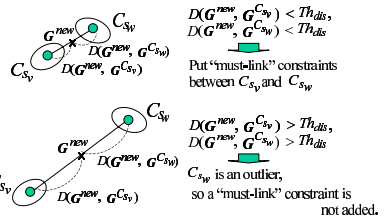


Fig. 3. Algorithm of excluding outlier word instances to add "must-link" constraint (left) and its overview (right)

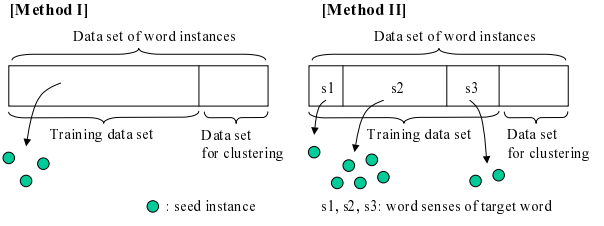


Fig. 4. How to select seed instances from the training data set

(I-1) Select initial seed instances randomly.
(I-2) Select initial seed instances on the basis of "KKZ" [10].
(I-3) As seed instances, select the centroid of a cluster generated by the K-means algorithm [14] whose initial instances are randomly selected (I-3rnd) or selected on the basis of KKZ (I-3KKZ).

"KKZ" in (I-2) is a cluster initialization method that select instances distant from each other [10]. In (I-1) and (I-2), we conduct experiments by introducing constraints of "cannot-link" only, "must-link" only, both constraints, and "cannot-link" and "must-link" without outliers. In (I-3), we introduce "cannot-link" constraints by simply assuming that the selected instances have different senses.

Method II

We compared the following cases: **(II-1)** select the seed instances by considering the frequency of word senses; and **(II-2)** select the seed instances in proportion to the frequency of word senses.

(II-1) We compared the following ways of selecting seed instances for semi-supervised clustering:

(II-1-1) Randomly select initial seed instances in order of word sense frequency,
(II-1-2) Select initial seed instances based on "KKZ" [10] in order of word sense frequency,
(II-1-3) First perform K-means clustering. Then set the centroids of the generated clusters as seed instances in order of word sense frequency for semi-supervised clustering. In this case, the initial instances for K-means clustering are either randomly selected (II-1-3rnd) or selected on the basis of KKZ (II-1-3KKZ).

As in Method I, in our experiment, we add constraints of "cannot-link" only, "must-link" only, both constraints, and "cannot-link" and "must-link" without outliers in (II-1-1) and (II-1-2), but only "cannot-link" constraints in (II-1-3).

(II-2) We use the D'Hondt method [17], a method for allocating seats to candidates in a proportional representation party list. The example in Figure 5 (left) assumes that parties A, B, and C gain votes of 1600, 700, and 300, respectively. When we allocate 10 seats to these parties, the seats are allocated in order of the value in parentheses. These figures are obtained by dividing votes by seat number. Parties A, B, and C gain 5, 4, and 1 seat, respectively. "Seat" and "party" correspond to "number of seed instances" and "word sense," respectively. Similarly, let us assume that word senses s1, s2, and s3, have 20, 50, and 15 instances, respectively (Fig. 5 (right)). When we select 10 instances, the seed instances are selected in order of the value in parentheses. We select 3, 5, and 2 seed instances from s1, s2, and s3, respectively.

As in (II-1), we compared the following three ways of selecting seed instances for semi-supervised clustering.

(II-2-1) Randomly select seed instances for semi-supervised clustering from each of the word senses selected using the D'Hondt method,
(II-2-2) Select seed instances for semi-supervised clustering on the basis of "KKZ" [10] from each of the word senses selected using the D'Hondt method,
(II-2-3) First select the initial instances randomly or by using KKZ for K-means clustering from each of the word senses selected using the D'Hondt method. Then set the centroids of the generated clusters as the seed instances for semi-supervised clustering. The initial instances for K-means clustering are either randomly selected (II-2-3rnd) or on the basis of KKZ (II-2-3KKZ).

In our experiment, we add constraints of "cannot-link" only, "must-link" only, both constraints, and "cannot-link" and "must-link" without outliers, but only "cannot-link" constraints in (II-2-3).

	Party A (1600)	Party B (700)	Party C (300)
seat 1 (/1)	1600 (1)	700 (3)	300 (8)
seat 2 (/2)	800 (2)	350 (6)	150
seat 3 (/3)	533 (4)	233 (9)	
seat 4 (/4)	400 (5)	175 (10)	
seat 5 (/5)	320 (7)		

D'Hondt method

	s2 (50)	s1 (20)	s3 (15)
seed 1 (/1)	50 (1)	20 (3)	15 (5)
seed 2 (/2)	25 (2)	10 (8)	8 (9)
seed 3 (/3)	17 (4)	7 (10)	3
seed 4 (/4)	13 (6)		
seed 5 (/5)	10 (7)		

* s1, s2, s3: word senses
Selecting word senses using D'Hondt method

Fig. 5. D'Hondt method and its application to our system

3.3 Word Sense Disambiguation

3.3.1 Features Obtained Using Clustering Results

We add features obtained from the clustering results to the "baseline features" described in Section 3.2.1 for WSD. Word instances in the generated clusters are aggregated on the basis of their similarity to the seed instances. Therefore, we expect that we can obtain features such as context information from the generated clusters. In particular, we compute features for WSD from the generated clusters. We believe that these features will contribute to the accuracy of WSD. We extracted features from:

(a) inter-cluster information,
(b) context information regarding adjacent words $w_i w_{i+1}$, $(i = -2, \cdots, 1)$, and
(c) context information regarding two words to the right and left of the target word, $w_{-2} w_{-1} w_0 w_{+1} w_{+2}$.

Features (b) and (c) are often used to extract collocations. We use them as features that reflect the concept of "one sense per collocation" [22].

Regarding (a), we employ the term frequency (TF) in a cluster, cluster ID (CID), and the sense frequency (SF) of seed instances. If the values of TF to the right and left of the target word are large, its word sense can be easily identified. Moreover, each generated cluster aggregates similar word instances. Thus, if we use the CID as features for WSD, we can obtain an effect equivalent to assigning the correct word sense. Furthermore, our semi-supervised clustering uses seed instances with sense tags. Therefore, if we use SF as a feature, the word sense of the target word can be easily determined. TF and SF are normalized by the total number of terms and seed instances in each cluster, respectively.

Table 1. two-by-two contingency table showing the dependence of occurrences of w_i and w_{i+1}.

	w_i	$\neg w_{i+1}$
w_{i+1}	O_{11}	O_{12}
$\neg w_{i+1}$	O_{21}	O_{22}

Regarding (b), we compute mutual information (MI), T-score (T), and χ^2 ($CHI2$) for adjacent words. MI is defined as

$$MI = \log \frac{p(w_i, w_{i+1})}{p(w_i)p(w_{i+1})},$$

where $p(w_i)$ and $p(w_i, w_{i+1})$ are the probability of occurence of w_i and the probability of the co-occurrence of w_i and w_{i+1}. T-score is defined as

$$T = \frac{p(w_i, w_{i+1}) - p(w_i)p(w_{i+1})}{\sqrt{s^2/N}},$$

where s^2 and N are sample variance and sample size, respectively. Based on the 2-by-2 contingency table (Table 1), $CHI2$ is defined as

$$CHI2 = \frac{N(O_{11}O_{22} - O_{12}O_{21})^2}{(O_{11} + O_{12})(O_{11} + O_{21})(O_{12} + O_{22})(O_{21} + O_{22})}.$$

In (c), we employ information gain regarding two words to the right and left of the target word. We first compute the entropy by using the set D of feature vectors of word instance as

$$entropy(D) = -\sum_{j=1}^{|s_j|} P_r(s_j) log_2 P_r(s_j), \tag{4}$$

where s_j, $|s_j|$ and $P_r(s_j)$ are word sense, the number of word senses, and its probability of occurrence, respectively. Then, using Equation (5), we compute the entropy of w_i after the clusters are generated:

$$entropy_{w_i}(D) = \sum_{j=1}^{|\nu|} \frac{|s_j|}{|D|} entropy(D), \tag{5}$$

where $|\nu|$ is the number of generated clusters. Using Equations (4) and (5), the information gain $IG(w_0)$ for target word w_0 is defined as

$$IG(w_0) = entropy(D) - entropy_{w_0}(D).$$

Finally, by considering the context for two words to the right and left of the target word w_0, the information gain for w_0 is computed as follows:

$$IG(w_0) = \sum_{i=-2}^{2} IG(w_i).$$

These features for seed instances are also computed in order to verify WSD accuracy.

4 Experiments

4.1 Experimental Data

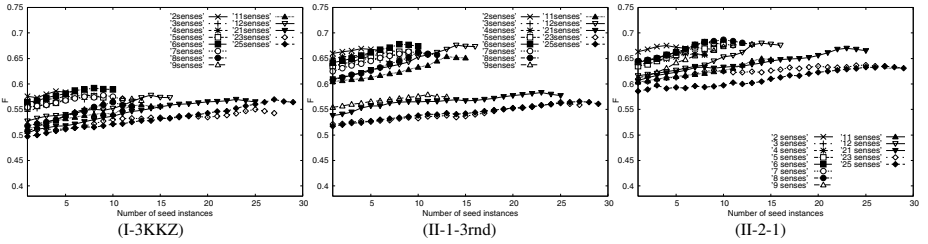
We used the RWC corpus from the "SENSEVAL-2 Japanese Dictionary Task" [11]. In this corpus, sense tags were manually assigned to 3,000 Japanese newspaper (Mainichi Shimbun) articles issued in 1994. The sense tags were assigned to 148,558 ambiguous words that had headwords in a Japanese dictionary (Iwanami Kokugo Jiten) [15] and whose POS was either noun, verb, or adjective. We used the same 100 target words (50 nouns and 50 verbs) as in the SENSEVAL-2 Japanese Dictionary Task.

4.2 Semi-supervised Clustering

In this experiment, we first introduce seed instances and constraints as described in Section 3.2.3. Seed instances are selected from the training data set that corresponds to 80% of the data set of word instances, and test data set for clustering corresponds to 20% of the data set of word instances. The clustering results shown in Section 4.2.2 are based on 5-fold cross validation.

4.2.1 Evaluation Measure

We evaluated the accuracy of our semi-supervised clustering based on F, i.e., the harmonic mean of "*purity*" and "*inverse purity*" [2].



(I-3KKZ) (II-1-3rnd) (II-2-1)

Fig. 6. Clustering accuracy obtained from (I-3KKZ), (II-1-3rnd), and (II-2-1)

Table 2. Comparison of clustering accuracies (F)

	Method I	Method (II-1)	Method (II-2)
Proposed method	**0.543** (I-3KKZ)	**0.592** (II-1-3rnd)	**0.646** (II-2-1)
Bar-Hillel et al. [1]	0.516	0.570	0.608
Xing et al. [9]	0.494	0.539	0.591
Klein et al. [5]	0.448	0.504	0.570
Fixed centroid	0.385	0.402	0.514
Agglomerative clustering	0.380	0.389	0.471

4.2.2 Experimental Results

Because of space limitations, we only show the best clustering results for Methods I, II-1, and II-2. We attempted to add constraints of (a) "cannot-link" only, (b) "must-link" only, (c) both constraints, and (d) "cannot-link" and "must-link" without outliers. Figure 6 shows clustering results when we add constraint (d) because we found that the best clustering accuracy is obtained by using this constraint. These results are obtained using I-3KKZ in Method I, II-1-3rnd in Method II-1, and II-2-1 in Method II-2. In these graphs, each line shows the clustering accuracy obtained for words of each number of word senses. Some of the number of word senses are absent (e.g., 10, 13-20, 22, 24) because such ambiguous words do not exist. The number of seed instances was from one to four plus the original number of word senses defined in the dictionary [15]. Table 2 summarizes the clustering accuracy F obtained by our semi-supervised clustering, distance-based semi-supervised clustering reviewed in Section 2.1, clustering in the case that the centroid of a cluster is fixed, and ordinary agglomerative clustering. These F values are average results for the case of two seed instances in addition to the original word senses, since this number gave the best clustering accuracy.

4.2.3 Discussion

Regarding Method I, the best clustering accuracy is obtained for the centroid of a cluster generated by the K-means algorithm whose initial instances were selected on the basis of KKZ as the seed instances for semi-supervised clustering. When the seed instances are selected from the whole set of training data set, the representative instances tend to be selected by K-means clustering after selecting distant initial instances on the basis of KKZ. Regarding Method II, II-2, which selects seed instances in proportion to the frequency of word senses, is more effective than II-1 which selects seed instances by considering the frequency of word senses. In particular, we found that randomly selecting seed instances is more effective than selecting them by KKZ for seed instances

in the same word senses. In addition, we could obtain the best clustering accuracy in II-2-1 among all of our experiments. From the results, we found that it is effective to take into account the frequency distribution in selecting seed instances.

As described in Section 4.2.2, in most cases, we found that the best clustering accuracy is obtained when two more seed instances are added to the original number of word senses. Although these seed instances are sense-tagged ones, we consider that, in the clustering process, such extra seed instances contribute to discovering new word senses that are not defined in a dictionary by applying semi-supervised clustering to word instances.

According to the results in Table 2, our semi-supervised clustering outperforms other distance-based approaches. We believe that it is better because it locally adjusts the centroid of a cluster whereas the other distance-based semi-supervised clustering approaches transform the feature space globally.

4.3 Word Sense Disambiguation

In order to verify WSD accuracy, we also compute the features described in Section 3.3.1 for sense-tagged training data.

4.3.1 Evaluation Measure

We employ "accuracy" as an evaluation measure for WSD. This measure is based on "fine-grained scoring" that judges the right answer when the word sense that the system outputs completely corresponds to a predefined correct word sense.

4.3.2 Experimental Results

We constructed classifiers using the features described in Section 3.2.1 and 3.3.1 and conducted experiments using five-fold cross validation. Table 3 shows the experimental results for our WSD system (OURS) and for features employed by the participants (CRL, TITECH, NAIST) of the SENSEVAL-2 Japanese Dictionary task. "OURS" means using the baseline features described in Section 3.2.1.

4.3.3 Discussion

For each machine learning approach (SVM, NB, and ME), our WSD had the best accuracy when we added features from clustering results, especially CID, MI and IG, to the

Table 3. WSD accuracies

Features	SVM	NB	ME	Features	SVM	NB	ME
OURS (not clustered)	0.663	0.667	0.662	CRL (not clustered)	0.775	0.778	0.773
OURS + MI (not clustered)	0.666	0.669	0.664	CRL + MI (not clustered)	0.776	0.780	0.775
OURS + CID + MI + IG	**0.780**	**0.782**	**0.779**	CRL + CID + MI + IG	**0.778**	**0.783**	**0.780**
OURS + CID + T + IG	0.768	0.777	0.764	CRL + CID + T + IG	0.778	0.779	0.777
OURS + CID + CHI2 + IG	0.762	0.765	0.757	CRL + CID + CHI2 + IG	0.776	0.779	0.775
TITECH (not clustered)	0.661	0.663	0.660	NAIST (not clustered)	0.745	0.747	0.743
TITECH + MI (not clustered)	0.663	0.665	0.662	NAIST + MI (not clustered)	0.747	0.748	0.745
TITECH + CID + MI + IG	**0.767**	**0.770**	**0.764**	NAIST + CID + MI + IG	**0.765**	**0.767**	**0.764**
TITECH + CID + T + IG	0.765	0.767	0.759	NAIST + CID + T + IG	0.756	0.760	0.755
TITECH + CID + CHI2 + IG	0.756	0.759	0.751	NAIST + CID + CHI2 + IG	0.752	0.754	0.747

baseline features. Among the features (a) (see Section 3.3.1), we found that CID contributed to improvement in WSD accuracy compared with TF and SF. Moreover, among the features (b) (see Section 3.3.1), MI was more effective, and T and $CHI2$ were not so effective. This shows that word instances that have similar contexts can be aggregated into seed instances in the generated clusters. Although our method, TITECH, and NAIST use simple features such as the BOW of the target word, POS, and so on, WSD accuracy was significantly improved by adding features computed from clustering results. For these systems, we obtained 0.020 to 0.117 improvement compared with results for which clustering was not performed. This indicates that the information required for WSD is complemented by adding features computed from clustering results. On the other hand, for the CRL system, we obtained only a 0.003 to 0.007 improvement relative to the results for which clustering was not performed. The CRL system achieve high WSD accuracy using a lot of features. Therefore, we consider that, even if more features are added to original features, they are not so effective to improve WSD accuracy significantly.

5 Conclusion

We verified how a semi-supervised clustering approach contributes to word sense disambiguation (WSD). We found that method II-2-1 that selects the word sense by using the D'Hondt method and randomly selects seed instances from ones that belong to the word sense is effective in semi-supervised clustering for word instances. We also found that the accuracy of WSD is improved by constructing a classifier using features such as CID, MI, and IG obtained from semi-supervised clustering results. In the future, we plan to develop a much more accurate semi-supervised clustering approach and look for features that can lead to higher accuracy for WSD.

References

1. Bar-Hillel, A., Hertz, T., Shental, N.: Learning Distance Functions Using Equivalence Relations. In: Proc. of the 20th International Conference on Machine Learning (ICML 2003), pp. 577–584 (2003)
2. Hotho, A., Nürnberger, A., Paaß, G.: A Brief Survey of Text Mining. GLDV-Journal for Computational Linguistics and Language Technology 20(1), 19–62 (2005)
3. Cai, J.F., Lee, W.S., Teh, Y.W.: Improving Word Sense Disambiguation Using Topic Features. In: Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP 2007), pp. 1015–1023 (2007)
4. Davidov, D., Rappoport, A.: Classification of Semantic Relationships betwen Nominals Using Pattern Clusters. In: Proc. of 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2008:HLT), pp. 227–235 (2008)
5. Klein, D., Kamvar, S.D., Manning, C.D.: From Instance-level Constraints to Space-level Constraints: Making the Most of Prior Knowledge in Data Clustering. In: Proc. of the 19th International Conference on Machine Learning (ICML 2002), pp. 307–314 (2002)
6. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)

7. Agirre, E., Ansa, O., Hovy, E., Martínez, D.: Enriching Very Large Ontologies Using the WWW. In: Proc. of 1st International Workshop on Ontology Learning (OL 2000). Held in Conjunction with the 14th European Conference on Artificial Intelligence (ECAI 2000) (2000)
8. Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., Weischedel, R.: OntoNotes: The 90% Solution. In: Proc. of the Human Language Technology Conference of the North American Chapter of the ACL (HLT-NAACL 2006), pp. 57–60 (2006)
9. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.J.: Distance Metric Learning with Application to Clustering with Side-Information. Advances in Neural Information Processing Systems 15, 521–528 (2003)
10. Katsavounidis, I., Kuo, C., Zhang, Z.: A New Initialization Technique for Generalized Lloyd Iteration. IEEE Signal Processing Letters 1(10), 144–146 (1994)
11. Shirai, K.: SENSEVAL-2 Japanese Dictionary Task. In: Proc. of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL-2), pp. 33–36 (2001)
12. Wagstaff, K., Cardie, C.: Clustering with Instance-level Constraints. In: Proc. of the 17th International Conference on Machine Learning (ICML 2000), pp. 1103–1110 (2000)
13. Wagstaff, K., Rogers, S., Schroedl, S.: Constrained K-means Clustering with Background Knowledge. In: Proc. of the 18th International Conference on Machine Learning (ICML 2001), pp. 577–584 (2001)
14. MacQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. In: Proc. of the 5th Berkeley Symposium on Mathmatical Statistics and Probability, pp. 281–297 (1967)
15. Nishio, M., Iwabuchi, E., Mizutani, S.: Iwanami Kokugo Jiten Dai Go Han. Iwanami Shoten (1994) (in Japanese)
16. Muggleton, S.: Inductive Logic Programming. New Generation Computing 8(4), 295–318 (1991)
17. Taagepera, R., Shugart, M.S.: Seats and Votes: The Effects and Determinants of Electoral Systems. Yale University Press (1991)
18. Basu, S., Banerjee, A., Mooney, R.: Semi-supervised Clustering by Seeding. In: Proc. of the 19th International Conference on Machine Learning (ICML 2002), pp. 27–34 (2002)
19. Specia, L., Stevenson, M., Nunes, M.G.V.: Learning Expressive Models for Word Sense Disambiguation. In: Proc. of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007), pp. 41–48 (2007)
20. Sugiyama, K., Okumura, M.: Personal Name Disambiguation in Web Search Results Based on a Semi-supervised Clustering Approach. In: Goh, D.H.-L., Cao, T.H., Sølvberg, I.T., Rasmussen, E. (eds.) ICADL 2007. LNCS, vol. 4822, pp. 250–256. Springer, Heidelberg (2007)
21. The National Language Research Institute. Bunrui Goi Hyou. Shuueisha (1994) (in Japanese)
22. Yarowsky, D.: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In: Proc. of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL 1995), pp. 189–196 (1995)
23. Niu, Z.-Y., Ji, D.-H., Tan, C.L.: A Semi-Supervised Feature Clustering Algorithm with Application to Word Sense Disambiguation. In: Proc. of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005), pp. 907–914 (2005)
24. Zhong, Z., Ng, H.T., Chan, Y.S.: Word Sense Disambigation Using OntoNotes: An Empirical Study. In: Proc. of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 2008), pp. 1002–1010 (2008)

Alleviating the Problem of Wrong Coreferences in Web Person Search

Octavian Popescu and Bernardo Magnini

papsi@racai.ro, magnini@fbk.eu

Abstract. In this paper we present a system for the Web People Search task, which is the task of clustering together the pages referring to the same person. The vector space model approached is modified in order to develop a more flexible clustering technique. We have implemented a dynamic weighting procedure for the attributes common to different cluster in order to maximize the between cluster variance with respect with the within cluster variance. We show that in this way the undesired collateral effect such as superposition and masking are alleviated. The system we present obtains similar results to the ones reported by the top three systems presented at the SEMEVAL 2007 competition.

Keywords: Web People Search, Cascade Clustering, Dynamic Threshold, Masking, Superposition.

1 Introduction

The exponential development of the Web brings with it the need for Web search tools. While for a certain class of queries the search engines on the market offer good answers, there is a significant part of queries that remains unfulfilled. Consulting the first 200 results returned by Google for the query "Bush engineer", one can notice that those pages refer only to two different persons. And one can learn unexpected connections between the US president, "George W. Bush", and the word "Engineer". However, it will probably be more useful if a list with different persons named "Bush" who are engineers would be returned instead.

The task of clustering the result pages of a search engine for a person name query according to the person they refer to has been recently undertaken in the Web People Search competition (WPS), under the Semeval 2007 workshop ([1]). The WPS task is slightly different from the Cross-Document Coreference task ([5]), which is the task of establishing coreferences among the entities present in a corpus. This is because some pages may be clustered together even without indicating which person mentions are actually corefered. In this paper we present a system for WPS task and its performances on the WPS test and training data set. The technique is based on corefered entities, so it is applicable to the cross-document coreference task as well.

Personal Names (PNs) are highly polysemous - the same name may stand for different persons ([2], [7]). Ideally, each PN can be disambiguated from the context. Unlike what happens in other disambiguation tasks, like word sense disambiguation for example, the context may be extended beyond the textual evidence. Therefore, one

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 280–293, 2009.

can expect to benefit from rich meta-contextual information, such as the name of the page, urls etc. But, on the other hand, a web page may be very heterogenous, an example being lists-like web pages, where each paragraph, consisting of very few words, introduces a different entity. Also, in a web page, some of the textual information may be substituted by page arrangement. It is very common for web pages to have a tabular format; therefore, the link between different text components - such as a pair name, e-mail address - is not directly expressed. The missing text makes it hard for various text processing tools, such as named entities recognizer, shallow parser etc., to analyze the document. This may result in a serious drop in performances.

The most common technique of corefering different instances is to represent their context as a set of vectors; the cluster structure is determined by computing the similarity among those vectors. The computation of a sensible threshold is rather an empirical question. One method to set this is to manually analyze the algorithm's performances on a training set. Unfortunately, WPS is a complex task, in the sense that the training set may be quite different from the test sets. Traditional clustering methods, such as agglomerative or hierarchic ones, have to be modified in order to avoid the masking effect. The masking effect appears when, by correctly identifying the instances of one (or two) dominant entities, a second (or third) less frequent entity is completely masked by the other one (two) and wrongly corefered with it. For example, in the first 200 pages returned by Google for the "Michael Jackson" query, only the famous singer is mentioned, in spite of the fact that the name by itself is a popular one, therefore there are many different people carrying it. By the masking effect the boundaries between two clusters are canceled due to the “insensitiveness” of the vector similarity metric (usually a quadratic formula, instead of at least a high degree polynomial one).

Another negative consequences of using vectors is the superposition effect, which is manifested when the vectorial space is dense. In this case spurious association between features and clusters are created. For example by searching “Bush Engineer” on the web will notice that the great majority of returned pages refers to the George W. Bush , one of the two former US presidents, and not to distinct engineers named Bush (with one exception).

Last, but not least importantly, there is the problem of stop conditions. Many clusters algorithms are requiring it as an input parameter, which is not possible for this task. The number of persons carrying the same name in a collection of web pages is hardly estimable at all at a prior time.

Our system is built while keeping in mind all these issues. While there is no definite solution, using alleviating techniques leads to a substantial improvement over a baseline determined by applying a "raw" clustering algorithm. We start by calculating extended vectors of terms ([3])which we subsequently modify. We call them association sets and our goal is for each association set to contain highly discriminative terms, and term chains. Association sets are structured in social networks ([8]), in order to compute the relevance of each term and establish the seed clusters. Clustering is realized hierarchically obeying the Fisher's principle of maximizing the between-cluster variance with respect to within-cluster variance.

The paper is organized as follows. In the next section we review the related work and point out the key issues. Section 3 presents the general architecture and the next sections detail each module. Section 4 specifies the preprocessing steps, resolving the

meta-context clues and building the initial association sets. Section 5 is devoted to the construction of social networks and to the presentation of the clustering algorithm. In Section 6 we analyze the results on training and test sets for Semeval 2007. The paper ends with Section 7 - Conclusion and Further Research.

2 Related Works

A system of resolving the cross-document coreference takes the same decision with each new person name mention (PNM). This is a mention of an already seen entity or a mention of a new entity. Let N be the total PNMs in the corpus. Let us suppose further that the real state of facts is that there are p persons and there are n_1, n_2, ...n_p mentions referring to each person respectively. Then, $S(n_1...n_p) = p$ and the possible configurations are isomorphic with the set of different decompositions of N as a sum of p terms, order relevant (all the ways to write N as addition using p numbers, with p going from 1 to N). The number of variants, which is our search space, is huge ([6]). The only chance is to find a heuristics that gives a resolution by taking local decisions.

In Baga ([3]), a vector space model is introduced and the clustering is done among all the mentions with their vector similarity higher than a predefined threshold. The vectors are computed based on term frequency, document frequency, and normalized cosine factor, and their similarity is computed by making the dot product. This approach is the base for almost all solutions proposed so far. The main variations are on the method of determining the length of the window from where the terms are picked-up, and the clustering method: agglomerative, hierarchic, spectral, merged, etc. A major drawback of this method comes from a property of the dot product called superposition. By adding high dimensional vectors, the vector sum is similar with any of the factor vectors. In IR, superposition is a desired feature for document clustering, because, generally, there are large documents and relatively few categories. If the coreference space is sparse enough (Baga and Baldwin report that in their study 24 out of a total of 35 PNMs stand for different persons) the dot product works fine. As "John Smith" is one of the most common names, the probability of many different contexts is high, therefore the search space is sparse enough. However, by considering more names, the coreference space is not that sparse and the similarity metric is indiscriminative, especially with a low threshold. In the work of Kulkarni and Pedersen ([12]) this phenomenon is observed. They choose the terms by computing the contingency table for all bigrams within a window around the target PNM and filter them out by using a statistical test. They notice an unexpected drop in performances if the correct number of entities is given as an input parameter. In our opinion, this is exactly what is expected to happen by the superposition principle if the space gets crowded. In conclusion, without further refinements, it is likely that a vector space model does not work properly for an arbitrarily large corpus, such as the Web.

Malin ([8]) and Wand ([15]) used random walks. PNMs are connected in a graph structure through the contextual terms. Each edge is weighted with a number representing the conditional probability of co-occurrence of the respective terms. The probability of corefering two PNMs is determined by the sum of weights on the possible paths in the graph between them. The length of a maximum path is empirically chosen. The

paths are considered social networks that define the domain of a person. These methods are particularly exposed to the masking problem. If a PNM stands for a famous person, then its social networks tend to be overrated. Therefore, a non-famous person carrying the same name is masked by spurious paths. The same problem occurs with the global estimation algorithm, such as the EM algorithm ([14], [16]) .

Mann and Yarowski ([9]) consider contextual terms based on their term frequency and mutual information. They also consider biographical dates as sure coreference features. The clusters realized by these features are considered anchors for further potential coreferences. A third type of feature is represented by dates occurring in special syntactic patterns. However, there is no analysis of the contribution of this type of features to coreference. It is a rather improbable event for two different pages to contain the same biographical information. Therefore, the anchors based only on the biographical date may have a reduced importance on the whole.

Some researchers ([4], [8], [13]) have shown that using Named Entities (NEs) is an effective way of addressing the ontology population, which is also a task that requires cross-document coreference. Also Ng, ([11]) has used NEs to cross document coreference. Our experiments also confirm that NEs play a major role in cross-document coreference. While the entire semantic analysis of the text is still an unresolved problem, we can we use a shallow parser to identify special key terms, especially NEs. In small, predefined syntactic structures, such as modifier or apposition, the relevance of terms is guaranteed. We show that we can use these features in a cascade like clustering algorithm. Cascade here stands for priority in clustering process of a set of features over another set (among others [16]).

In the next section we present a model designed to alleviate the problems related to the vector space model.

3 System Architecture

In this section we present the reasons we considered in order to device our algorithm. We start from empirical findings regarding the data distribution. We show that these findings suggest that, in some respect, the Web people Search is a very difficult tasks. In section 3.1 we describe two ways to approximate the difficulty of the task. Unlike the complexity of an algorithm that measures the number of operations, the difficulty of a task is given by the expected number of cases that are difficult to resolve. We introduce a few parameters, in order to compare the training and test sets. It is shown that WEP is a difficult task. The general architecture is described in section 3.2.

3.1 Data Variation and Expected Difficulty

One way to estimate the difficulty of a task is to estimate the required amount of training examples. However, if the data does not converge, the number of required training examples cannot be computed. One way to test the convergence is to analyze the variance between two sets of data. A big variation implies a large number of training examples, possibly infinite – a totally unpredictable behavior. Similarly, we can refer to the error prediction, but from the point of view of the proposed model. If the error on the test set after analyzing the training set is the one predicted, then the model has a constant behavior with respect to the task.

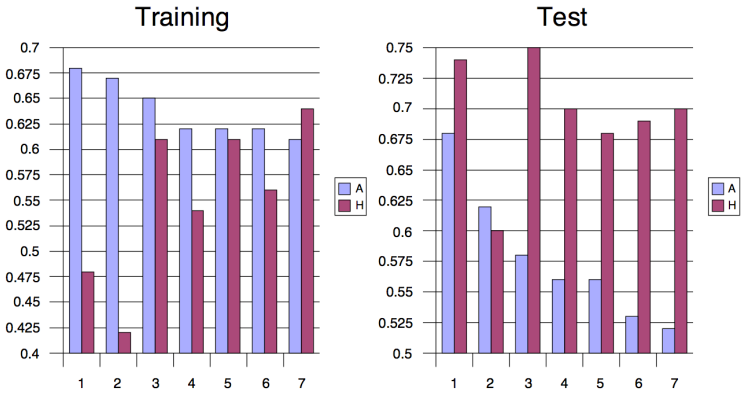


Fig 1. Variation in training and test (SEMEVAL 2007)

We randomly choose seven names from the training data and seven names from the test data and compare the results of two clustering algorithms, agglomerative (A) and hierarchic (H). The input was the same - the tf/idf computed on the whole corpus respectively. In Figure 1 we present the variation on the training and test data, 1a and 1b respectively.

In Table 1 we present the overall results and their variation. As it can be seen in the third column, the variation is big, but the most indicative clue comes from the comparison of 1a with 1b. As we can see, the performances of each model are determined by each individual case. This shows that one set cannot be used to predict the behavior of another.

Table 1. Total Variation on a sample of 14 names

	Agglomerative	Hierarchic	Variance
Training	.65	.61	.4
Test	.57	.69	.12
Variance	.6	.8	

A measure of the 'disorder' of a data set is given by the coreference density parameter. It is the ratio between the number of entities and the number of pages. The number of different person varies with the same name greatly; the figures are {32, 59, 83, 1, 19, 72, 2, 14, 60, 3, 37, 15, 32, 32}. If there are many cases with their coreference density near the extremes –like the "John Smith" example presented in Section 2, then the task is simple. However, for figures between 15 and 60 the tasks is rather complex.

Both measures indicate that WPS is a difficult tasks. It seems that the main problem a system for Cross Document Coreference is data variation. The same similarity measure and set of features may lead to totally different level of accuracy in different cases (different names). Therefore an effective and reliable cross document coreference system must provide explicit ways to deal with the above problems.

3.2 System Architecture

There are basically four steps that are taken. First, we compute the association sets for each PNM based on NEs, and each term is weighted differently. Second, using meta-contextual information and a very high threshold we determined the anchor pages. Third, the association sets of the anchor pages are compared using a graph structure and the weights are modified according to their discriminative power. Fourth, the rest of the pages are clustered using a low threshold by choosing the best match against the anchor pages. Step three is the core of the system. At this step we compute a specific difference among seed clusters. The common part is considered non discriminative. The indiscriminateness is decided for each term and a pair of anchor pages. In order to account for this fact, the weights are altered with a linear kernel (see Figure 2)

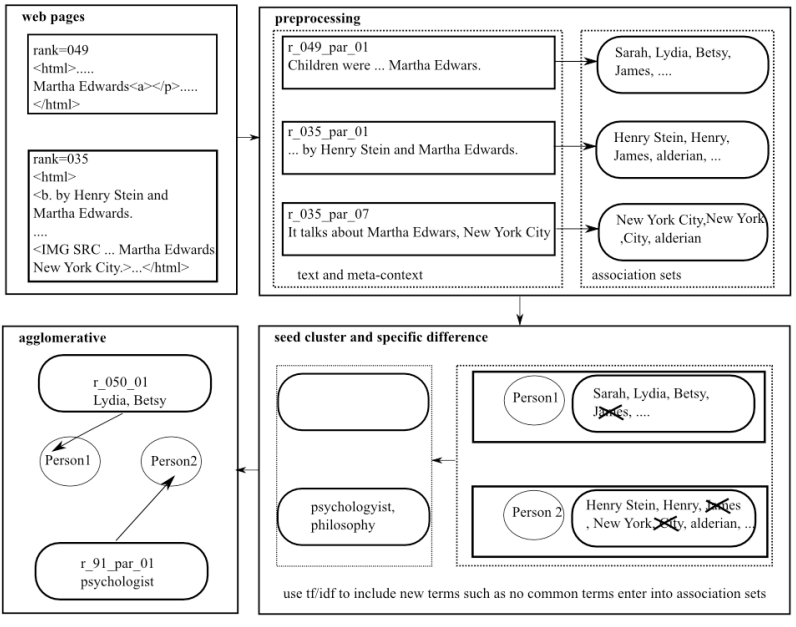


Fig. 2. System Architecture

In what follows we briefly describe the role of each module. A detailed description will be given in the next sections.

- Preprocessing.

  The text is passed through a Named Entity Recognizer, and a dependency parser; the term frequency is computed for all the nouns. Special scripts extract meta contextual information. The text is split in paragraphs such that only one occurrence of the target name is present. Each paragraph delimits the initial textual context of a paragraph entity. The set of all NEs and some of the terms from the same paragraph are kept in a list, called association set, with some initial weights. The coreference among paragraph entities is decided using association sets.

- Seed Clusters and Specific Difference.

  A high threshold is used to form the initial clusters in a two step cascade algorithm. We first consider the e-mails and phone numbers, and then the special NEs. At this step only the association sets that are mutually distinct are clustered. These clusters represent different persons, with a high probability. The association set of a cluster is made by unifying the association sets of the clustered paragraph entities. The terms in the association set have different weights that are dynamically changed such that the distance between any two seed clusters is maximal. New terms are introduced in the cluster association sets and the weights are adjusted according to the discriminative power of these terms.

- Agglomerative

  The unclustered paragraph entities are associated with one of the seed clusters if their association set is closer to a certain degree to the cluster association set. To each paragraph entities a list of candidate seed clusters is associated. The weights are readjusted according to a linear kernel computed dynamically according to the common part of the candidates. A non-corefered paragraph entity makes a cluster by itself.

4 Preprocessing

4.1 Feature Extraction

The first step consists in splitting the Web pages into paragraphs such that inside each paragraph there is only one mention of the target name. Only the full name was considered. Therefore, it is assumed that if a part of a target name appears within a paragraph, it always refers to the same person. This split is necessary, as some of the Web pages may contain long lists of names with many repetitions of the target name. If there are no tabular tags dividing the name occurrences, the paragraphs are not separated. A paragraph entity corresponds to each paragraph.

Secondly, from each paragraph, the e-mail addresses and the phone numbers are extracted by means of regular expressions. The graph in Figure 3 shows the number of times these were identified compared to their occurrences, for the seven test person

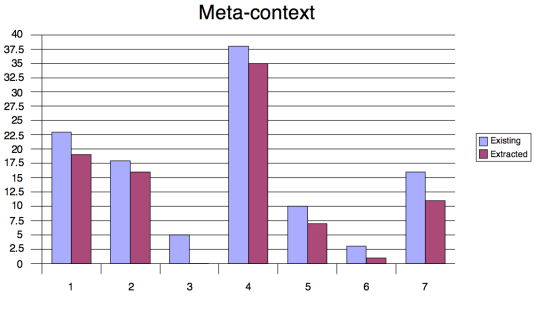


Fig. 3. Meta Context in test corpus

chosen in Section 3. These are very reliable features for coreference, but they have a low recall. In less than 5% of the coreferences, they are effectively used. However, this number may have a great variance according to the type of corpus.

Web pages can be processed by using an HTML parser (we used lynx). Unfortunately, the text is not output in a format that makes it appropriate for automatic processing. Therefore, we have preferred to write our own Perl scripts. Isolated words containing no verb, such as "psychologist", or "..." are rewritten within a full sentence. We have used the phrase "It talks about ..." in order to have full fledged sentences. We give two examples of such transformations:

<title>Arthur Ernest Morgan - Wikipedia, the free encyclopedia</title>
It talks about Arthur Ernest Morgan - Wikipedia, the free encyclopedia.
<img src ... Martha Edwards, New York City>
It talks about Martha Edwards, New York City.

In this way we have a neat input for the Name Entity Recognizer (NER). We have used an in-home built SVM NER, with state of the art performances. The text has been parsed using MiniPar and we extracted all the Ns that are included in dependency paths rooted in NEs. If the NEs are in the subject or object positions, the head of all their sisters are also considered. We call these nouns related nouns. The NEs and their related nouns are the main features of our approach.

4.2 Filtering and Weighting

An empirical linear kernel associates one of the scores in {5, 3, 2, 1} to each NEs. The NEs in the same sentence receive the maximum score if they are not separated by another name; Table 2 presents the kernel's table, where "SENT" stands for "in the same sentence" ,"SEP1" stands for "separated by a different name" and "SEP2" stands for "separated by a more than two names".

Table 2. Linear kernel weights associated with NEs

	not SEP	SEP 1	SEP 2
SENT (in sentence)	5	2	1
Not SENT	3	1	1

The order preserving parts , of a compound NEs are also considered. Therefore, each compound NE of n words contributes n(n-1) new terms.

In the related noun set we can find common nouns that have little relevance for coreference tasks. A classical way to measure the relevance of a term is to consider the ratio between its frequency inside a document and its frequency in the whole corpus. In our case, the document has the length of a paragraph and it is not very usual for the same NE to have more than two occurrences. Therefore tf/df is a biased measure in this case. We used the probabilities computed on the BNC corpus to decide whether a noun is common or not. All the nouns in the most used 500 nouns are considered common nouns and they are discarded. Implicitly, the stop words are discarded. If the number of occurrences of a rare word is higher on the whole corpus

than its estimation based on BNC, it is considered an important clue for coreference. According to this variation, the respective noun can be weighted from 1 to 5.

For all NEs and the related noun we computed the contingency table and we used the mutual information table to determine their relatedness factor.

5 SEED CLUSTERS

5.1 ASSOCIATION SETS

The first step in clustering is to identify the paragraph entities that are very similar; therefore, their coreference is highly probable. We used a high threshold, 12, which basically requires that only very closed NEs and special related nouns be considered. If the phone numbers or e-mails are available, the coreference is established without the use of the association set. The number of coreferences under this restrictive condition is low. The average number of clusters is 2,61 with a maximum of 5 distinct clusters. The average number of paragraph entities clustered is steadily 2, with few exceptions.

The second step concerns a subset of the clusters obtained before. We considered only the clusters who do not have any common NEs in their high and middle ranked terms, that is, those marked within [2..5]. We consider these clusters to stand for different people. These are used as seed clusters. Figure 4 shows the associations sets for two seed clusters for "Martha Edwards", an arbitrarily chosen name from the test data.

At the second step, the relevant terms in the association sets of seed clusters, i.e terms whose weights are bigger than 1, are expanded with the terms that are found relevant by mutual information. In the association set of "person1" enters "psychologist", "Jung", "Freud" etc. (see figure 5). "New York" is also extended, but, by itself, is not sufficient to overpass the coreference thresholds set for the next step.

The essential operation was to identify the possible "troubling" terms in the association sets. The main problem is to know which paths are safer to explore for coreference in order not to obtain spurious "social networks". By computing the specific difference we have a way to alleviate this problem.

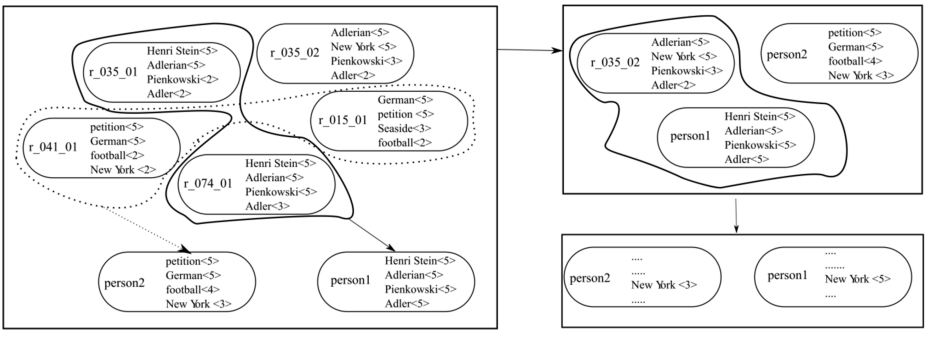


Fig. 4. Building the seed Clusters

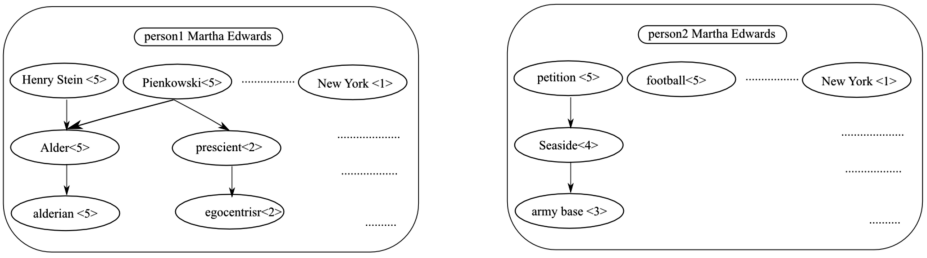


Fig. 5. Recalculating weights

5.2 Agglomerative Cascade Clustering

In the third step of the clustering algorithm the paragraph entities are added to the association sets. The threshold is dynamically set. If the similarity between a paragraph entity and a seed cluster is due to two terms that come from a path that does not contain devalorized terms - terms whose weight has been decreased - then the paragraph entity is corefered with the respective seed cluster. If the similarity contains only devalorized terms, then the coreference is postponed to the next cycle. The threshold is set to 3, and in 95% of the cases it takes at least one term of relevance 2.

When the coreference is realized, a maximum of two top relevance entities can enter the association set. If they are in the common set of two seed clusters then their weights are decreased. If some of the expansion terms obtained through mutual information are common to two different seed clusters, they are added with weight 1. That is, they are not expanded any more. In this way the overgrowing of association sets is stopped. In practice, the algorithm stops after three iterations, but the overwhelming majority of the coreferences are realized after the first iteration.

For each paragraph entity we compute a list of cluster candidates. The weights are dynamically modified according to the cluster candidate list. In the simplest case, the paragraph entity has disjunct intersections and the clusters do not have any "border issue" (The border of two cluster is created by the terms that are in common between two seed clusters.). In this case the similarity formula is given by:

$$sim(PE, C) = \sum w(t) \text{ , for all } t \text{ in } T \qquad (1)$$

PE is the respective paragraph entity and C is one of the candidate cluster T is the set of the common terms, w(t) is the weight of the term t. In this case the algorithm is a classical agglomerative one.

In case when a term, t is on the border of C, but the cluster candidates do not share that border, the w(t) is increased by 1/2;

$$sim(PE, C) = \sum w(t) + \sum (w(t') + 1/2) \qquad (2)$$

However the threshold is also raised with two. That is, the system asks for more evidence. A border term, even with weight 2, is not sufficient as evidence for coreference. With the formula (1) it would have passed.

When the term t is on the border of two competing clusters, its weight is zero (it is not considered). The formulas (1) and (2) make use of weights without any factor terms. But this is only apparent. The related nouns, are initially weighted according to

```
preprocessing;
        extractDependencyPaths;
        computeNEsWeights(linearKernelSchema);
        filtering_out(computetf_idf;mutualInformation;chiSq);

computeSeedClustersSpecificDifference
        threshold = 12;
        foreach pair (P1,P2) of type paragraphEntity
                computeSimilarity(Formula1);
                if (sim(P1,P2) >= threshold)
                        cluster(P1,P2);
        endforeach;
        foreach (C1,C2) of type cluster
                computeSpecificDifference;
                reweightCommonParts;

agglomerativeCascadeClustering
        threshold=4;
        foreach P1 of type paragraphEntity
                foreach C1 of type cluster
                        computeCommonpart(P1,C1)
                        if (commonPart > 0)
                        push C1, clusterCandidateList;
                endfor
                foreach pair (C1, C2) of type cluster in clusterCandidateList
                        foreach t of type term in P1
                        if (t in comm(C1,C2))
                                threshold = 6;
                                computeSimilarity(Formula2);
                        else
                                threshold = 4;
                                computeSimilarity(Formula1);
                if (sim(P1,Ci) >= threshold)
                        cluster(P1,Ci); //i from 1 to 2
        endforeach;
```

Fig. 6. Cross Document Coreference Algorithm

their tf/df coefficient and their mutual information score, and only after that are the weights normalized. By clustering and recomputing the weights, the within-cluster frequency is computed. Therefore, the product factors are implicit in both (1) and (2). See the algorithm in pseudo code in Figure 6.

Usually the association set of each paragraph entity has more than one possible cluster to associate with. The average number of common terms is 2,05. That is, it is improbable that the common set has more than three or more terms in common. If a relevant term is seen then the clustering is realized. However, it is hard to have relevant terms in common. Actually, this is a reflection of the bias/variance principle. We revert to this in the next section, after the presentation of the results.

6 Evaluation

The test data for WPS is made out of three different categories of names for a total of 30 names: (1) names of the people who are involved in ACL, (2) names chosen arbitrarily from census data and (3) names of the persons that have Wikipidia pages The results show a strong improvement over a baseline of a simple agglomerative clustering method based on bag of words. We use the official WPS scorer with α=0.5.

In Table 3, in the first column, “A”, the agglomerative results are given, in the second column “H” stands for hierarchic clustering and “S” for our system.

Table 3. Results on WPS test set

	Agglomerative (A)	Hierarchic (H)	System (S)
Group1	.66	.70	.77
Group2	.56	.60	.73
Group3	.62	.68	.78

The results in Table 3 confirm that the variation in data is great. A measure of the 'disorder' of a data set is given by the coreference density parameter. It is the ratio between the number of entities and the number of pages. If there are many cases with their coreference density near the extremes, then the task is simple. In Figure 7 we present these figures for the first set, the ACL names. For each name all its 100 instances are considered. As we can see, this parameter also indicates a great variation.

Knowing the number of the clusters would help both agglomerative and hierarchical algorithms to increase their performances by computing a new threshold. However, setting the parameters for one group of people is likely to produce over fitting. The over fitting effect is amplified by the variation of data from one set to another.

In Figure 8 we draw separately the graph of the results for each set in order to have a better view. We can notice that the problems were not solved entirely. First of all, the number of non-corefered entities is always too large. We have manually inspected the non-corefered pages for seven people (see Section 3, Figure 1b). We have found that in more than roughly 70% of the cases, the relevant information was missing from the text files. In our opinion, this is a bottleneck. It seems that a large body of world knowledge is required in order to correctly resolve the top 20% difficult cases. Another issue concerns the spurious coreferences. As can be read from table 3, the system's performances are less accurate on Wikipedia People. This is the problem of superposition, when the space gets crowded.

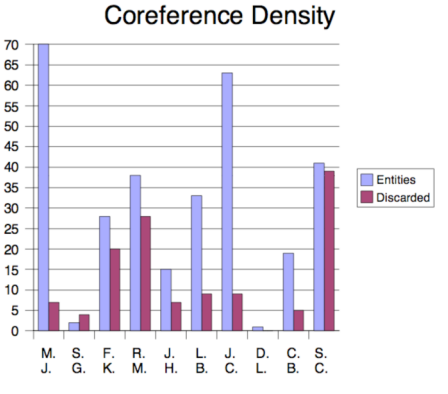


Fig. 7. Coreference Density computed on ACL group

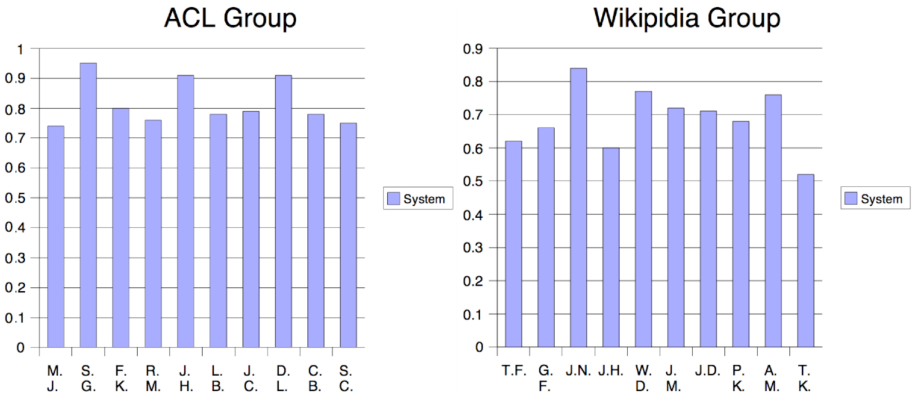


Fig. 8. The System compared against G=old Standard

7 Conclusions and Further Research

We have presented a system for the WPS task based on the resolution of the Cross-Document coreference task. Its main characteristics is that it expands the association sets according to a strict principle - the preservation of the specific difference between the clusters that have a high probability to represent different people. This leads to a methodology that dynamically modifies the weights. The system ranks among the best on the WPS task. However, there are still some debatable issues that require more research.

The first thing is to exploit a Markov chains approach in order to determine the specific difference. Here we have implemented a simple approach which is basically a one step random walk between two seed clusters. Our approach differs from the usual implementation in that if a seed cluster node is reached by starting from another seed cluster, the respective intermediary nodes are weighted with a small number.

Secondly, a much better HTML parser should be developed. Some piece of information is transmitted via formatting tabs. Therefore, relevant information can be deduced recursively. It will also be interesting, and very useful, to implement a heuristics that determines the typology of a page, according to a close class of possibilities.

Thirdly, a more efficient method to extract related terms needs to be implemented. While the syntactical patterns offer a good accuracy, their recall is low. One possibility is to consider all the terms in the lexical family of a noun, especially the adjectival ones. Besides, a shallow encyclopedic knowledge repository may be extremely beneficial.

References

[1] Artiles, J., Gonzalo, J., Sekine, S.: Establishing a benchmark for the Web People Search Task: The Semeval WePS Track. In: Proceedings of Semeval 2007, Association for Computational Linguistics (2007)

[2] Artiles, J., Gonzalo, J., Verdejo, F.: A Testbed for People Searching Strategies in the WWW. In: SIGIR 2005 (2005)

[3] Bagga, A., Baldwin, B.: Entity-based cross-document co-referencing using the vector space model. In: Proceedings of the 17th international conference on Computational linguistics, pp. 75–85 (1998)
[4] Buitelaar, P., Cimiano, P., Magnini, B. (eds.): Ontology Learning from Text: Methods, Evaluation and applications. IOS Press, Amsterdam (2005)
[5] Grishman, R.: Whither Written Language Evaluation? In: Human Language Technology Workshop, pp. 120–125. San Mateor Morgan Kaufmann, San Francisco (1994)
[6] Luo, X., Ittycherian, Y., Jing, H.: A mention-synchronous coreference resolution algorithm based on the Bell tree. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Barcelona (2004)
[7] Magnini, B., Pianta, E., Popescu, O., Speranza, M.: Ontology Population from Textual Mentions: Task Definition and Benchmark. In: Proceedings of the OLP2 workshop on Ontology Population and Learning, Sidney, Australia. Joint with ACL/Coling (2006)
[8] Malin, B.: Unsupervised Name Disambiguation via social Network Similarity. In: Proceedings of the SIAM Workshop on Link Analysis, Counterterrorism and Security, CA (2005)
[9] Mann, G., Yarowsky, D.: Unsupervised Personal Name Disambiguation, CoNLL, Edmonton, Canada (2003)
[10] Niu, C., Li, W., Srihari, R.: Weakly supervised learning for cross-document person name disambiguation supported by information extraction, ACL, Spain (2004)
[11] Ng, V.: Shallow Semantics for Coreference Resolution. In: IJCAI (2007)
[12] Pedersen, T., Purandare, A., Kulkarni, A.: Name discrimination by clustering similar contexts. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 226–237. Springer, Heidelberg (2005)
[13] Popescu, O., Magnini, B., Pianta, E., Serafini, L., Speranza, M., Tamilin, A.: From Mentions to Ontology: A Pilot Study. In: Proceedings SWAP 2006, Pisa, Italy (2006)
[14] Song, Y., Councill, I., Li, J., Giles, C.: Efficient Topic-based Unsupervised Name Disambiguation. In: JCDL 2007, Vancouver, British Columbia, Canada (2007)
[15] Wan, X., Yang, J., Xiao, J.: Using Cross-Document RandomWalks or Topic-Focused Multi-Document. In: Proceedings of IEEE/WIC/ACM, International Conference on Web Intelligence (2006)
[16] Wei, Y., Lin, M., Chen, H.: Name Disambiguation in Person Information Mining. In: IEEE/WIC/ACM International Conference on Web Intelligence (2006)

Improved Unsupervised Name Discrimination with Very Wide Bigrams and Automatic Cluster Stopping

Ted Pedersen

University of Minnesota, Duluth, MN 55812, USA

Abstract. We cast name discrimination as a problem in clustering short contexts. Each occurrence of an ambiguous name is treated independently, and represented using second–order context vectors. We calibrate our approach using a manually annotated collection of five ambiguous names from the Web, and then apply the learned parameter settings to three held-out sets of pseudo-name data that have been reported on in previous publications. We find that significant improvements in the accuracy of name discrimination can be achieved by using very wide bigrams, which are ordered pairs of words with up to 48 intervening words between them. We also show that recent developments in automatic cluster stopping can be used to predict the number of underlying identities without any significant loss of accuracy as compared to previous approaches which have set these values manually.

1 Introduction

Person name ambiguity is an increasingly common problem as more and more people have online presences via Web pages, social network sites, and blogs. Since many distinct people share the same or similar names, it is often difficult to sort through results returned by search engines and other tools when looking for information about a particular person. There are many examples of identity confusion that have been widely publicized. For example, television talk show host Charlie Rose included his friend George Butler the filmmaker in his list of notable deaths from 2008. The only problem was that this George Butler was still alive, and it was George Butler the recording company executive who had died.

In general the goal of name discrimination is to associate or group the occurrences of person names with their true underlying identities. Our approach is completely unsupervised, and relies purely on the written contexts surrounding the ambiguous name. Our goal is to group these contexts into some (unspecified) number of clusters, where each cluster is associated with a unique individual. We assume that named entity recognition (NER) has already been carried out, so the input consists of text where the occurrences of person names are already identified.

There are various ways to formulate solutions to the problems surrounding name ambiguity or identity confusion, and so this paper tries to clarify exactly

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 294–305, 2009.

where this works falls in that spectrum. We make distinctions between our approach of name discrimination versus name *disambiguation*, and between our approach of treating contexts independently versus those that consider them to be dependent (as in cross–document co–reference resolution). We then go on to describe the general methodology of representing contexts with second–order co-occurrences, and then our specific enhancements to that approach that we arrived at via an extensive comparison with previously published results. In particular we focus on using very wide bigrams, which allow for up to 48 intervening words between an ordered pair of words, and on the use of automatic cluster stopping based on the clustering criterion function.

2 Discrimination versus Disambiguation

Name discrimination and name disambiguation are often confused or viewed as the same problem, yet there are important differences.

Name ambiguity can be approached as a problem in word sense disambiguation, where the goal is to assign a meaning to the surface form of a word. In this case, the different forms of a name (e.g., John Smith, Mr. Smith) are surface forms, and the underlying meaning is the unique identity associated with each occurrence of a name. In the case of disambiguation, these identities must be specified in a pre–existing inventory. In word sense disambiguation these inventories take the form of dictionaries, whereas in name disambiguation the inventory may be found in a biographical database or Wikipedia.

The key characteristic of name disambiguation is that the nature and number of possible identities is specified in advance of solving the problem. This means that discovering the number of identities or the nature of those identities has already been accomplished via some means, and is not a part of the problem.

However, in many practical settings a pre–defined inventory of possible identities simply isn't available. In that case, the best that can be hoped for is to carry out *name discrimination*, where we cluster the contexts in which the surface forms of a name occur into groups and thereby discover the number of distinct identities that share the same name. It may also be possible to describe each unique identity by analyzing the contents of each cluster; this is a task we refer to as *cluster labeling* (c.f., [6]). Given the rapidly changing nature of the online world, we believe that in general it's not realistic to assume that a pre–existing identity inventory will be available or can easily be constructed, so our work has focused on discrimination.

For example, in Web search it is very hard to predict what names a user might choose to search for, and the number of new names and identities that appear on the Web changes rapidly. A user searching for a given name (e.g., George Miller) will find some number (often more than they expect) of pages that include this name. The user must then determine which pages belong to the *George Miller* they are interested in, and which ones are about a different *George Miller*. As the user surveys the results, they may ask questions like : "... Did the George Miller that developed WordNet also write *The Magical Number Seven*?"

(Yes). "... Did Professor George Miller of Princeton direct the movie *Mad Max*?" (No). Performing this type of discrimination task manually can be difficult and error prone (as the Charlie Rose example above possibly suggests). Automatic methods that help organize the different contexts in which a name occurs into different clusters could be useful in identifying the underlying identities more conveniently and reliably.

The problem of name discrimination has drawn increasing attention, including the recent Web People Search Task (WePS) at SemEval-2007 [1], which included 16 participating teams. There is a second Web People Search Task taking place in late 2008 and early 2009 as well. The goal of the WePS tasks is to cluster Web pages based on the underlying identity of a given name. This is a variant of the name discrimination task where the context in which the name occurs is an entire web page. This task resulted in the release of a manually disambiguated corpus of names. It consists of 4,722 contexts (web pages) in which 79 names occur. These names represent a total of 1,954 entities. This corpus has a average of 59.8 pages per name, and 24.7 entities per name.

3 Relation to Cross–Document Co–reference Resolution

Our approach to name discrimination treats each context as an independent occurrence of a name, and clusters each occurrence without regard to the document from which it originally came (or its position within that document).

It is also possible to take the view that all the occurrences of a name in a document collectively form a single context, and simply assume that all of the surface forms of a name in that document refer to the same underlying identity. This is a slight variation on the one sense (of a word) per discourse hypothesis of Gale, Church and Yarowsky [3] for word sense disambiguation.

This representation of context is characteristic of cross-document co–reference resolution, where the problem is to determine if the surface forms of a particular name found in multiple documents refer to the same identity or different ones. The sequence of named entities in a document are referred to as chains, and the goal is to link together chains across documents that refer to the same person.

While this might appear to be a somewhat different problem, it is in fact very similar to name discrimination. The only real difference is the size of the context, which now includes all the occurrences of a name in a document, and either the sentences in which they occur or some fixed size window of surrounding context. As a result, the methods we describe in this paper can be used without modification to solve cross–document co–reference resolution simply by allowing contexts to include multiple occurrences of a name. If there is only one occurrence of a name in a document, then it is exactly equivalent to the name discrimination problem.

In fact, linking chains of references found across documents is nearly equivalent to assigning the contexts in which a name occurs to a cluster. The only difference is that the order of the documents is preserved when linking rather

than clustering. However, such orderings are relatively easy to recover from a cluster, especially if the document names indicate the order.

An early approach to cross–document co–reference resolution is described by Bagga and Baldwin [2]. They identify co–reference chains within a document between the multiple occurrences of a person name. Then they create a first order bag of words feature set from the sentences that occur around the person names in the chain in order to create a context vector that will represent the person name in that document. Thus, there is one vector per person name per document, and these vectors are clustered to determine the number of underlying individuals that share that name. They make pairwise comparisons between these vectors and those that are sufficiently similar (according to a predetermined threshold) are judged to refer to the same underlying identity and are placed in the same cluster. They did experiments using the John Smith Corpus, which includes 197 articles from the 1996-1997 New York Times that include the surface form John Smith (or various variations). There were 11 different entities mentioned more than one time in 173 articles, and 23 singleton entities, leading to 197 total articles in the corpus.

Gooi and Allan [4] evaluated several statistical methods on the John Smith corpus, and on their Person-X corpus, which is made up of name conflations or pseudo–names, where multiple named entities are disguised by replacing their surface form with Person-X. They created this corpus by searching for different subject domains in TREC volumes, which yielded 34,404 documents. Then, each document was processed with a named entity recognizer. A single person name was randomly selected from each document, and disguised throughout all the documents with Person-X. There were 14,767 distinct entities that were disguised.

Each occurrence of Person-X is represented by a first order bag of words created from a 55 word snippet of text that surrounds the entity. The authors experimented with incremental and agglomerative vector space models, as well as Kulback-Liebler divergence. They report that agglomerative clustering of the vectors fares better than the incremental clustering methods of Bagga and Baldwin [2]. In this case the number of clusters was not specified apriori, rather a threshold was set that determined if one chains was sufficiently similar to another to be grouped together.

In our earlier work we drew inspiration from cross–document co–reference resolution research, and in particular utilized first order context vectors to represent the context surrounding ambiguous names. While this sometimes works quite well, we often found that we did not have large enough numbers or sizes of context to obtain sufficient feature coverage to discriminate names reliably.

4 Name Discrimination Methodology

Over the last few years, we have developed an unsupervised approach to name discrimination, where our input is N contexts in which a single surface form of a given name occurs. Our goal is to group these contexts into k clusters, where the

value of k is automatically determined. Each discovered cluster is made up of contexts associated with a unique person, and the number of clusters indicates the number of distinct individuals that share the given name. Since this is name discrimination, each context is treated independently and there is no pre–existing sense inventory.

The evolution of this approach has been described in a series of publications that began at CICLing-2005 [14] and continued at IICAI-2005 [6] and CICLing-2006 [13]. In those papers, we explored different ways of constructing second–order context vectors, and the effect of Singular Value Decomposition (SVD) on the context representation prior to clustering. At this stage in the development of this work, we would manually set the number of clusters to be discovered to the proper value, or some arbitrary value.

This work continued to evolve, with an emphasis on comparing first and second–order feature representations, and studying the effect of SVD. In addition, we developed numerous methods to automatically identify the number of clusters/identities in a data set (c.f., [5], [10]). This work was reported on at an IJCAI-2007 workshop [11] and CICLing-2007 [12].

It must be said that it is impossible to provide a short summary of the different methods used in these five previous papers, since in each paper many different settings were employed for each word, and the best results per word were reported. Thus, there is no single method that emerged from those earlier papers as dominant, rather these were explorations of the capabilities of the SenseClusters system, and the range of possibilities for solving this problem in general. Our goal in this paper is to try and arrive at a more generic method of name discrimination which can hopefully be applied to a wide range of data with good effect.

One of the dominant themes in our work has been the use of second–order context representations. In this framework, the contexts in which ambiguous names occur are approximately 50 words wide. All of the bigrams in the contexts to be clustered that have a log–likelihood ratio or pointwise mutual information (PMI) score above a pre–determined threshold are identified as features. These bigrams are then used to construct a word by word matrix, where the first words in the bigrams represent the rows, and the second words in the bigrams represent the columns. Then, each context with an ambiguous name to be discriminated is re–processed, such that every word in that context that has a row entry in this word by word matrix is replaced by the vector formed by that row. This replacement step creates the second–order representation, where the words in the context are now represented not by the words that occur in that context, but by the words that occur with them in the contexts to be clustered. After all possible substitutions are made, the vectors are averaged and then the resulting vector becomes the representation of the context. Rather than bigrams, co–occurrences can be employed exactly as described above. The only difference is that since co–occurrences are not order dependent, the resulting word by word matrix that is created is symmetric.

We have also used first–order contexts, where unigram features are identified via frequency counts, and then the contexts are are represented simply by indicating which words have occurred surrounding the ambiguous name.

After the context representation has been created (whether it is first or second order), modified forms of k–means clustering are performed using the PK2 or the Adapted Gap Statistic method of cluster stopping. In either case clustering is performed on a range of values of k in order to determine which best fits the data, and automatically determine the number of underlying identities.

Each ambiguous name is is processed separately, and evaluation is done via the F-Measure, which finds the maximal agreement between the discovered clusters and the actual identities of the names. Note that this style of evaluation results in stiff penalties if the method predicts the wrong number of clusters.

5 Development Experiments

Our most recent work is reported on in papers at CICLing-2007 [12] and an IJCAI-07 workshop [11]. In those papers we experimented on the Kulkarni Name Corpus[1]. This consists of 1,375 manually disambiguated contexts, each of which are approximately 100 words wide. The center of each context includes one of five different ambiguous names as retrieved from the Web in May 2006. Over the five names a total of 14 different identities are represented, which results in an average of 2.8 identities per name. Note that there was is some variation in the surface forms of names in this data, where first initials or titles may also be used (e.g., Professor Miller, G. A. Miller, etc.)

The names, the number of distinct identities (I), the number of contexts (N) and the percentage of the most common identity per name (the Majority class) are shown in Table 1. Note that the Majority class corresponds to the F-measure that would be obtained by a simple baseline method that simply assigns all the contexts for a given name to a single cluster (in effect assuming that there is no ambiguity in the name).

As we reviewed our results from these 2007 papers, we noticed that some of the Web contexts were relatively impoverished and had very little text. Our features in these papers were based on using unigrams and first–order contexts, or adjacent bigrams and second–order contexts. We realized that for this data, first order representations of contexts might have little chance of success, since there is such a small number of features. While second–order features are sometimes seen as a solution to small data problems, there are limits to have effective they can be with very sparse or noisy feature sets.

Thus, we decided to try and increase the amount of context by using bigrams and co–occurrence features with very wide windows. This means that rather than requiring that a pair of words occur together to form a bigram (in order) or a co-occurrence (in either order), we would allow up to 48 intervening words to occur between them. This will increase the number of bigram or co–occurrence features available for second order methods, and we felt that might be a promising method

[1] http://www.d.umn.edu/~tpederse/namedata.html

Table 1. Development Results with Kulkarni Name Corpus

Name	I	N	Maj.	Best	(k)	New-Coc	(k)	New-Big	(k)
IJCAI-2007, CICLing-2007:									
Sarah Connor	2	150	72.7	90.0	(2)	79.0	(3)	**100.0**	(2)
Richard Alston	2	247	71.3	**99.6**	(2)	**99.6**	(2)	98.8	(2)
George Miller	3	286	**75.9**	**75.9**	(1)	61.2	(4)	61.9	(3)
Ted Pedersen	3	333	**76.6**	**76.6**	(1)	61.3	(3)	62.2	(3)
Michael Collins	4	359	74.9	93.0	(3)	88.9	(4)	**94.4**	(3)

of improving performance. In addition, we were concerned that our methods were a bit brittle. For example, we used the log–likelihood ration or Pointwise Mutual Information (PMI) to identify the bigram and co–occurrence features. However, in both cases the values that we use for determining which bigrams and co–occurrences are interesting will vary depending on the sample size, and so there isn't a clear way to make this process fully automatic. Finally, we also observed that the Adapted Gap Statistic, which we used for cluster stopping in the 2007 experiments, had a tendency to simply find one cluster, which resulted in F-measures that sometimes converged on the Majority class percentage.

Thus, we made a few modifications to our approach for this most recent round of experiments. First, we allowed the words in the bigram or co-occurrence to be separated by up to 48 intervening words, which greatly increases the number of bigrams that are considered as possible features. Second, we switched to using Fisher's Exact Test for identifying significant bigrams [9], which is relatively robust in the face of changing sample sizes, and allows for a single p–value (0.99) to be used for assessing significance. We also began to use the PK2 method of cluster stopping as our default method. We conducted an extensive series of experiments on the Kulkarni Name Corpus using these new ideas, and in general found some improvement in results. However, we wanted to make sure that we were not tuning the results too closely to this one particular data set, so after arriving at what appeared to be a robust and effective set of parameter settings[2] we evaluated those on data sets we had used in earlier name discrimination experiments.

Our final choices for the parameter settings based on both our intuitions and observed results on the Kulkarni Name Corpus as are follows:

1. A context of 50 words to the left and right of the ambiguous name is used both for feature selection and context representation.
2. Bigrams or co–occurrences may have up to 48 intervening words between them (a 50 word window), and are selected based on Fisher's Exact test (left-sided) with a p–value of 0.99.
3. Any bigram or co–occurrence that includes at least one stop word from the standard Ngram Statistics Package (version 1.09) stop list is discarded, and any bigram or co–occurrence that occurs less than 2 times is discarded.
4. SVD is not employed.

[2] These parameter settings refer to option values given to the SenseClusters package, which is the tool we have developed and used in all of our name discrimination work.

5. Cluster stopping is done with the PK2 method. This means that clustering is done with k–means for successive values of k, and the criterion function for clustering indicates at what value of k we should stop.

In our experiments with the Kulkarni Name Corpus we noticed fairly significant advantages to using bigrams rather the co-occurrences. This surprised us since the only difference between them is that bigrams are ordered pairs of words while co–occurrences occur in either order. Even though we observed better performance for bigrams, we decided to continue to study the difference between these two kinds of features (where everything else is held steady) in the held–out data as well.

The results of our experiments on the Kulkarni Name Corpus with the settings above are shown in Table 1. The high F-Measure in the 2007 papers for a given name is shown in the column labeled Best, and it's important to understand that this is the best result selected from a large number of different methods reported on in those two papers. Our goal in this paper is to arrive at a single set of parameter settings that will result in more consistent and accurate performance across a range of names. After the Best column we show the number of clusters (k) discovered by the Adapted Gap Statistic. We then show the F-Measure obtained using co–occurrence features (New-Coc) and bigram features (New-Big). Both of these values are followed by the number of clusters predicted by PK2.

In general we were pleased with the results on the development data set. For three of the names the results of the new settings are as good or better than the previous results (which are the best of over 60 different experimental settings for each word). However, the results for *Ted Pedersen* and *George Miller* were disappointing. In the case of TP this was because one of the identities was associated with e–commerce sites that had little textual content, and were essentially unrepresented by textual features. A similar effect was observed for GM, where the movie director identity had quite a number of low text contexts, and again it was difficult to represent that identity. However, in our development experiments we achieved an F-measure of 88.29 on TP using SVD, a window size of 10 for the bigrams, and a much smaller window of 5 words to the left and right of TP (rather than 50) when building our second order representation of the context. However, this combination of settings seemed to be uniquely effective with TP. In general GM very rarely rose above the value of the Majority class even in our development experiments.

6 Evaluation Data

After arriving at the New-Big and New-Coc parameter settings described above, we evaluated them on the datasets used our papers from CICLing-2005 [14], IICAI-2005 [6] and CICLing-2006 [13]. These datasets were kept out of the development process completely, and were only processed with our new sets of parameter settings (New-Big and New-Coc).

The data from 2005 and 2006 is not manually disambiguated, but rather is a more artificial form of data that is based on creating ambiguous names from relatively unambiguous names found in newspaper text. This is very much like

pseudo-words have been created for word sense disambiguation experiments, where for example all occurrences of the words *banana* and *door* are combined together to create the newly ambiguous word *banana–door*.

In creating the pseudo–name data, we tried to select names to conflate together that might have some relation to each other, in order to avoid obviously easy cases. Discriminating between two soccer players, for example, is probably more difficult than discriminating between a soccer player and an investment banker (due to the distinct contexts in which these names will occur).

Once the names to be conflated are determined, we select some number of contexts containing one of these names from the English GigaWord Corpus, which consists of newspaper text from the 1990's and 2000's. Then all the occurrences of the names to be conflated are replaced by a pseudo–name which is now ambiguous. For example, we identified all occurrences of the different forms of *Bill Clinton* and *Tony Blair*, and conflated those together into a newly ambiguous name that could mean the former US President or the former British Prime Minister. In creating the pseudo–names, we also controlled the frequency distribution of the individual names to provide a variety different experimental scenarios. The creation of the pseudo–name data was carried out with our *nameconflate* program[3].

The pseudo-names we used in previous studies are shown in Table 2. These are referred to via their abbreviations as used in the original papers, except for the IICAI-2005 data where we only referred to the original names. In that case we have introduced abbreviations. This table also shows the number of identities (I), the total number of contexts (N), and the percentage of the most common underlying identity (Majority class).

The details surrounding the creation of this data and the underlying identities can be found in the original publications, but but briefly the CICLing-2005 data includes pseudo-names with 2 underlying identities. These include two soccer players (Robek : David Beckham and Ronaldo), an ethnic group and a diplomat (JikRol : Tajik and Rolf Ekeus), two high–tech companies (MSIBM : Microsoft and IBM), two political leaders (MonSlo : Shimon Peres and Slobodon Milosovic), a nation and a nationality (JorGypt : Jordan and Egyptian), and two countries (JapAnce : Japan and France). Note that these are generally unambiguous, although *Jordan* no doubt includes some contexts referring to the basketball player *Michael Jordan* and *Ronaldo* is a relatively common name outside of professional soccer.

In the IICAI-2005 data there are 2 and 3-way ambiguities. The identities are indicated via the initials of a person name of the first few letters of a single entity name. The 2-identity pseudo-words include people occupying similar roles in the world : Tony Blair and Vladimir Putin; Serena Williams and Tiger Woods; and Sonia Gandhi and Leonid Kuchma. It also includes pairs of entities of the same type : Mexico and Uganda (countries) plus Microsoft and Compaq (high–tech companies). The three identity conflations are based on mixing names found in the 2–way distinctions: Tony Blair, Vladimir Putin, and Saddam Hussein; Mexico, Uganda, and India; and Microsoft, Compaq and Serena Williams.

[3] http://www.d.umn.edu/~tpederse/tools.html

Table 2. Evaluation Data Results

Name	I	N	Maj.	Best	New-Coc	(k)	New-Big	(k)
CICLing-2005:								
Robek	2	2,542	69.3	**85.9**	46.5	(13)	38.5	(14)
JikRol	2	4,073	73.7	96.2	99.0	(2)	**99.4**	(2)
MSIBM	2	5,807	58.6	**68.0**	57.4	(3)	58.0	(3)
MonSlo	2	13,734	56.0	96.6	**99.5**	(2)	**99.5**	(2)
JorGypt	2	46,431	53.9	**62.2**	58.8	(3)	54.8	(2)
JapAnce	2	231,069	51.4	51.1	51.4	(2)	**51.5**	(2)
IICAI-2005:								
SG-LK	2	222	50.5	91.0	83.4	(3)	**97.8**	(2)
SW-TW	2	599	51.4	69.0	**83.8**	(3)	69.0	(2)
Mi-Co	2	760	50.0	70.3	80.8	(3)	**83.0**	(3)
Me-Ug	2	2,512	50.0	60.1	62.8	(3)	**63.5**	(3)
TB-VP	2	3,224	55.5	96.2	**96.7**	(2)	**96.7**	(2)
Mi-Co-SW	3	1,140	33.3	56.6	**94.2**	(3)	66.3	(2)
Me-Ug-In	3	3,768	33.3	46.4	64.1	(4)	**64.6**	(4)
TB-VP-SH	3	4,272	41.9	75.7	**94.9**	(3)	72.4	(2)
CICLing-2006:								
Me-Ug	2	2,512	50.0	59.2	62.8	(3)	**63.5**	(3)
BC-TB	2	3,800	50.0	**81.0**	50.0	(3)	75.0	(3)
IBM-Mi	2	5,807	58.6	**63.7**	57.4	(3)	58.0	(3)
BC-TB-EB	3	5,700	33.3	47.9	55.7	(3)	**56.5**	(3)
Me-In-Ca-Pe	4	6,000	25.0	**28.8**	26.4	(2)	27.9	(3)

For the CICLing-2006 data we used similar identities to create 2, 3 and 4-way ambiguities, including : Bill Clinton and Tony Blair; Microsoft and IBM; Mexico and Uganda; Bill Clinton, Tony Blair, and Ehud; and Mexico, India, California, Peru.

7 Discussion and Future Work

In Table 2 we show the Best results from the original publications from 2005 or 2006, and then the results with our new settings of New-Coc and New-Big. Note that there is a very significant difference in the old and new methods; the old results (Best) required that the number of clusters be manually set to the known value, whereas in the new method this has been automatically predicted. Thus, in the (Best) results the value of I was given to the clustering algorithm and it simply found that number of clusters. In the new approach no such information is given, and the value of k is automatically discovered by the PK2 cluster stopping method. Thus, the new methods are in fact solving a somewhat more difficult problem, and are doing so with more success than the previous best methods.

For the 19 names shown in Table 2, the previous Best methods from 2005 and 2006 remain the most accurate for only six names. Only in the case of *Robek* did the automatic cluster stopping with PK2 go badly wrong. That may be due to the fact that *Ronaldo* was more ambiguous than originally anticipated (referring

to many more people than simply the well known soccer player who goes by that one name). In all other cases the automatic cluster stopping performed quite well, and predicted the number of identities exactly correctly or was off by at most one identity. This is a significant improvement over the previous results, and makes the methods much easier to apply on a wider range of data.

Of the 13 names that were most accurately discriminated by the very wide bigrams or co–occurrences, eight were best handled by the bigrams, three by the co–occurrences, and there were two ties between the bigrams and co–occurrences. While this doesn't provide overwhelming evidence that bigrams are always superior, it is an intriguing result that the order of the words matters even with very wide bigrams.

While we did not employ SVD in these experiments, it is clear from the *Ted Pedersen* results that there remain some cases where dimensionality reduction will be helpful. One of the main goals of our future work is to be able to automatically identify situations where SVD should or should not be applied.

SenseClusters also provides support for Latent Semantic Analysis [7], where we represent words in a context relative to the contexts in which they occur (rather than relative to the other words with which they occur, as is the case in this paper). In fact we included the LSA mode in our development experiments, and in general it did not fare well. However, we believe that there are still possible ways to formulate LSA for that name discrimination, as was shown by Levin, et al. [8].

Finally, our experiments have thus far been limited to small numbers of underlying identities. We plan to experiment with data like the John Smith data, where there are many individuals sharing that name.

8 Conclusions

We find that using very wide bigrams improves the results of unsupervised name discrimination, and that automatic cluster stopping can be employed to accurately identify the number of underlying identities.

Acknowledgments

All of the experiments in this paper were carried out with version 1.01 of the SenseClusters package, freely available from http://senseclusters.sourceforge.net. All of the data used in this paper is freely available from the author.

References

1. Artiles, J., Gonzalo, J., Sekine, S.: The SemEval-2007 WePS evaluation: Establishing a benchmark for the web people search task. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval 2007), Prague, Czech Republic, June 2007, pp. 64–69. Association for Computational Linguistics (2007)
2. Bagga, A., Baldwin, B.: Entity–based cross–document co–referencing using the vector space model. In: Proceedings of the 17th International Conference on Computational Linguistics, pp. 79–85 (1998)

3. Gale, W., Church, K., Yarowsky, D.: One sense per discourse. In: Proceedings of the Fourth DARPA Speech and Natural Language Workshop (1992)
4. Gooi, C.H., Allan, J.: Cross-document coreference on a large scale corpus. In: HLT-NAACL 2004: Main Proceedings, Boston, Massachusetts, USA, May 2 - 7, pp. 9–16 (2004)
5. Kulkarni, A.: Unsupervised context discrimination and automatic cluster stopping. Master's thesis, University of Minnesota (July 2006)
6. Kulkarni, A., Pedersen, T.: Name discrimination and email clustering using unsupervised clustering and labeling of similar contexts. In: Proceedings of the Second Indian International Conference on Artificial Intelligence, Pune, India, December 2005, pp. 703–722 (2005)
7. Landauer, T., Dumais, S.: A solution to Plato's problem: The Latent Semantic Analysis theory of acquisition, induction and representation of knowledge. Psychological Review 104, 211–240 (1997)
8. Levin, E., Sharifi, M., Ball, J.: Evaluation of utility of LSA for word sense discrimination. In: Proceedings of the Human Language Technology Conference of the NAACL, New York City, June 2006, pp. 77–80 (2006)
9. Pedersen, T., Kayaalp, M., Bruce, R.: Significant lexical relationships. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, Portland, OR, August 1996, pp. 455–460 (1996)
10. Pedersen, T., Kulkarni, A.: Selecting the right number of senses based on clustering criterion functions. In: Proceedings of the Posters and Demo Program of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, April 2006, pp. 111–114 (2006)
11. Pedersen, T., Kulkarni, A.: Discovering identities in web contexts with unsupervised clustering. In: Proceedings of the IJCAI 2007 Workshop on Analytics for Noisy Unstructed Text Data, Hyderabad, India, January 2007, pp. 23–30 (2007)
12. Pedersen, T., Kulkarni, A.: Unsupervised discrimination of person names in web contexts. In: Proceedings of the Eighth International Conference on Intelligent Text Processing and Computational Linguistics, February 2007, pp. 299–310 (2007)
13. Pedersen, T., Kulkarni, A., Angheluta, R., Kozareva, Z., Solorio, T.: 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)
14. Pedersen, T., Purandare, A., Kulkarni, A.: Name discrimination by clustering similar contexts. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 226–237. Springer, Heidelberg (2005)

Enriching Statistical Translation Models Using a Domain-Independent Multilingual Lexical Knowledge Base

Miguel García, Jesús Giménez, and Lluís Màrquez

TALP Research Center, LSI Department
Universitat Politècnica de Catalunya
Jordi Girona Salgado 1–3, E-08034, Barcelona
{tgarcia,jgimenez,lluism}@lsi.upc.edu

Abstract. This paper presents a method for improving phrase-based Statistical Machine Translation systems by enriching the original translation model with information derived from a multilingual lexical knowledge base. The method proposed exploits the Multilingual Central Repository (a group of linked WordNets from different languages), as a domain-independent knowledge database, to provide translation models with new possible translations for a large set of lexical tokens. Translation probabilities for these tokens are estimated using a set of simple heuristics based on WordNet topology and local context. During decoding, these probabilities are softly integrated so they can interact with other statistical models. We have applied this type of domain-independent translation modeling to several translation tasks obtaining a moderate but significant improvement in translation quality consistently according to a number of standard automatic evaluation metrics. This improvement is especially remarkable when we move to a very different domain, such as the translation of Biblical texts.

1 Introduction

One of the main criticisms against empirical methods in general, and Statistical Machine Translation (SMT) in particular, is their strong domain dependence. Since parameters are estimated from a corpus in a specific domain, the performance of the system on a different domain is often much worse. This flaw of statistical and machine learning approaches is well known and has been largely described in the NLP literature, for a variety of tasks, e.g., parsing [1], word sense disambiguation [2], and semantic role labeling [3].

In the case of SMT, domain dependence has very negative effects in translation quality. For instance, in the 2007 edition of the ACL MT workshop (WMT07), an extensive comparative study between in-domain and out-of-domain performance of MT systems built for several European languages was conducted [4]. Results showed a significant difference in MT quality between the two domains for all statistical systems, consistently according to a number of automatic evaluation metrics. In contrast, the differences reported in the case of rule-based or hybrid MT systems were less significant or inexistent, and even in some cases the performance of such systems out of the domain was higher than in the corpus domain. The reason is that, while these systems are often built on the assumption of an open or general domain, SMT systems are heavily

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 306–317, 2009.

specialized on the training domain. A change in domain implies a significant shift in the sublanguage (i.e., lexical choice and lexical order) employed, and, consequently, statistical models suffer a significant lack both of recall —due to unseen events— and precision —because event probability distributions differ substantially. Notice that we intentionally talk about *events* instead of words or phrases. In this manner, we have intended to emphasize that the decrease is not only due to unknown vocabulary, but also to other types of linguistic events, such as syntactic or semantic structures, either unseen or seen in different contexts. In other words, domain-dependence is not only a problem related to lexical selection, but also to other aspects such as syntactic ordering and semantic interpretations.

Therefore, statistical systems require an adaptation process before being ported to a new domain. This issue has been recently studied in [5]. Authors suggested building new specialized language and translation models built on small or middle-size monolingual and multilingual corpora belonging to the new target domain, or to a similar domain. These models are intended to contribute mainly with precision, by providing more accurate translations for frequent events. However, since they are estimated on a small amount of data, in principle, they offer a low recall. Besides, this approximation is based on the availability of corpora for the new domain, which is not always the case.

As a complementary alternative, we suggest using domain-independent knowledge sources which allow for increasing both the system recall and precision. Specifically, we have worked on translation modeling. We present a method to improve a phrase-based SMT system using an external multilingual lexical knowledge base. We exploit the Multilingual Central Repository (MCR) [6], a lexical database which aligns WordNets [7] among several European languages. This resource has been used by other authors. For instance, in [8] it is suggested using the MCR basically as a multilingual dictionary, i.e., to provide alternative word candidate translations for a phrase-based SMT system. They defined a very simple heuristic, based on the number of senses of each word and their distinct lexicalizations, so as to accompany each possible translation with a certain probability. A moderate improvement was reported, mainly due to the translation of unknown words, thus confirming that domain-independent models are a valid means of increasing the system recall. However, they did not take advantage of the semantic organization of the MCR so as to provide more precise translation probabilities. Besides, the interaction between their domain-independent translation models and domain-dependent ones was approached in a hard manner, i.e., by forcing the system to choose among the set of translation candidates provided by one model or the other. In this work, we have intended to additionally exploit the WordNet topology (i.e., concepts and relationships among them) to define a series of heuristics which allow for computing more informed translation probabilities. These heuristics take into account also the context surrounding each word form in the source sentence. Translation probabilities are softly integrated into the SMT system by allowing them to cooperate with other probability models. Our approach is deeply described in Section 2.

In Section 3, we discuss experimental results on the out-of-domain Spanish-to-English translation task of the WMT07 workshop. Our method yields a significantly improved translation quality according to a wide number of well-known standard automatic evaluation metrics. We have verified that the gain is mainly related to unknown

words for which the domain-dependent translation model did not have any translation. The system improves also for the case of known words also consistently to all metrics, but with less significance. Finally we test our method on a system that we built from a Biblical corpus. In this case the improvement in the quality of translation is even more important. Therefore, we believe that the suggested technique may result specially appropriate for porting an empirical MT system to a new domain, or so as to improve a system trained on a small-size corpus.

2 Exploiting the MCR

The MCR is not just a multilingual dictionary. Based on WordNet, the MCR is indeed a multilingual lexical database in which word forms are grouped in synsets (i.e, sets of synonyms), and interrelated through different types of additional semantic relations (antonymy, hyponimy, and meronymy, etc.). Besides, the MCR contains ontological information which allow us to associate each concept to a domain according to a given ontology.

In this work, we have exploited the MCR topology to provide alternative translation probabilities, $P(t|s)$, for each possible translation t of each content word s in the source sentence. We inspect every possible lexicalization of each of the senses of s according to the MCR and score the according to several heuristics:

- **Analysis of Domains:** This heuristic collects topical information by inspecting the ontological domains associated to each word in the source sentence, other than s. Formally, let us define a function $Dcount(s,t)$ which counts the number of source content words which may belong to any of the possible domains associated to t. A translation pair is scored as:

$$h_1(t|s) = \frac{Dcount(s,t)}{\sum_t Dcount(s,t)} \tag{1}$$

  We have used the WordNet domains, which consists of a list of 163 general domains such as 'administration', 'art', 'biology', etc.
- **Source Gloss Analysis:** This heuristic analyzes the similarity between the source sentence and the gloss associated to each sense of s, by comparing content words in them. Formally, let us define a function $SGcount(s,t)$ which counts the number of source content words which are also found in any of the source sense glosses associated to t. A translation pair is scored as:

$$h_2(t|s) = \frac{SGcount(s,t)}{\sum_t SGcount(s,t)} \tag{2}$$

- **Target Gloss Analysis:** This heuristic is similar to the previous one except that in this case, instead of directly source content words against source sense glosses, we compare possible content word translations against target sense glosses. Formally, let us define a function $TGcount(s,t)$ which counts the number of source content

word translations which are also found in any of the target sense glosses associated to t. A translation pair is scored as:

$$h_3(t|s) = \frac{TGcount(s,t)}{\sum_t TGcount(s,t)} \quad (3)$$

– **Analysis of Relations:** This heuristic analyzes relations between the words in source sentence and the translation t. We count how many content words in the source sentence are directly related to each possible translation t of each sense of s. Formally, let us define a function $Rcount(s,t)$ which counts the number of direct relations[1] between all possible senses of all content words in the source sentence and all possible senses of t. A translation pair is scored as:

$$h_4(t|s) = \frac{Rcount(s,t)}{\sum_t Rcount(s,t)} \quad (4)$$

– **Word Senses:** This heuristic is identical to that suggested in [8]. Translation pairs are scored by relative frequency according to the number of senses that lexicalize in the same manner. Formally, let us define a function $Scount(s,t)$ which counts the number of senses of s which may lexicalize as t in the target language. A translation pair is scored as:

$$h_5(t|s) = \frac{Scount(s,t)}{\sum_t Scount(s,t)} \quad (5)$$

Table 1. An example on the behavior of MCR-based heuristics

"Juan está sentado en el banco del parque ." vs. *"Juan fue al banco con un cheque ."*

	h_1	h_2	h_3	h_4	h_5	H	h_1	h_2	h_3	h_4	h_5	H
shoal	0	0	0	0	1	0.025	0	0	0	0	1	0.0125
banking concern	0	0	0	0	1	0.025	1	1	2	1	1	**0.1438**
shoals	0	0	0	0	1	0.025	0	0	0	0	1	0.0125
bank building	0	0	0	0	1	0.025	1	0	0	0	1	0.0438
sandbank	0	0	0	0	1	0.025	0	0	0	0	1	0.0125
banking company	0	0	0	0	1	0.025	1	1	2	1	1	**0.1438**
bank	0	0	0	0	4	0.1	2	1	2	1	4	**0.2125**
banks	0	0	0	0	4	0.1	2	1	2	1	4	**0.2125**
schools	0	0	0	0	1	0.025	0	0	0	0	1	0.0125
benches	0	0	4	0	1	**0.275**	0	0	0	0	1	0.0125
bench	0	0	4	0	1	**0.275**	0	0	0	0	1	0.0125
school	0	0	0	0	1	0.025	0	0	0	0	1	0.0125
financial institution	0	0	0	0	1	0.025	1	1	2	1	1	**0.1438**
sandbanks	0	0	0	0	1	0.025	0	0	0	0	1	0.0125
	0	0	8	0	20		8	5	10	5	20	

[1] Currently, we rely on relations such as hyponimy, meronymy, and other specific relations defined in the MCR.

Scores conferred by these heuristics are uniformly combined into a single score:

$$H(t|s) = \frac{\sum_j h_j(s|t)}{j} \tag{6}$$

As an illustration of the behavior of these heuristics, Table 1 shows the case of the Spanish word *'banco'*, in two different sentences. Scores at the left correspond to the sentence *"Juan está sentado en el banco del parque."* (which could be translated into English as *"Juan is sitting on a bench at the park"*), whereas scores at the right correspond to the sentence *"Juan fue al banco con un cheque."* (*"Juan went to the bank with a check"*). Note that individual heuristic scores have been replaced by the counts conferred by theit respective count function. It can be observed in the first sentence *'bench'* and *'benches'* are clearly preferred, mainly due to heuristic h_3 (target gloss analysis), whereas in the second sentence highest scores are attained by *'bank'* and *'banks'* with a wide consensous among heuristics.

3 Experimental Work

3.1 Baseline System

SMT systems address the translation task as a search process over a space determined by several probability models whose parameters are automatically derived from the analysis of large amounts of bilingual text corpora [9]. Therefore, their construction involves setting up three main components: (i) translation model(s), (ii) language model(s), (iii) and a search algorithm.

Our baseline system implements a typical phrase-based SMT architecture in which the different models are combined in a log-linear fashion [10]. For translation modeling, we have followed the approach described in [11] in which phrase pairs are automatically induced from word alignments. We use the *GIZA++ SMT Toolkit*[2] to generate word alignments [12]. Phrase extraction is performed following the *phrase-extract* algorithm described in [13]. This algorithm takes as input a word alignment matrix and outputs a set of phrase pairs that is *consistent* with it. A phrase pair is said to be consistent with the word alignment if all the words within the source phrase are only aligned to words within the target phrase, and viceversa. We work with the union of source-to-target and target-to-source alignments, with no heuristic refinement. Phrase pairs are scored on the basis of unsmoothed relative frequency, i.e., Maximum Likelihood Estimation. No smoothing is performed.

For language modeling, we have utilized the *SRI Language Modeling Toolkit*[3] [14]. We build trigram language models applying linear interpolation and Kneser-Ney discounting for smoothing.

Regarding the argmax search, we have used the *Pharaoh*[4] beam search decoder [15], which naturally fits with the previous tools.

[2] http://www.fjoch.com/GIZA++.html

[3] http://www.speech.sri.com/projects/srilm/download.html

[4] http://www.isi.edu/licensed-sw/pharaoh/

Table 2. Hard integration via XML markup

"Juan está sentado en el banco del parque ."
juan está <adj english="seated|sitting" prob="0.5|0.5">sentado</adj> en el <noun english="bench|benches|banks|bank|banking company|bank building|sandbank|shoals|banking concern|shoal|school|schools|depository financial institution|sandbanks" prob="0.275|0.275|0.1|0.1|0.025|0.025|0.025|0.025|0.025|0.025|0.025|0.025|0.025|0.025">banco</noun> del <noun english="pens|playpen|commons|green|parks|park|playpens|pen|commonses|common" prob="0.1|0.1|0.1|0.1|0.1|0.1|0.1|0.1|0.1|0.1">parque</noun> .
"Juan fue al banco con un cheque ."
juan fue al <noun english="banks|bank|banking company|banking concern|bank building | shoal|sandbank|shoals|depository financial institution|sandbanks|school|bench|schools| benches" prob="0.2125|0.2125|0.1438|0.1438|0.0438|0.0125|0.0125|0.0125|0.1438|0.0125| 0.0125|0.0125|0.0125|0.0125">banco</noun> con un <noun english="check|cheques|cheque| bank check|checks" prob="0.2|0.2|0.2|0.2|0.2">cheque</noun> .

3.2 Integration of Domain-Independent Translation Models into the Statistical Framework

The *Pharaoh* decoder offers the possibility of providing it with outer knowledge, by annotating the input with alternative translation options via XML-markup. See, for instance, in Table 2, the XML enriched input for the examples discussed in Section 2. Hower, this is a *'hard'* type of integration in the sense that predicions by the different translation models are not allowed to interact. Besides, in our case, XML marked up predictions are word-based, and may, thus, easily break phrasal cohesion during translation. Therefore, it is curcial that the two models fully cooperate.

In order to allow for a softer cooperation, we have applied the imaginative technique described in [16] for introducing dedicated translation probabilities as a log-linear feature. This technique consists basically in indexing every single word in the input sentences, and modifying the translation tables accordingly. Thus, every distinct instance of every possible input phrase may be uniquely identified.

3.3 Accessing the MCR

Let us note that in the MCR, just like in WordNet, a synset may be accessed either by providing the numerical synset id, or through the lemma and part-of-speech (PoS) of any of its members[5]. Therefore, prior to ellaborating on translation probabilities, the source sentence must annotated with PoS and lemma information. For instance, the word *"comió"* is replaced by *"(comer, v)"*. Analagously, translation candidates must

[5] PoS $\in (n, v, a, r)$, 'n' for nouns, 'v' for verbs, 'a' for adjectives, and 'r' for adverbs.

Table 3. Numerical Description of the News Commentary corpus

	Spanish	English
#sentences	51,109	51,109
#words	1,399,482	1,220,804
#distinct_words	47,125	35,757

Table 4. Automatic evaluation based on BLEU scores

	Baseline	MCR-XML	MCR-SOFT	MCR-SOFT$_u$	MCR-SOFT$_=$
nc-test2007	0.2495	0.2500	**0.2534**	0.2534	0.2525
test2006	0.1495	0.1494	**0.1521**	0.1518	0.1517
bible-test	0.1778	0.1778	**0.1872**	0.1866	0.1841

be transformed from lemmas into word forms. For instance, *"(comió, v)"*, which may translate into *"(eat, v)"*, would thus generate scores for *"ate", "eat", "eated", "eaten", "eating", and "eats"*.

We have used the *SVMTool*[6] for PoS-tagging [17], and the *Freeling*[7] suite of language analyzers [18] for lemmatization.

3.4 Results Training on the News Commentary Corpus

First, we have built a Spanish-to-English phrase-based system using the News Commentary (NC) corpus available for the *"ACL 2007 Second Workshop on Statistical Machine Translation (WMT07)"* [4]. A brief numerical description of this corpus may be found in Table 3. As test data, we have used the data sets provided by the WMT'07 organizers. This involves *'in-domain'* data from NC: nc-dev2007 (1,057 sentences) and nc-test2007 (2,007 sentences), and *'out-of-domain'* data from the Euparl corpus of European Parliament Proceedings [19]: dev2006 (2,000 sentences) and test2006 (2,000 sentences). Besides, because Euparl and NC corpora belong to a very similar domain (mostly political), we have also used two out-of-domain data sets extracted from a very different scenario, namely, translations of the Bible: bible-dev and bible-test each of 1,023 sentences.

Translation results have been automatically evaluated. We have used the IQ$_{MT}$ package [20][8], which includes a number of evaluation metrics such as BLEU [21], NIST [22], WER [23], PER [24], GTM [25], ROUGE [26], METEOR [27], and TER [28]. However, for the sake of readibility, and because we have observed a similar behavior for all metrics, results are discussed in terms of BLEU.

Table 4 shows translation quality results according to the BLEU metric. Results on a wider set of standard evaluation metrics, all based on lexical similarity, are reported in Table 10.

[6] http://www.lsi.upc.es/~nlp/SVMTool/

[7] http://www.lsi.upc.es/~nlp/freeling/

[8] The IQ$_{MT}$ software is available at http://www.lsi.upc.edu/~nlp/IQMT.

Table 5. MCR-based enrichment

	#words	#words in MCR	#transl. candidates
nc-test2007	56,578	24,501	412,649
test2006	62,354	26,347	455,709
bible-test	24,819	8,647	213,580

Table 6. MCR-based enrichment (unknown words only)

	#words	#words in MCR	#transl. candidates
nc-test2007	3,128	2,071	41,647
test2006	4,492	3,155	60,816
bible-test	3,567	2,514	54,042

In a first experiment, we try the integration via XML markup as described in Section 3.2. Table 5 presents a brief numerical description of the enrichment process for all test sets, including the total number of words, the number of words available inside the MCR, and the number of translation candidates found for these words.

The *'baseline'* system (Table 4, column 1) is compared to the *'MCR-XML'* system (column 2) enriched with MCR-based predictions, via XML markup, for all source words available in the English WordNet. No improvement was attained. The reason is, as described in Section 3.2, in the hard type of interaction. In order to overcome this problem, in a second experiment, we have applied the soft integration strategy based on word indexing described in Section 3.2 which allows MCR-based predictions to softly interact with those estimated from the NC corpus (*'MCR-SOFT'*). Translation quality results are shown in Table 4, column 3. In contrast to the XML markup case, this strategy yields a significant improvement, specially in the case of the 'bible-test' set, whereas for the 'test2006' and 'nc-test2007' sets it is much more moderate.

As expected, the MCR improves the system recall by providing translation for unknown words. However, it remains pending to test whether MCR predictions are also able to contribute to system precision. With that intent we study the case of providing MCR predictions only for unknown words (*'MCR-SOFT$_u$'*). Table 6 presents a brief numerical description of the enrichment process for all test sets, including the total number of words, the number of words available inside the MCR, and the number of translation candidates found for these words.

Translation quality results are shown in Table 4, column 4. By comparing these results to results in column 3, it can be seen that the gains are mainly related to the case of unknown words.

Finally, in order to test the performance of the heuristics described in Section 2, we compare the system using these heuristics to a system in which MCR-predictions for a given source word are set equi-probable (*'MCR-SOFT$_=$'*). Translation quality results are shown in Table 4, column 5. Heuristics exhibit a very modest improvement over the equi-probable setting. Only the case of the 'bible-test' presents a significant improvement. This fact evinces the need for improving the set of heuristics presented.

Table 7. Numerical Description of the Biblical corpus

	Spanish	English
#sentences	38,886	38,886
#words	947,354	959,583
#distinct_words	26,064	15,118

Table 8. MCR-based enrichment (unknown words only)

	#words	#words in MCR	#transl. candidates
nc-test2007	14,098	11,959	152,114

Table 9. Automatic evaluation using a set of lexical metrics (Biblical system)

	nc-test2007									
	1-PER	**1-TER**	**1-WER**	**BLEU**	**GTM $e=1$**	**GTM $e=2$**	**MTR exact**	**MTR wnsyn**	**NIST**	**ROUGE W-1.2**
Baseline	0.3534	0.2546	0.2383	0.0987	0.4569	0.1759	0.3920	0.4330	4.0997	0.2206
MCR-SOFT	0.4019	0.2853	0.2630	0.1168	0.5038	0.1892	0.4303	0.5325	4.7549	0.2629
MCR-SOFT$_u$	0.3980	0.2823	0.2597	0.1139	0.4998	0.1874	0.4264	0.5316	4.6986	0.2621
MCR-SOFT$_=$	0.3816	0.2702	0.2514	0.1087	0.4831	0.1833	0.4133	0.5197	4.4651	0.2467

3.5 Results Training on the Biblical Corpus

We built another statistical system using a Biblical corpus[9]. Table 7 contains a brief description of this corpus. As we can see, this corpus is slightly smaller than the NC corpus, and the vocabulary size is around twice smaller.

In order to compare with the results in the previous section (using the soft integration strategy), we have used nc-test2007 as test data. First, we apply the enrichment process for every word of nc-test2007 available inside the MCR (*'MCR-SOFT'*). Next, we work with equi-probable predictions (*'MCR-SOFT$_=$'*). Finally, we apply the method only for unknown words (*'MCR-SOFT$_u$'*). Table 8 presents a brief numerical description of the enrichment process.

Translation quality results, according to a number of evaluation metrics, are reported in Table 9. These show a significant improvement in the quality of translation obtained and clearly reflect the benefits of our technique. For example, we can see that the Baseline system scores 9.87 in terms of BLEU, whereas the *'MCR-SOFT'* system attains a BLEU score of 11.68, which represents an 18% relative improvement. When we apply the enrichment process only for unknown words (*'MCR-SOFT$_u$'*), a 15% relative improvement is attained.

Table 9 also shows a marked difference between the BLEU score obtained for *'MCR-SOFT'* (11.68) and *'MCR-SOFT$_=$'* (10.87). This fact supports the applicability

[9] New World Translation of the Holy Scriptures of the Watch Tower Bible and Tract Society of New York, Inc.

Table 10. Automatic evaluation using a set of lexical metrics (NC system)

	1-PER	1-TER	1-WER	BLEU	GTM $e=1$	GTM $e=2$	MTR exact	MTR wnsyn	NIST	ROUGE W-1.2
nc-test2007										
Baseline	0.6055	0.4866	0.4512	0.2495	0.6471	0.2681	0.5445	0.5912	7.7222	0.3240
MCR-XML	0.6047	0.4855	0.4501	0.2500	0.6461	0.2680	0.5443	0.5918	7.7057	0.3242
MCR-SOFT	0.6102	0.4900	0.4541	0.2534	0.6519	0.2698	0.5493	0.6078	7.7997	0.3310
MCR-SOFT$_u$	0.6100	0.4900	0.4539	0.2534	0.6517	0.2697	0.5493	0.6076	7.7989	0.3308
MCR-SOFT$_=$	0.6085	0.4886	0.4529	0.2525	0.6502	0.2692	0.5480	0.6068	7.7670	0.3292
test2006										
Baseline	0.4676	0.3359	0.3008	0.1495	0.5181	0.1911	0.4211	0.4689	5.4966	0.2358
MCR-XML	0.4675	0.3359	0.3008	0.1494	0.5180	0.1910	0.4210	0.4689	5.4955	0.2358
MCR-SOFT	0.4735	0.3407	0.3048	0.1521	0.5243	0.1929	0.4262	0.4841	5.5911	0.2420
MCR-SOFT$_u$	0.4731	0.3402	0.3044	0.1518	0.5239	0.1928	0.4258	0.4833	5.5837	0.2418
MCR-SOFT$_=$	0.4724	0.3399	0.3042	0.1517	0.5231	0.1926	0.4252	0.4828	5.5720	0.2407
bible-test										
Baseline	0.5221	0.4481	0.4340	0.1778	0.5574	0.2289	0.4653	0.4992	5.5536	0.2652
MCR-XML	0.5212	0.4471	0.4328	0.1778	0.5566	0.2286	0.4650	0.4991	5.5472	0.2650
MCR-SOFT	0.5326	0.4543	0.4383	0.1872	0.5683	0.2334	0.4750	0.5460	5.7564	0.2838
MCR-SOFT$_u$	0.5318	0.4533	0.4375	0.1866	0.5675	0.2331	0.4743	0.5450	5.7443	0.2837
MCR-SOFT$_=$	0.5280	0.4506	0.4353	0.1841	0.5635	0.2316	0.4711	0.5416	5.6797	0.2764

of our set of heuristics based on the MCR topology to assign translation probabilities when moving to a new domain.

We also want to point out that the improvement obtained in *'MCR-SOFT'* (compared to the Baseline system) and the difference between *'MCR-SOFT'* and *'MCR-SOFT$_=$'* (in favour of the former) is much larger in the case of the Bible system (Table 9) than in the case of the NC system (Table 10). This fact evinces that this technique is especially useful to improve the translation on systems with strong domain dependence or so as to improve a system where the training corpus has a reduced vocabulary.

4 Conclusions

We have showed that the MCR may be successfully applied to the MT scenario. The technique presented proves specially effective as a source of recall, by providing translations for unknown words. This might be of special interest when porting SMT systems to a new domain or in cases where the training corpus has a reduced vocabulary. However, the current set of heuristics attains only a minor improvement over the naïve baseline of setting equi-probable predictions for all the translation candidates associated to a given source word. The set of heuristics must be deeply analyzed and refined.

As a possible alternative to the heuristic approach, we plan to exploit the information in the MCR so as to assist a Discriminative Phrase Selection engine [29], by providing additional features.

Acknowledgments

This research has been funded by the Spanish Ministry of Education and Science, project OpenMT (TIN2006-15307-C03-02). Our NLP group has been recognized as a Quality Research Group (2005 SGR-00130) by DURSI, the Research Department of the Catalan Government. Thanks also goes out to the Michoacán University of San Nicolás of Hidalgo (México) for all the support to Miguel García in his studies/researchs.

References

1. Sekine, S.: The Domain Dependence of Parsing. In: Proceedings of the Fifth Conference on Applied Natural Language Processing, pp. 96–102 (1997)
2. Escudero, G., Marquez, L., Rigau, G.: An Empirical Study of the Domain Dependence of Supervised Word Disambiguation Systems. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 172–180 (2000)
3. He, S., Gildea, D.: Self-training and Co-training for Semantic Role Labeling: Primary Report. Technical report, TR 891, Department of Computer Science, University of Rochester (2006)
4. Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., Schroeder, J.: (Meta-) Evaluation of Machine Translation. In: Proceedings of the ACL Workshop on Statistical Machine Translation, pp. 136–158 (2007)
5. Giménez, J., Màrquez, L.: Low-cost Enrichment of Spanish WordNet with Automatically Translated Glosses: Combining General and Specialized Models. In: Proceedings of COLING-ACL (2006)
6. Atserias, J., Villarejo, L., Rigau, G., Agirre, E., Carroll, J., Magnini, B., Vossen, P.: The MEANING Multilingual Central Repository. In: Proceedings of the 2nd Global WordNet Conference (GWC) (2004)
7. Fellbaum, C. (ed.): WordNet. An Electronic Lexical Database. The MIT Press, Cambridge (1998)
8. Giménez, J., Màrquez, L., Rigau, G.: Automatic Translation of WordNet Glosses. In: Proceedings of Cross-Language Knowledge Induction Workshop, EUROLAN Summer School (2005)
9. Brown, P.F., Cocke, J., Pietra, S.A.D., Pietra, V.J.D., Jelinek, F., Lafferty, J.D., Mercer, R.L., Roossin, P.S.: A statistical approach to machine translation. Computational Linguistics 16(2), 76–85 (1990)
10. Och, F.J., Ney, H.: Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In: Proceedings of the 40th ACL, pp. 295–302 (2002)
11. Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-Based Translation. In: Proceedings of the Joint Conference on Human Language Technology and the North American Chapter of the Association for Computational Linguistics (HLT-NAACL) (2003)
12. Och, F.J., Ney, H.: A Systematic Comparison of Various Statistical Alignment Models. Computational Linguistics 29(1), 19–51 (2003)
13. Och, F.J.: Statistical Machine Translation: From Single-Word Models to Alignment Templates. PhD thesis, RWTH Aachen, Germany (2002)
14. Stolcke, A.: SRILM - An Extensible Language Modeling Toolkit. In: Proceedings of ICSLP (2002)
15. Koehn, P.: Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In: Proceedings of AMTA (2004)

16. Giménez, J., Màrquez, L.: Context-aware Discriminative Phrase Selection for Statistical Machine Translation. In: Proceedings of the ACL Workshop on Statistical Machine Translation, pp. 159–166 (2007)
17. Giménez, J., Màrquez, L.: SVMTool: A general POS tagger generator based on Support Vector Machines. In: Proceedings of 4th LREC, pp. 43–46 (2004)
18. Carreras, X., Chao, I., Padró, L., Padró, M.: FreeLing: An Open-Source Suite of Language Analyzers. In: Proceedings of the 4th LREC, pp. 239–242 (2004)
19. Koehn, P.: Europarl: A Multilingual Corpus for Evaluation of Machine Translation. Technical report (2003), http://people.csail.mit.edu/people/koehn/publications/europarl/
20. Giménez, J., Amigó, E.: IQMT: A Framework for Automatic Machine Translation Evaluation. In: Proceedings of the 5th LREC, pp. 685–690 (2006)
21. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation, rc22176. Technical report, IBM T.J. Watson Research Center (2001)
22. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: Proceedings of the 2nd Internation Conference on Human Language Technology, pp. 138–145 (2002)
23. Nießen, S., Och, F.J., Leusch, G., Ney, H.: An Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. In: Proceedings of the 2nd LREC (2000)
24. Tillmann, C., Vogel, S., Ney, H., Zubiaga, A., Sawaf, H.: Accelerated DP based Search for Statistical Translation. In: Proceedings of European Conference on Speech Communication and Technology (1997)
25. Melamed, I.D., Green, R., Turian, J.P.: Precision and Recall of Machine Translation. In: Proceedings of the Joint Conference on Human Language Technology and the North American Chapter of the Association for Computational Linguistics (HLT-NAACL) (2003)
26. Lin, C.Y., Och, F.J.: Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statics. In: Proceedings of the 42nd ACL (2004)
27. Banerjee, S., Lavie, A.: METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In: Proceedings of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization (2005)
28. Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J.: A Study of Translation Edit Rate with Targeted Human Annotation. In: Proceedings of AMTA, pp. 223–231 (2006)
29. Giménez, J., Màrquez, L.: Discriminative Phrase Selection for Statistical Machine Translation. In: Learning Machine Translation. NIPS Workshop. MIT Press, Cambridge (2008)

Exploiting Parallel Treebanks to Improve Phrase-Based Statistical Machine Translation

John Tinsley, Mary Hearne, and Andy Way

National Centre for Language Technology
Dublin City University, Ireland
{jtinsley,mhearne,away}@computing.dcu.ie

Abstract. Given much recent discussion and the shift in focus of the field, it is becoming apparent that the incorporation of syntax is the way forward for the current state-of-the-art in machine translation (MT). Parallel treebanks are a relatively recent innovation and appear to be ideal candidates for MT training material. However, until recently there has been no other means to build them than by hand. In this paper, we describe how we make use of new tools to automatically build a large parallel treebank and extract a set of linguistically motivated phrase pairs from it. We show that adding these phrase pairs to the translation model of a baseline phrase-based statistical MT (PBSMT) system leads to significant improvements in translation quality. We describe further experiments on incorporating parallel treebank information into PBSMT, such as word alignments. We investigate the conditions under which the incorporation of parallel treebank data performs optimally. Finally, we discuss the potential of parallel treebanks in other paradigms of MT.

1 Introduction

The majority of research in recent years in machine translation (MT) has centred around the phrase-based statistical approach. This paradigm involves translating by training models which make use of sequences of words, so-called phrase pairs, as the core translation model of the system [1]. These phrase pairs are extracted from aligned sentence pairs using heuristics over a statistical word alignment. While phrase-based models have achieved state-of-the-art translation quality, evidence suggests there is a limit as to what can be accomplished using only simple phrases, for example, satisfactory capturing of context-sensitive reordering phenomena between language pairs [2]. This assertion has been acknowledged within the field as illustrated by the recent shift in focus towards more linguistically motivated models.

Aside from the development of fully syntax-based models of MT, [3,4,5,6] to list a few, there have been many extensions and improvements to the phrase-based model which have endeavoured to incorporate linguistic information into the translation process. Examples of these can be seen in the work of [7] and [8] who make use of syntactic supertags and morphological information respectively. [9,10] describes a phrase-based model which makes use of generalised templates while [11] exploit semantic information in the form of phrase-sense disambiguation. All of these approaches have a common starting point: the set of phrase pairs initially extracted in the phrase-based model.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 318–331, 2009.

Given this, we raise two questions: 1) would translation quality improve in a baseline phrase-based system if the translation model included linguistically motivated, constituent-based phrase pairs? and 2) would subsequent extensions to the phrase-based model, such as those outlined above, improve even further if they were implemented on a base of linguistically motivated phrase pairs? In this paper we will address the first question, with the second question being discussed in terms of future work.

We have shown previously that, on a small scale, incorporating linguistically motivated phrase pairs extracted from parallel treebanks can improve phrase-based statistical MT (PBSMT) systems [12]. We further examine this hypothesis by scaling up the experiments of [12] by approximately 2 orders of magnitude. We then carry out a detailed series of experiments to determine how to optimally use parallel treebank phrase pairs within the phrase-based model. In addition to this, we investigate some alternative ways of incorporating the information encoded in parallel treebanks, such as word alignments, into the translation process of a PBSMT system.

The remainder of this paper is outlined as follows: Section 2 gives some background on SMT phrase extraction and parallel treebanks. Section 3 describes the data used in all experiments in this paper. Section 4 details the experiments carried out along with results, analysis and discussion. Finally, we conclude and present some avenues for future work in Section 5.

2 Background

At the core of any phrase-based SMT system lies a table of translationally equivalent phrase pairs. These phrase pairs are extracted from parallel corpora, on a sentence pair by sentence pair basis, using heuristics which operate on a set of high-recall word alignments between the sentence pairs. The phrase pairs are then scored in a log linear model combining a number of different features. It was shown by [1] that restricting the set of extracted phrase pairs to those which correspond to syntactic constituents in a context-free phrase-structure tree harms translation accuracy. We carried out experiments previously [12], whereby rather than *restrict* the set of phrase pairs to those corresponding to constituents, we *supplement* the phrase-based translation model with all linked constituent pairs in a syntactically annotated version of the same parallel data used to train the PBSMT system. This led to improved accuracy across four translation tasks. The results of these experiments are summarised in Table 1.

The acquisition of such syntactically annotated parallel resources, so-called parallel treebanks, has been the topic of much recent research [13,14,15]. A parallel treebank comprises syntactically parsed aligned sentences in two or more languages. In addition to this, sentences are aligned below the level of the clause [16], i.e. there are alignments between nodes in the tree pairs, which indicate translational equivalence between the

Table 1. Summary of translation results reported in [12] in terms of Bleu score

Config.	en-es	es-en	en-de	de-en
Baseline	0.1765	0.1754	0.1186	0.1622
+Tree	**0.1867**	**0.1880**	**0.1259**	**0.1687**

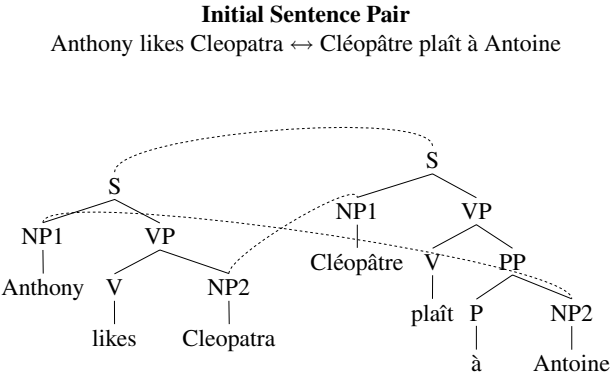


Fig. 1. An example English–French parallel treebank entry for the given sentence pair

surface strings dominated by the linked node pairs. An example parallel treebank entry is shown in Figure 1.

Until relatively recently parallel treebank acquisition was a manual task. It is a time-consuming, error-prone process which requires linguistic expertise in both the source and target languages. This makes it an impractical task on a large scale, such as the scale on which we may need to work in MT. For these reasons, parallel treebanks are thin on the ground and those that are available are relatively small [17,18]. However, recent advances in technology, such as improvements in monolingual parsing and the development of subtree alignment tools, such as those described in the work referred to earlier in this section, have paved the way for the automatic creation of large high-quality parallel treebanks. In the following section we detail the construction of the parallel treebank used in our experiments.

3 Parallel Data

The principal resource used for the experiments described in this paper is the English–Spanish section of the Europarl corpus. After cleaning, which involved the removal of blank lines, erroneous alignments and sentences over 100 tokens in length, there were 729,891 aligned sentence pairs remaining. The process of building a parallel treebank from this parallel corpus was completely automated. Firstly, each monolingual corpus was parsed using freely available phrase-structure parsers. For the English corpus we used the Berkeley parser [19]. The Spanish corpus was parsed using Bikel's parser [20] trained on the Cast3LB Spanish treebank [21].

The final step in the annotation process was to automatically align the newly parsed parallel corpus at sub-sentential level. This is done by inserting links between constituent node pairs in the tree which imply translational equivalence between the surface strings dominated by the linked node pairs. Tree alignment is a precision-based task – the goal is not to aggressively align as many nodes as possible in the tree. To leave a node unaligned is not to say it has no translational equivalent. Instead, translational equivalences for unaligned nodes are encapsulated in wider contexts by links higher up

in the tree pair. For example, looking back to the tree pair in Figure 1, although there is no direct link from the source tree V node, dominating *likes*, to the target tree, does not mean it has no translation in this sentence pair. Instead, its translational equivalence to the non-constituent *plaît à* is captured implicitly by the links between the S nodes and the NP nodes. To insert these links between the parallel tree pairs we used our own subtree alignment algorithm [22]. This algorithm automatically induces links between nodes (at both word- and phrase-level) in a tree pair by exploiting statistical word alignment probabilities estimated over the sentence pairs of the tree pairs to be aligned.

Given the parallel treebank is built automatically, the issue of its quality arises. Of course, there are parse errors and misalignments to be found, but we are satisfied that the quality is high enough to demonstrate our hypothesis. The papers describing the two parsers we use both report high accuracy: 90.05% labelled f-score for English, and 83.96% labelled f-score for Spanish. The reported accuracy of the sub-tree alignment algorithm is also high. We refer the interested reader to the original alignment paper for a more detailed evaluation.

4 Experiments

This section reports on the various experiments we carried out in which we incorporate phrase pairs extracted from the parallel treebank into a phrase-based SMT system. We first describe how we use the parallel treebank phrase pairs directly in translation, in Section 4.1. We follow this up in Sections 4.2–4.5 by examining a number of different approaches to incorporating the information encoded in the parallel treebank into the translation process.

For all translation experiments the setup included a development set of 1,000 sentence pairs, a test set of 2,000 sentence pairs,[1] all chosen at random, with the remaining 726,891 sentence pairs (and tree pairs where relevant) used for training. The baseline MT system was built using Moses [23]. For the phrase-extraction step of the training process, phrases pairs up to a maximum of 7 tokens in length were extracted using the *grow-diag-final* heuristic. 5-gram language modelling was carried out using the SRI language modelling toolkit [24]. System tuning was performed on the development set using minimum error-rate training as implemented in Moses. All translations were performed from English into Spanish and were automatically evaluated using the metrics BLEU [25], NIST [26] and METEOR [27]. Statistical significance was calculated using bootstrap resampling [28] (with p=0.05 unless otherwise stated).

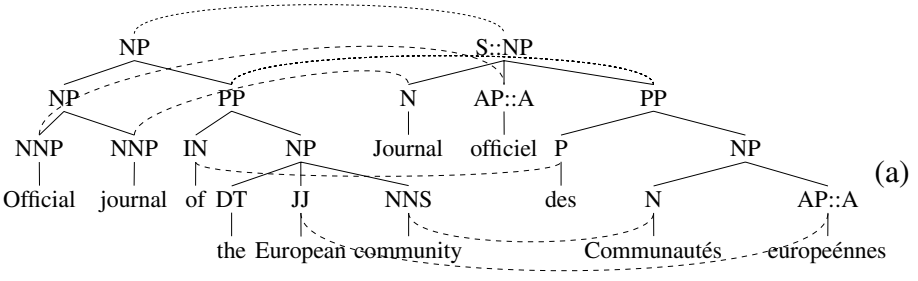
4.1 Combining Phrase Resources

The first question we want to answer is: can linguistically motivated phrase pairs extracted from our parallel treebank improve translation when incorporated into a baseline phrase-based SMT system? To find out we must first extract the set of phrase pairs from the parallel treebank. These phrases correspond to the yields of all linked constituent pairs in the treebank. We then add these phrase pairs to the translation model of the

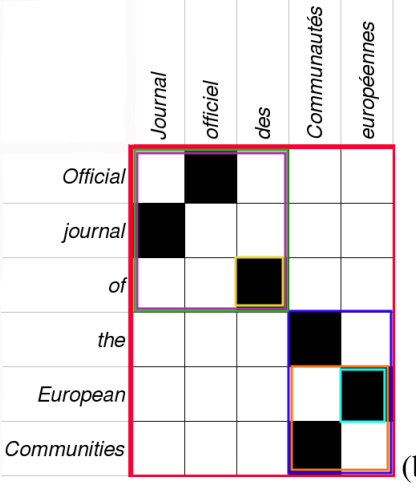
[1] Test sentences were restricted in length to between 5 and 30 tokens.

Training Sentence Pair

"Official journal of the European Community" ↔ "Journal officiel des Communautés européennes"



(a)



(b)

† Official journal ↔ Journal officiel
† Official journal of ↔ Journal officiel des
∗ Official journal of the/ European Communities ↔ Journal officiel des/ Communautés européennes
∗ of ↔ des
∗ of the European Communities ↔ des Communautés européennes
∗ the European Communities ↔ Communautés européennes
∗ European ↔ européennes
◇ Communities ↔ Communautés
◇ Official ↔ officiel
◇ journal ↔ Journal

(c)

Fig. 2. Example of phrase extraction for the given sentence pair depicting: (a) the aligned parallel tree pair; (b) the word alignment matrix (the rectangled areas represent extracted phrase pairs); (c) the combined set of extracted phrase pairs where: ◇ = only extracted from (a); † = only extracted from (b); ∗ = extracted from both (a) and (b)

baseline MT system and reestimate the phrase translation probabilities over the combined set of phrase pairs. We will illustrate this process with an example. In Figure 2 we see an example sentence pair from an English–French parallel corpus. Figure 2(a) illustrates the parallel treebank entry for this pair, while Figure 2(b) shows its statistical word alignment according to the PBSMT system. The combined set of extracted phrase pairs, to be added to the translation model, is given in Figure 2(c). We can see that while there is overlap between the two sets of phrase pairs, there are also a certain number of phrase pairs unique to the parallel treebank. Our hypothesis is that these unique constituent-based phrase pairs, along with the increase in probability mass given to those overlapping phrase pairs, will improve translation quality.

Table 2 shows the results of translation experiments using different combinations of data in the translation model.

Table 2. Evaluation of combinations of data in translation models. Baseline = PBSMT phrase pairs. Tree = phrase pairs from the parallel treebank.

Config.	BLEU	NIST	%METEOR
Baseline	0.3341	7.0765	57.39
+Tree	**0.3397**	**7.0891**	**57.82**
Tree only	0.3153	6.8187	55.98

We see that adding parallel treebank phrase pairs to the baseline model (+Tree) significantly improves translation accuracy (1.68% relative increase in BLEU score) across all metrics. We attribute this to the increase in coverage of the translation model given the new phrase pairs combined with the increased probability mass of the phrase pairs in common between the two sets. This effect is desireable as we would assume those phrase pairs extracted by both methods would be more reliable. Of the treebank phrase pair types added to the translation model, 77.5% of these were not extracted in the baseline system. These ultimately constituted 16.79% of the total phrases in the translation model. The remaining treebank phrase pairs which were also extracted in the baseline system comprised 4.87% of the total phrase pairs. The full figures are provided in Table 3.

Table 3. Frequency information regarding the numbers of phrase pairs extracted from the baseline system and from the parallel treebank. $\bigcap$ is the number of phrase pair types extracted by both methods.

Resource	#Tokens	#Types	$\bigcap$
Baseline	72,940,465	24,708,527	1,447,505
Treebank	21,123,732	6,432,771	

Using the data from the parallel treebank alone (Tree only) leads to a significant drop in translation accuracy (5.96% relative BLEU) compared the baseline. We attribute the drop to the insufficient translation coverage of this model. This was to be expected as we can maximally extract the number of phrase pairs from a tree pair as there are linked node pairs. This number will never approach the number necessary to achieve translation coverage competitive with that of the baseline system.

We carried out one further experiment where we added only strict phrase pairs[2] from the parallel treebank into the baseline phrase-based system. The motivation for this was the discovery that of all the data extracted from the parallel treebank, 20.3% were word alignments and 7.35% of these were alignments between function words and punctuation that occurred more than 1,000 times. By removing these high-risk alignments we reduce the potential for search errors while keeping the vast majority of useful translation units. The outcome of this experiment, presented in Table 4, was even further significant improvement (2.18% relative increase in BLEU score) across all metrics over the baseline phrase-based system than using all the parallel treebank data.

Table 4. Effect of using strictly phrase pairs from the parallel treebank

Config.	BLEU	NIST	%METEOR
Baseline	0.3341	7.0765	57.39
+Tree	0.3397	7.0891	57.82
Strict phrases	**0.3414**	**7.1283**	**57.98**

Given these findings, which corroborate our findings in [12], we now describe further experiments we carried out to investigate additional ways to exploit the information encoded in the parallel treebank to use with the PBSMT framework.

4.2 Weighting Treebank Data

In the previous section we showed that we can improve over the baseline PBSMT system by simply adding parallel treebank phrases to the translation model. Our next set of experiments investigate whether giving more weight to the syntactic phrase pairs in the translation model will further improve performance. The motivation here is that the syntactic phrase pairs may be more reliable, as we suggested in [12], and thus preferable for use in translation. To do this we built 3 translation models – of the form Baseline+Tree – in which we count the parallel treebank phrase pairs twice, three times and five times when estimating phrase translation probabilities. The results of these experiments are shown in Table 5.[3]

The findings here are slightly erratic. Doubling the presence of the parallel treebank phrase pairs (+Tree x2) leads to insignificant differences compared to the baseline

Table 5. Effect of increasing relative frequency of parallel treebank phrase pairs in the translation model

Config.	BLEU	NIST	%METEOR
Baseline+Tree	**0.3397**	**7.0891**	**57.82**
+Tree x2*	0.3386	7.0813	57.76
+Tree x3	0.3361	7.0584	57.56
+Tree x5*	0.3377	7.0829	57.71

[2] A strict phrase pair is an *m-to-n* alignment where both *m* and *n* are greater than 1.

[3] A * next to a particular configuration in the table indicates the results reported are statistically insignificant *compared to the baseline*. We assume this to be the case in all proceeding tables.

across all metrics, while counting them three times (+Tree x3) leads to a significant drop (p=0.02) in translation accuracy. Counting them five times (+Tree x5) again leads to insignificant differences.

Given the ineffectiveness of this crude method of weighting, we built a system using two distinct phrase tables, one containing the baseline phrase-based SMT phrases and the other containing the phrase pairs from the parallel treebank. This allows the tuning process to choose the optimal weights for the two phrase tables and the decoder can chose phrase pairs from either table as the model dictates. Table 6 shows the performance of this system relative to the Baseline+Tree configuration. Again, no improvement was found. We see a significant decrease in translation accuracy but it is not uniform across the metrics.

Table 6. Effect of using two separate phrase tables in the translation model

Config.	BLEU	NIST	%METEOR
Baseline+Tree	**0.3397**	**7.0891**	**57.82**
Two Tables	0.3365	7.0812*	57.50

We know from the experiments of Section 4.1 that adding parallel treebank data to the baseline phrase-based system can improve translation quality. However, simply increasing their frequency in the translation model has a detrimental effect on translation. This may be due to the fact that we are also increasing the influence of those treebank phrase pairs which are not as useful – such as those word alignments also mentioned in the previous section – and this is having a negative effect.

A potential way to proceed along these lines may be to find a more balanced compromise between the two sets of phrase pairs in the translation model, but for now we can conclude that when adding parallel treebank phrase pairs to the model, it is optimal to add them a single time into the baseline model.

4.3 Filtering Treebank Data

Phrase pairs extracted in the baseline system were restricted in length to 7 tokens as previous experiments have shown that phrases longer than this yield little improvement and are occasionally detrimental to translation quality [1]. In our previous experiments no such restriction was placed on the parallel treebank phrase pairs. To investigate whether longer treebank phrase pairs were harming translation quality, we built a translation model – Baseline+Tree – including parallel treebank phrase pairs up to a maximum of 7 tokens in length only. The filtered phrase table was 11.7% smaller than that which contained unrestricted phrase pairs. The effect of this filtering on translation performace is shown in Table 7 where we see statistically insignificant fluctuation across the metrics. This indicates that the longer phrases were inconsequential during decoding. Further analysis confirms this, with longer phrases rarely being used in the Baseline+Tree configuration, and only a small percentage (8%) of the sentences being translated differently when filtering them out. From this we can conclude that when adding treebank phrase pairs, we need only add in those phrase pairs of similar length to the ones in the baseline model.

Table 7. Effect of using filtering longer phrase pairs from the parallel treebank data

Config.	BLEU	NIST	%METEOR
Baseline+Tree	**0.3397**	7.0891	**57.82**
-Filtered*	0.3387	**7.0926**	57.67

4.4 Treebank-Driven Phrase Extraction

In this section we describe experiments in which we used the alignment information encoded in the parallel treebank to seed the phrase extraction heuristic in the PBSMT system.

One oft-cited reason for the inability of syntactic translation models to improve upon the state-of-the-art is that only using constituent-based phrase pairs is too restrictive [1,9]. Translation units such as the English–German pair *there is* ↔ *es gibt* will never be extracted as a constituent phrase pair despite being a perfectly acceptable translation pair. To attempt to overcome this problem, we sought some ways in which to use the linguistic information encoded in the parallel treebank to extract a set of non-constituent-based phrase pairs. By doing this we would have "linguistically informed" phrase pairs as opposed to purely constituent phrase pairs.

In order carry this out, we built a translation model by seeding Moses' phrase extraction heuristic with the word alignments from the parallel treebank. The motivation for this is that we have syntax-based word alignments in the parallel treebank guided by the non-lexical links higher up in the tree [22] and thus subsequent phrases extracted based on these would possibly have more of a linguistic foundation than those based on statistical word alignments, and be potentially more reliable.

We also built a translation model using the union of the Moses word alignments and the parallel treebank word alignments. Finally, we built two more translation models in which both of the models above were supplemented with the phrase pairs extracted from the parallel treebank, as this was the original hypothesis we were examining. The results of translation experiments using all of these models are presented in Table 8.

The first two rows in the table showing the results from Section 4.1 represent our baseline here. In the third row (TBX), we see that seeding the phrase extraction with the treebank alignments leads to a significant drop in translation performance compared

Table 8. Evaluation of translations using different word alignments to seed phrase extraction. TBX = extraction seeded by parallel treebank word alignments. UnionX = extraction seeded by union of parallel treebank and Moses word alignments.

Config.	BLEU	NIST	%METEOR
Baseline	0.3341	7.0765	57.39
+Tree	**0.3397**	**7.0891**	57.82
TBX	0.3102	6.6990	55.64
+Tree	0.3199	6.8517	5639
UnionX	0.3277	6.9587	56.79
+Tree	0.3384*	7.0508	**57.88**

to the baseline. Adding the treebank phrase pairs (+Tree) to this model significantly improves performance as we would expect given our previous findings, however, it still does not approach the performance of the baseline.

Seeding the phrase extraction using parallel treebank word alignments leads to an unwieldy amount of phrase pairs in the translation model – approximately 88.5 million (92.9 million when including treebank phrase pairs)– many of which are useless e.g. *framework for olaf, in order that* ↔ *marco*. This is due to the fact that the parallel treebank word alignments have quite low recall and thus the phrase extraction heuristic is free to extract a large number of phrases anchored by a single word alignment.[4] This tells us that the parallel treebank word alignments are too sparse to be used to seed the phrase extraction heuristics.

The intuition behind the next experiment – using the union of the parallel treebank and Moses word alignments to seed phrase extraction – was to simultaneously increase the recall of statistical word alignments and the precision of the parallel treebank word alignments and creating a more robust, reliable word alignment overall.

We see from the fifth row (UnionX) of Table 8 that using the union of alignments led to a small, but significant, drop in translation accuracy compared to the baseline. More interestingly we note that adding the parallel treebank phrase pairs to this model (UnionX+Tree) led to comparable performance to the baseline.[5] This is interesting as the baseline translation model including treebank phrases, Baseline+Tree, has approximately 29.7M entries. However, the UnionX+Tree translation model contains only 13.1M phrase pair entries. This constitutes a 56% decrease in translation model size without any significant decrease in translation accuracy. These figures, and those for the other models described in this section, are given in Table 9. This discovery is a very positive by-product of these experiments. We can conclude that using the union of statistical and treebank-based word alignments may be effective for producing smaller translation models without suffering a reduction in translation performance. We intend to investigate these findings in greater depth in the near future.

Table 9. Comparison of the phrase table size for each model. #Phrase = number of phrases extracted using a given word alignment. #Phrase+Tree = size of model when treebank phrases are included.

Word Alignment	#Phrases	#Phrases+Tree
Moses	24.7M	29.7M
Treebank	88.5M	92.89M
Union	7.5M	13.1M

4.5 Alternative Lexical Weighting

In this section we discuss experiments carried out in which we used the information encoded in the parallel treebank to calculate the values for the lexical weighting feature in the log-linear model.

[4] In the example *framework for olaf, in order that* ↔ *marco* the only word alignment was between *framework* and *marco*.

[5] Differences were either statistically insignificant or inconsistent across the evaluation metrics.

The translation model in a phrase-based SMT system, in addition to calculating a phrase translation probability, calculates a lexical weighting score for each phrase pair. This feature checks how well the words in the source and target phrases translate to one another by scoring each phrase pair according to its word alignment using the word translation table extracted during training.

In order to potentially improve these lexical weighting scores, we recalculate them according to the word alignments found in the parallel treebank, as opposed to the statistical word alignment. Firstly we reassign each phrase pair in the translation model (Baseline+Tree) a word alignment according to the parallel treebank word alignments. We then estimate a word translation distribution over the word alignments in the parallel treebank and use this to calculate new lexical weights for the phrase pairs in the translation model.

We then replicate this setup by assigning the phrase pairs new alignments according to the union of the statistical and parallel treebank word alignments – as we did in Section 4.4 – and scoring them from a word translation probability distribution over all the word alignments from both resources. The results of these experiments are given in Table 10.

Table 10. Effect of using linguistically motivated word alignments to calculate lexical weighting for phrase pairs in the translation model. TB_words = lexical weights according to treebank word alignments. Union_words = lexical weights according to union of treebank and Moses word alignments.

Config.	BLEU	NIST	%METEOR
Baseline+Tree	**0.3397**	**7.0891**	**57.82**
TB_words	0.3356	7.0355	57.32
Union_words	0.3355	7.0272	57.41

We see from these results that performance degrades slightly, but significantly, when using the new lexical weights and that the results are almost identical between the two new methods of scoring.[6]

The ineffectiveness of this approach can be attributed to the fact the the majority of the phrase-pairs, i.e. those extracted in Moses, were extracted according to the statistical word alignments and thus would have a high-recall word alignment. To replace these word alignments with the parallel treebank alignments, however precise, will give a much lower recall word alignment between the extracted phrase pairs. This, coupled with the fact that the lower recall word alignments give a less reliable word translation table, leads to poorer lexical weights and, ultimately, a decrease in translation quality.

5 Conclusions and Future Work

Augmenting the standard phrase-based model with linguistically motivated phrase pairs from a parallel treebank can improve translation quality. Some ongoing work we are

[6] This is not to say there was no difference between them. 18.5% of the sentences in the test set were translated differently.

carrying out along these lines involves investigating the effect of the treebank phrase pairs on translation performance as the size of the training set increases. Early results seem to indicate that increasing the training set leads to a decrease in the influence of the treebank phrases.

As per the second question we raised in Section 1, it would be interesting to investigate whether some of the approaches mentioned in the introduction, which improved over the standard model, would yield further improvement by building on the treebank-induced model described here.

In Section 4.2 we saw that simply increasing the relative frequency of the treebank phrases in the model did not help, nor did using separate phrase tables. But we believe there is still a better compromise to be found between all the phrase resources in the model. Section 4.3 indicated that filtering the phrase table has negligible effect on translation accuracy.

We saw in Sections 4.4 and 4.5 that variations on incorporating the parallel treebank data, specifically the word alignments, did not lead to any improvements. The phrase-based model is tailored to high-recall statistical word alignments and reductions in recall as seen here, regardless of the precision, do not lend themselves to improved translations. However, we also saw that we can induce much smaller translation models from the parallel treebank without a significant drop in MT performance.

Finally, what we have described here is only scratching the surface in terms of the exploitability of parallel treebanks in MT. We are currently working on the extraction of generalised translation templates and translation rules from parallel treebanks with a view to evaluating their performance in more syntax-aware models of MT, such as those of [4] and [6]. Such models are illustrative of the potential of this linguistically rich resource.

Acknowledgements

This work is supported by Science Foundation Ireland (grant no. 05/RF/CMS064) and the Irish Centre for High-End Computing[7].

References

1. Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-Based Translation. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Canada, pp. 48–54 (2003)
2. Zollmann, A., Venugopal, A., Och, F., Ponte, J.: A Systematic Comparison of Phrase-Based, Hierarchical and Syntax-Augmented Statistical MT. In: Proceedings of the 22nd International Conference on Computational Linguistics, Manchester, England, pp. 1145–1152 (2008)
3. Yamada, K., Knight, K.: A Syntax-Based Statistical Translation Model. In: Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL 2001), Toulouse, France, pp. 523–530 (2001)

[7] http://www.ichec.ie/

4. Hearne, M.: Data-Oriented Models of Parsing and Translation. PhD thesis, Dublin City University, Dublin, Ireland (2005)
5. Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., Thayer, I.: Scalable Inference and Training of Context-Rich Syntactic Translation Models. In: Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp. 961–968 (2006)
6. Lavie, A.: Stat-XFER: A General Search-based Syntax-driven Framework for Machine Translation. In: Proceedings of thr 9th International Conference on Intelligent Text Processing and Computational Linguistics, Haifa, Israel, pp. 362–375 (2008)
7. Hassan, H., Sima'an, K., Way, A.: Supertagged Phrase-based Statistical Machine Translation. In: 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007), Prague, Czech Republic, pp. 288–295 (2007)
8. Koehn, P., Hoang, H.: Factored Translation Models. In: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, pp. 868–876 (2007)
9. Chiang, D.: A Hierarchical Phrase-Based Model for Statistical Machine Translation. In: 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005), Ann Arbor, MI, pp. 263–270 (2005)
10. Chiang, D.: Hierarchical Phrase-Based Translation. Computational Linguistics 33, 201–228 (2007)
11. Carpuat, M., Wu, D.: How Phrase Sense Disambiguation outperforms Word Sense Disambiguation for Statistical Machine Translation. In: Proceedings of TMI 2007, Skövde, Sweden, pp. 43–52 (2007)
12. Tinsley, J., Hearne, M., Way, A.: Exploiting Parallel Treebanks to Improve Phrase-Based Statistical Machine Translation. In: Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories (TLT 2007), Bergen, Norway, pp. 175–187 (2007)
13. Samuelsson, Y., Volk, M.: Alignment Tools for Parallel Treebanks. In: Proceedings of the Biennial GLDV Conference, Tübingen, Germany (2007)
14. Lavie, A., Parlikar, A., Ambati, V.: Syntax-driven Learning of Sub-sentential Translation Equivalents and Translation Rules from Parsed Parallel Corpora. In: Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation (SSST-2), Columbus, OH (2008)
15. Zhechev, V., Way, A.: Automatic Generation of Parallel Treebanks. In: Proceedings of the 22nd International Conference on Computational Linguistics (CoLing 2008), Manchester, UK, pp. 1105–1112 (2008)
16. Volk, M., Samuelsson, Y.: Bootstrapping Parallel Treebanks. In: Proceedings of the 7th Conference of the Workshop on Linguistically Interpreted Corpora (LINC), Geneva, Switzerland, pp. 71–77 (2004)
17. Čmejrek, M., Cuřín, J., Havelka, J., Hajič, J., Kuboň, V.: Prague Czech-English Dependency Treebank. Syntactically Annotated Resources for Machine Translation. In: Proceedings of LREC 2004, Lisbon, Portugal, pp. 1597–1600 (2004)
18. Gustafson-Čapková, S., Samuelsson, Y., Volk, M.: SMULTRON - The Stockholm MULtilingual parallel TReebank (2007), http://www.ling.su.se/dali/research/smultron/index
19. Petrov, S., Klein, D.: Improved Inference for Unlexicalized Parsing. In: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics, Rochester, NY, April 2007, pp. 404–411 (2007)
20. Bikel, D.: Design of a Multi-lingual, parallel-processing statistical parsing engine. In: Human Language Technology Conference (HLT), San Diego, CA (2002)
21. Civit, M., Martí, M.A.: Building Cast3LB: A Spanish Treebank. Research on Language and Computation 2(4), 549–574 (2004)

22. Tinsley, J., Zhechev, V., Hearne, M., Way, A.: Robust Language-Pair Independent Sub-Tree Alignment. In: Machine Translation Summit XI, Copenhagen, Denmark, pp. 467–474 (2007)
23. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open Source Toolkit for Statistical Machine Translation. In: 45th Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic, pp. 177–180 (2007)
24. Stolcke, A.: SRILM - An Extensible Language Modeling Toolkit. In: Proceedings of the International Conference Spoken Language Processing, Denver, CO (2002)
25. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a Method for Automatic Evaluation of Machine Translation. In: 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), Philadelphia, PA, pp. 311–318 (2002)
26. Doddington, G.: Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In: Human Language Technology: Notebook Proceedings, San Diego, CA, pp. 128–132 (2002)
27. Banerjee, S., Lavie, A.: METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In: Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at ACL 2005, Ann Arbor, MI (2005)
28. Koehn, P.: Statistical Significance Tests for Machine Translation Evaluation. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, pp. 388–395 (2004)

Cross-Language Frame Semantics Transfer in Bilingual Corpora

Roberto Basili[1], Diego De Cao[1], Danilo Croce[1], Bonaventura Coppola[2], and Alessandro Moschitti[2]

[1] Dept. of Computer Science,
University of Roma Tor Vergata, Roma, Italy
{basili,croce,decao}@info.uniroma2.it
[2] University of Trento, Italy
{coppola, moschitti}@disi.unitn.it

Abstract. Recent work on the transfer of semantic information across languages has been recently applied to the development of resources annotated with Frame information for different non-English European languages. These works are based on the assumption that parallel corpora annotated for English can be used to transfer the semantic information to the other target languages. In this paper, a robust method based on a statistical machine translation step augmented with simple rule-based post-processing is presented. It alleviates problems related to preprocessing errors and the complex optimization required by syntax-dependent models of the cross-lingual mapping. Different alignment strategies are here investigated against the Europarl corpus. Results suggest that the quality of the derived annotations is surprisingly good and well suited for training semantic role labeling systems.

1 Motivation

The availability of large scale semantic lexicons, such as Framenet ([1]), has allowed the adoption of a vaste family of learning paradigms in the automation of semantic parsing. Building on the so called *frame* semantic model, the Berkeley FrameNet project [1] has developed a frame-semantic lexicon for the core vocabulary of English since 1997. As defined in [2], a frame is a conceptual structure modeling a prototypical situation. A frame is evoked in texts through the occurrence of its lexical units (LU), i.e. predicate words (verbs, nouns, or adjectives) that linguistically expresses the situation of the frame. Each frame also specifies the participants and properties of the situation it describes, the so called frame elements (FEs), that are the Frame Semantics instantiation of semantic roles. For example the frame CATEGORIZATION has lexical units such as: *categorize,classify,classification,regard*. Semantic roles shared by these predicates, are the COGNIZER (i.e. the person who performs the categorization act), the ITEM construed or treated, the CATEGORY (i.e. the class which the item is considered a member of) and CRITERIA. Semantic Role Labeling (SRL) is the task of automatic labeling individual predicates togheter with their major roles (i.e. frame elements) as they are grammatically realized in input sentences. It has been a popular task since the availability of the PropBank and Framenet annotated corpora [3], the seminal work of [4] and

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 332–345, 2009.

the successful CoNLL evaluation campaigns [5]. Statistical machine learning methods, ranging from joint probabilistic models to support vector machines, have been largely adopted to provide accurate labeling, although inherently dependent on the availability of large scale annotated resources.

It has been observed that the so called resulting *resource scarcity problem* affects a large number of languages for which such annotated corpora are not available [6]. Recent works thus explored the possibility of the cross-linguistic transfer of semantic information over bilingual corpora in the development of resources annotated with frame information for different European languages ([7,6,8]). As SRL on English texts can rely on extensive resources, the English portion of a bilingual corpus can be labelled with a significant accuracy: the cross-language transfer of predicate and role information is an appealing process aiming to produce large scale information in a relatively cheap way. The approach discussed by Sebastian Pado focused on methods for the cross-lingual induction of frame semantic information aiming at creating frame and role annotations for new languages. Based on Framenet, as a source of semantic information, it has been influential on later attempts, as for example in [7,8]. The main aspects of this work are the neat separation between alignment at the level of predicates (usually single words) and the level of roles. The first problem is tackled in [6] by relying on distributional models of lexical association that allow to estimate when a given lexical unit is in fact expressing a predicate (frame). This supported a light approach to the predicate alignment task with significant accuracy. The second problem is approached through the syntactic alignment of constituents that are *role bearing phrases*, i.e. that express sentential roles of the target predicates. These methods allow to rely on the linguistic information encoded in the syntactic bracketing and alleviate word alignment errors. Results are characterized by higher-precision projections even over noisy input data, typically produced by shallow parsing techniques (e.g. chunking).

The key problem of these classes of approaches is the complexity in devising the suitable statistical models that optimize the transfer accuracy. They have to account for word level alignments, syntactic constituency in both languages, the symmetry of the semantic role alignment relation that feed the model estimation and for the optimization process. In [6] different models are studied and several model selection strategies are presented. The best reported models are based on full parses for both languages that compensate against noisy word alignments. However, these are also shown to be sensible to the parse errors, that are quite common. As errors cumulate across complex preprocessing stages, one of the major limitation of the semantic transfer approaches is their sensitivity to noise in basic preprocessing steps, that may critically deteriorate the overall quality of the transfer outcome. Robust transfer methods of English annotated sentences within a bilingual corpus should avoid complex alignment models to determine more shallow and reusable approaches to semi-supervised SRL. The aim of this paper is the investigation of an architecture based on a controlled, yet scalable, statistical machine translation process. It exploits the conceptual parallelism provided by Framenet and a distributional model of frame instance parallelism between sentences, that guarantees a controlled input to the later translations steps. It also employs a unified semantic transfer model for predicate and roles. The result is a light process for semantic transfer in a bilingual corpus. In Section 2, the overview and details of the proposed

process are discussed, while the experimental evaluation on a bilingual English-Italian corpus is discussed in Section 3.

2 Cross-Language Transfer of Frame Semantics in Aligned Corpora

Reusing semantically annotated texts in English within bilingual corpora implies the ability of transferring semantic information from the source language sentences to the target ones, as a form of translation of semantic units (i.e. predicates and roles) from one language to the other. The specific semantic transfer problem can not be seen as a pure translation process. The presence of relatively free translations in bilingual corpora in fact does not allow to track and recover all semantic phenomena in the target sentences. Moreover, as the sentence in the target language is already available, proceeding through a translation from scratch is not even required. A more specific definition is thus necessary.

Given a bilingual corpus in English and in a second target language T (e.g. Italian), the semantic transfer needs first to select sentence pairs (s_E, s_T) that effectively realize a specific frame f, and then provide the frame annotations for f in the target language sentence s_T. This process may proceed by labeling the English sentence s_E through an existing highly-performant SRL system, deriving multiple translation possibilities of English segments in s_E through statistical MT tools, and then building the best available semantic annotations within the target language sentence s_T. While related work on this process (including [6,8]) is generally based on complex syntactic models, our aim is to define a method relatively independent on the syntactic constraints on the two languages, in order to support a larger scale approach. The proposed process is depicted in Fig. 1. It combines a statistical translation tool (i.e. Moses) and a sentence selection model. This latter allows to decide which sentence pairs in the aligned corpus are effective realizations of frames. Statistical machine translation here is used to collect translation candidates for semantic information: every annotated role in the English portion of the corpus gives rise here to segments whose partial translations are available in terms of phrase translation pairs (PT pairs in Figure 1) from the corpus ([9]). These are thus post processed to get the suitable role boundaries in the target sentences (Semantic alignment step in Fig. 1).

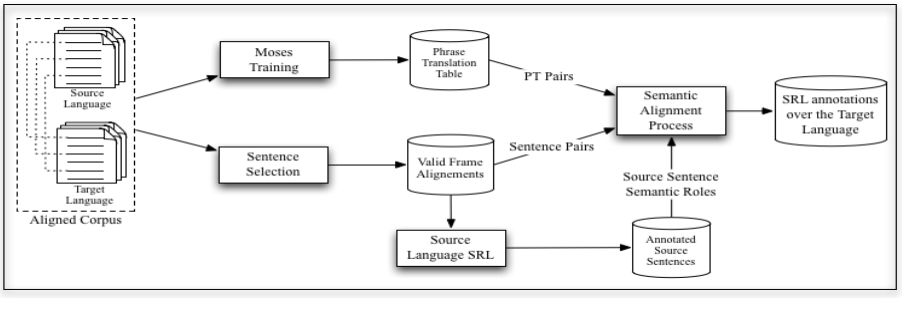


Fig. 1. The semantic transfer workflow

2.1 Cross-Language Predicate Level Alignment of Sentences

In a bilingual corpus, the parallelism of roles is conditional on the so-called *frame instance* parallelism ([6]): unless the frame expressed by two sentences is the same, the roles cannot be observed in parallel. The starting point of the semantic transfer approach is thus the selection of suitable sentences pairs as candidate expressions of frames. The underlying aligned corpus provides sentence paris (s_E, s_I) where Frame information about target predicates and roles (hereafter semantic elements) are both expressed in English and Italian. An aligned sentence pair (s_E, s_I) is a *valid* example of a frame f if both sentences express the specific semantic information related to f, i.e. exhibit conceptual and instance parallelism about f. We are interested to valid sentence alignments where the given frame f is known to manifest. The knowledge of predicate words of f (i.e. lexical units, $LU(f)$) in both languages is thus a starting point[1]. A pair (s_E, s_I) represents a *potentially valid frame alignment* for f iff $\exists p_E \in LU_E(f)$ and $\exists p_I \in LU_I(f)$ such that $p_E \in s_E$ and $p_I \in s_I$, where p_E or p_I are predicate words for f. However, this constraint is not sufficient as lexical units can be ambiguous so that not all valid frame alignments capture the same corresponding unique frame. In order for a pair to support the transfer of the semantic elements, the sentences must be known as expressions of the **same** frame f. For example in the sentence pair

s_E: *I will make his statement in English*
s_I: *Intendo farlo citando il suo intervento in inglese*

the verb *make* is not a predicate of the MANIFACTURE frame, although both *make* and *fare* are legal lexical units for the MANIFACTURE in both languages.

What it is needed here is a suitable model of valid frame alignments (s_E, s_I), that guarantees that a frame is expressed in s_E and s_I. At this aim we define the following function, called *pair frame relevance*, pf_rel:

$$pf\_rel((s_E, s_I), f) = \Gamma(\sigma_E(s_E, f), \sigma_I(s_I, f)) \quad (1)$$

where $\sigma_E(s_E, f)$ and $\sigma_I(s_I, f)$ measure the relevance of s_E and s_I respectively for f, and $\Gamma(.)$ is a composition function, such as the product or the linear combination.

The relevance $\sigma(s, f)$ of sentences for a given frame f is approached here according to methods based on semantic spaces already applied to LU classification ([11]). Semantic spaces are first built from co-occurrence analysis of lexical units, and distance in the resulting space is used to measure the suitable frame for possibly unknown predicate words. The method is semi-supervised as known lexical units of a frame f are used as examples of regions of the semantic space in which f manifests. First, a clustering process is applied to the set of known lexical units (LU) for f[2]. The centroids of the derived clusters are then used as a representation of f: distances from centroids are used to detect the suitable frames for vectors of unknown predicates[3]. In essence, the distance from clusters of a frame f represents cues to suggests frames of novel words. As Latent

[1] In [10], a LSA-based method to compute lexical classification also for Italian is presented, and, accordingly, a lexicon of about 15,000 predicate words has been made available. This resource is used across all the experiments reported in this paper.

[2] The adopted clustering process called *qt-kMean* [12] has been applied to collect these regions.

[3] In [10,11], this process is also strengthen by the use of Wordnet synonymy information.

Semantic Analysis [13] is applied to the original space, for its duality property, sentences (i.e. pseudo documents) can be expressed in the same space of LUs[4] : similarity between sentences and frames can be thus computed in terms of a distance function. Details of this process are discussed in [10].

Given a raw source corpus (e.g. the two monolingual portions of the bilingual corpus) a corresponding semantic space can be built. Then, the *Sentence Frame relevance* $\sigma(s, f)$ of a sentence s for a frame f is defined by:

$$\sigma(s, f) = max(0, max_{C_f}\{sim(\boldsymbol{s}, \boldsymbol{c(C_f)})\}) \quad (2)$$

where C_f are clusters derived from the known LU's of the frame f in the semantic space, $\boldsymbol{c(C)}$ is the centroid of the cluster C, $\boldsymbol{s}$ denotes the representation of s in the semantic space and $sim(.,.)$ is the usual *cosine similarity* among vectors. Notice how only k dimensions characterize the semantic space after the application of the Singular Value Decomposition (SVD) [13]. When any two corpora in English and Italian are available, two different semantic spaces are defined, but comparable scores $\sigma(.,.)$ can be obtained. As a consequence Eq. 2 and 1 can be computed for any language pair. The ranking determined among valid sentence pairs by Eq. 1 allows to automatically select the pairs for which conceptual parallelism for f is realized with high confidence. Notice that both sentences are constrained so that reliable pairs can be selected, SRL can be applied to their English side and, finally, the predicate and role alignments step towards the target language can be applied.

2.2 Robust Cross-Lingual Alignment of Frame Annotations

The task of computing the correct cross-lingual alignment of semantic information, as made available by an automatic frame annotation system, consists in the detection of segments expressing the semantic information related to the target predicate and to all the frame elements, as they are realized in the target language sentence s_I in a pair (s_E, s_I). As the translation s_I is often not literal, we can not assume that s_I *always* expresses *all* the FE observed in the English sentence s_E. However, exceptions are fewer, and the full labeling of s_I can proceed as a search for the segments in s_I triggered by the individual semantic elements found in s_E. In the following, we will adopt this view: each alignment choice is tailored to detect the unique segment in s_I able to realize the same information as one source semantic element annotated in s_E. Semantic elements here include the target predicates (usually verb phrases or nominal predicates in s_E) or phrases expressing some frame elements (FE): these are thus *always* explictly realized as segments in s_E. In the example, s_E: *I think this is something we should study in the future*, the segment "[*think*]" realizes the predicate OPINION while ["*this is something we should study in the future*"] accounts for the realization of the CONTENT FE.

Given a valid frame alignment pair (s_E, s_I), a role α and its realization in s_E, namely $s_E(\alpha)$, the alignment task can be thus formalized as the function $SemAl()$ defined by:

$$SemAl((s_E, s_I), \alpha, s_E(\alpha)) = s_I(\alpha) \quad (3)$$

[4] Any sentence s is represented as the linear combination of the vectors built from its words t, i.e. $\boldsymbol{s} = \Sigma_{t \in s}\omega(t, s) \cdot \boldsymbol{t}$, where $\omega(t, s)$ is the usual $tf \times idf$ score. $\boldsymbol{s}$ is finally normalized in the semantic space, where $\boldsymbol{t}$ are computed.

$SemAl(.)$ computes the proper segment $s_I(\alpha)$ that realizes α in s_I. As described in Fig. 1, the function $SemAl()$ proceeds by first detecting all the possible translations pairs for subsegments produced by a statistical MT tool (i.e. Moses), and then by merging and expanding the set of potential translation choices.

The Statistical Translation Step. Recently, MOSES ([9]), an open-source toolkit for statistical machine translation (SMT) has been released, exploiting the idea of factored translation models and confusion network decoding. It performs highly flexible phrase-level translation with respect to other traditional SMT models. Some of its key advantages are the exploitation of constraints (and resources) from different linguistic levels that are thus factored within a unique translation model. In [14], factored models on the Europarl corpus, [15], are shown to outperform standard phrase-based models, both in terms of automatic scores (gains of up to 2% BLEU) as well as grammatical coherence. In our work, the open source Moses system [9] has been used on the English-Italian aligned portion of the Europarl corpus [15]. During training, Moses produces translation models over phrase structures that are stored as phrase translation (PT) tables.

In this work, translation refers to the ability of cross-language mapping of individual semantic elements. The translation from English is thus not "blind" but guided by the expectations raised by the available sentence in the target language. Instead of relying on the automatic translation, it is possible to analyze only the partial translations of s_E that in fact appear in the target sentence s_I. In this case, simple phrase level translations are more useful, as they represent translations of partial elements from which the detection of the entire targeted role is enabled. Phrase-level alignments among the individual source sentences are made available as translation tables of English and Italian phrases (including singleton words). In the sentence "*I regard the proposed charter of fundamental rights as an opportunity to bring the european union closer to the people*" the following segment,

s_E: *as an opportunity to bring the European Union closer to the people.*

represents the role CATEGORY for the underlying CATEGORIZATION frame, introduced by the verb *regard*. The corresponding segment in the Italian counterpart is:

s_I: *come un'opportunita' per avvicinare l'Unione Europea ai cittadini.*

An excerpt of the phrase alignments provided by the Moses phrase translation (PT) table, acquired on the Europarl corpus, is shown in Fig. 2.(A-B). Notice how word pairs, e.g. *(closer, avvicinare, 0.06)*, *(closer, unione,0.00001)* in (A), are characterized by very low probabilities due to the relatively free translation: here the verb phrase "*bring closer*" is expressed by a single Italian verb *avvicinare*, and the translation mapping can not be more precise.

By extending the pairwise word alignments, Moses accounts for phrase-level alignments with probabilities. Moses phrase translation tables define all segments s_E that have a translation included in s_I, whereas, for a single semantic element, all its parts that have partial translations in s_I can be found, as shown in Fig. 2.(B). The output of the statistical alignment phase is thus a set of segment pairs (es^i, is^j) weighted according to a probability, describing a generally many to many mapping between an English semantic element and some is^j segments in s_I. Pairs include: word pairs as well as pairs where the English source is covered by a longer Italian segment (i.e.

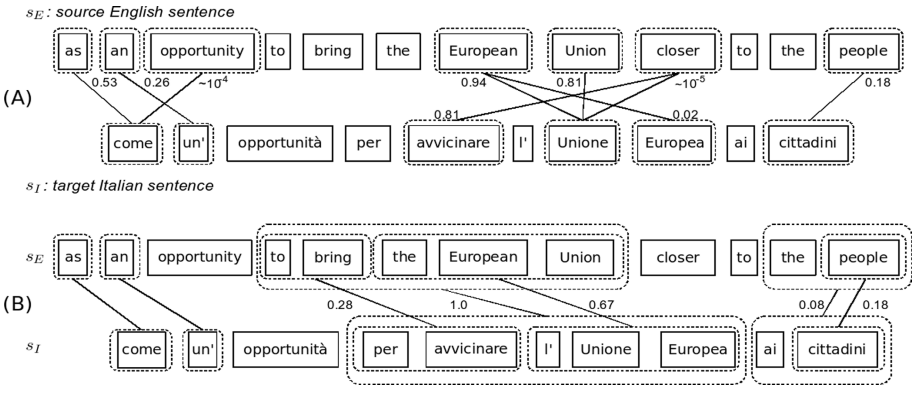


Fig. 2. An example of Moses alignments

$length(es^i) > length(is^j)$) or viceversa. A first *basic* algorithm for the function $SemAl((s_E, s_I), \alpha, s_E(\alpha))$ can be made dependent just on the Moses translation table. In this simple case, used hereafter as a baseline, the result $s_I(\alpha)$ is defined as the Italian segment is^j such that it exactly covers the English semantic element, i.e. such that it is translated from es^i with $es^i = s_E(\alpha)$.

In the example of Figure 2 (B), if a role α (e.g. THEME) characterize es^i=["*the European Union*"], the baseline alignment would result in ["*l'Unione Europea*"]. Unfortunately, in most cases, perfect matches are not made available as we will also see in section 3: roles are often realized in long segments, i.e. the targeted $s_E(\alpha)$, for which only partial segments es^i are translated. Further processing steps are thus needed to make a final decision about the best alignment of $s_E(\alpha)$ in s_I.

The above example shows that the length (k) of the English segment, the length of the Italian segment and the Moses output probabilities are all cues that characterize the quality of (partial) translation pairs (es^i, is^j) for the semantic transfer of a role α. Three different strategies can thus be used:

- English segment length policy, e_length: by adopting k as a ranking criterion, translation segments related to longer subsequences of the targeted ones, i.e. $s_E(\alpha)$ are preferred and selected first.
- Italian segment length, or i_length: the longer Italian are here preferred, so that better translation segments correspond to longer is^j.
- $simpleprob$: the $simpleprob$ policy ranks higher the segments es^i that appear in translation pairs with higher probabilities

Robust Cross-Lingual Semantic Alignments. The general algorithm for computing the semantic alignment is triggered by a sentence pair, (s_E, s_I), a specific element (e.g. a role) α and the English segment expressing the role $s_E(\alpha)$. It proceeds through the following steps:

1. *Rank phase*. Rank all the Moses translation segments related to at least one word in $s_E(\alpha)$, according to one policy (e.g. e_length).

2. *Collect Phase*. Scan the translation pair table, from the best pair to the worse ones, and select candidates for all token in $s_E(\alpha)$ until the target English segment is not covered by at least one translation. In this phase, all the Italian segments that are translations of a yet uncovered English segment es^i in $s_E(\alpha)$ are selected
3. *Boundary Detection Phase*. Process all the collected italian segments and compute the best boundary, i.e. $s_I(\alpha)$. This is done by possibly merging adjacent Italian candidate segments, or filling gaps between non-adjacent ones.
4. *Post-Processing Phase*. Refine the computed boundaries by applying heuristics based on the entire sentence, i.e. according to the candidate solutions of all different semantic elements. A typical task in this phase is the pruning of potential overlaps between translations $s_I(\alpha)$ of different roles built in the *Boundary Detection Phase*.

Notice that the above general process is greedy. First, the targeted English segment $s_E(\alpha)$ is early used to prune irrelevant portions of the (English and Italian) sentences. Second, the selected policy determines the order by which individual translation pairs are collected. Given the above general strategy, different ranking models and the adoption (or skip) of the post processing step characterize different workflows. As the *Boundary Detection Phase* provides complete solutions $s_I(\alpha)$, it can be also retained as a final step, without applying any post processing.

Collect Phase. The algorithm that compute translation candidates for individual roles α is in Fig. 3. The operators $\sqcap$ compute here the common subsequences among the segment operands, while $A \setminus B$ denotes the sequence obtained by removing the segment B from A.

FUNCTION Select(α, % The targeted role
$s_E(\alpha)$ % The targeted segment
$MosesPairs$) % Ranked pairs (es^i, is^j)
$CandidateSeq = \emptyset$
$CurrTargetSeq = s_E(\alpha)$
while $(CurrTargetSeq \neq \emptyset)$ AND $(MosesPairs \neq \emptyset)$ **do**
(es^i, is^j) = POP($MosesPairs$)
$Overlap = CurrTargetSeq \sqcap es^i$
if $Overlap \neq \emptyset$ **then**
$CurrTargetSeq = CurrTargetSeq \setminus Overlap$
$CandidateSeq = CandidateSeq \cup \{(es^i, is^j)\}$
end if
end while
return $(CandidateSeq)$

Fig. 3. The Algorithm for the *Collect Phase*

Boundary Detection Phase. Once candidate translation pairs are selected and ranked according to a given policy, a solution is then built by merging adjacent Italian segments is^j. As some words (or segments) may not appear in the translation tables, merging may not produce effective subsequences of the Italian sentence. In this case potential gaps between the selected is^j are filled. In the example of Fig. 2, the available translation pairs, are first selected and then merged to cover new portions of the English role segment, $s_E(\alpha)$. In this case, [*per avvicinare l'Unione Europea*] is first merged with

[*ai cittadini*] as they are the best selected segments in the first step. Then [*come*] and [*un*] are also added and merged as they translate new tokens in $\alpha(s_E)$ (i.e. [*as*], [*an*]). Finally, the gap between [*come un*] and [*per avvicinare l'Unione Europea ai cittadini*], due to the missing translation for the Italian [*opportunita'*], is filled: the final output boundary is [*come un'opportunita' per avvicinare l'Unione Europea ai cittadini*] that in fact captures the entire CATEGORY role for the underlying CATEGORIZATION frame.

Post-Processing Phase. The *Boundary Detection* process applies independently to individual roles (or predicates) α. It is thus possible that the produced solutions for different roles include partially overlapping segments. However, when the solutions for all roles α are made available, possible inconsistencies can be detected and ambiguities solved. For example, violations to the planarity of the solution (i.e. overlaps between the different output role segments), can be forced by some adjustment. One typical case, often caused by grammatical movements of inner constituents of roles, is given by output segments for frame elements that, in the Italian syntax, also include the target, as in:

s_E: [*I*]$_{Cognizer}$ [*think*]$_{target}$ [*this is something we should study in the future*]$_{Content}$.
s_I: *Lo reputo un tema meritevole di essere approfondito in futuro*.

Here the subject of the predicate is not expressed in Italian and the pronoun Lo, corresponding to the determiner $this$, is prefixed to the predicate. Here, the *Boundary Detection* algorithm produces the following wrong span for the CONTENT role: [*Lo reputo un tema meritevole di essere approfondito in futuro*], i.e. the entire s_I sentence. The objectve of the post processing here is to superimpose planarity by discarding embedded solutions. The original solution for CONTENT is first segmented in the two portions, [*Lo*] and [*un, tema, meritevole, di, essere, approfondito, in, futuro*], by cutting out the predicate. Then, the correct right segment [*un, tema, meritevole, di, essere, approfondito, in, futuro*] is selected as it constitutes the longer solution. The final full annotation of the CATEGORIZATION frame in the example s_I is:

$$Lo\ [reputo]_{Target,Cognizer}\ [un\ tema\ meritevole\ di\ essere\ approfondito\ in\ futuro]_{Content}.$$

that is in line with the semantic expectations provided by s_E[5].

3 Evaluation

There are mainly two different aspects of the proposed semantic transfer process worth of an in depth investigation. The first is the evaluation of the Sentence extraction step (Section 3.1) as determined by Equation 1. The second is the evaluation of accuracy of the overall semantic transfer, as reachable by the technique proposed in Section 2.2.

The computation of the ranking factor defined in Eq. 1 requires a vector representation for both the English and Italian sentences. As described in [10], the semantic space is derived through LSA, over the English and Italian components of the Europarl corpus

[5] The ellipsis of the agentive role, $Cognizer$, for the Italian sentence is here expressed through the multiple tags for the predicate word *reputo*. Notice that these multiple tags are not considered during the evaluation discussed in Section 3 and only the independently realised roles, i.e. the target in this case, are measured.

[15]. The vector components express occurrence of predicates in individual sentences (i.e. pseudo-documents), these latter used as features. The semantic space accounts for about 1 million sentences (i.e. 36 millions tokens), used as contexts for computing the co-occurrence vectors for individual words, including the targeted LUs. The SVD reduction with $k = 300$ allows to compute a 300-dimensional vectors for each word: sentences are accordingly represented by the linear combination of the vectors of their words. In all the experiments, the open source Moses system [9] has been used on the English-Italian aligned portion of the Europarl corpus [15]. Default settings are used in all the experiments.

For the evaluation of the semantic transfer accuracy, a gold standard, built from the aligned English-Italian component of the Europarl corpus, has been used. This gold standard, presented in [8], is made of 987 sentences in both languages English and Italian. The gold standard has not a complete alignment. As discussed in [8], only 61% of the sentences are annotated with the same frame, while only 82% have the same FEs in both languages. This is mainly due to the different versions of Framenet used for English (i.e. 1.1) and Italian (i.e. 1.3), as reported in [8]. As we are interested to the transfer achievable through automatic alignment of the source English annotations, we considered only the different FE alignments independently from the underlying Frame. As a consequence, the relevant test cases are only those FEs having the same label in both languages. In general this assumption does not cover all cases, but it gives a significant idea about the potential of the semantic transfer on a reasonable scale. In the gold standard, 1,727 and 1,730 frame elements were found respectively for the English and Italian component, where 881 were shared. In the 987 sentences, 984 target lexical units were aligned[6]. As the transfer of individual semantic elements proceeds from the English to the Italian sentences, we are interested in: (1) Perfect matches, i.e. the percentage of output Italian segments that are fully overlapping with the gold standard ones, (2) Partial Matches, i.e. the percentage of Italian segments with non empty intersection with the gold standard. Moreover, we also want to measure the quality of the computed approximation for each semantic element in terms of tokens. Thus we evaluate the token retrieval quality for all the translated source English roles against the Italian gold standard. The token retrieval task is measured according to the usual precision, recall and F-measure scheme: a token in $s_I(\alpha)$ is correct if it also part of the segment for α proposed by the oracle. False positives and negatives are given by tokens found only in $s_I(\alpha)$ or in the oracle respectively. These measures are a fine-grain evaluation of the overlaps between the solutions and the oracle.

3.1 Evaluating the Sentence Extraction Model

The evaluation of the sentence extraction accuracy is carried out by studying the probability distributions of the frame preference scores (Eq. 1), as computed over three sentence pair sets of similar cardinality (about 1,000 sentences). The first Control Set ($CS1$) includes sentence pairs where frame assignment is randomly applied: in this case, a randomly chosen frame f is selected for each pair and the scores $\sigma(s, f)$ are

[6] Three sentences have been neglected as for text encoding problems in the original gold standard.

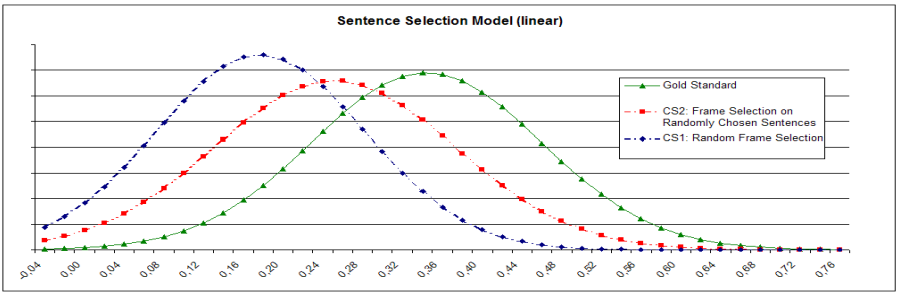


Fig. 4. Distributions of frame preference scores over the oracle and two reference Control Sets

used to compute Eq. 1 from the English and Italian sentences. A second Control Set ($CS2$) is obtained by selecting pairs for which the English sentence includes a known lexical unit of a frame f: such sentences and their Italian equivalent are then used to compute the $\sigma(s, f)$ scores in Eq. 1. Finally, the model in Eq. 1 is computed over the Oracle sentence pairs: here the frame f is known to be correct. In Fig. 4 the normal probability distributions $P(\Gamma = x)$ are reported for the three sets, where the composition function Γ is the linear combination of scores $\sigma(s, f)$ with equal weights (i.e. 0.5). As clearly indicated by the plots the mean values of the three distributions are significantly different. Increasing evidence given by higher semantic relevance scores Γ of sentence pairs corresponds to correct frames (as in the oracle). The difference between the first and the second Control Sets suggests that the knowledge about lexical units is important and it is well captured by the LSA similarity. When frame relevance holds for both languages (as implicitly true in the oracle, where the frame preference $\sigma(s, f)$ of a sentence is guaranteed to be correctly applied on both languages), the result is a strikingly higher score for the sentence pair (i.e. $\mu_{Oracle} \cong 0.36$ vs. $\mu_{CS1} \cong 0.18$). Evidence in Fig. 4 confirms that Eq. 1 allows to accurately rank sentence pairs as suitable representations for a given frame. This is useful for all the material to be annotated by an automatic process outside the gold standard, where a good conceptual (i.e. frame) parallelism is needed.

3.2 Evaluating the Overall Accuracy of the Semantic Transfer

The evaluation of the semantic transfer from English to Italian (i.e. the task described in Section 2.2) has been carried over the English-Italian gold standard. As for the mentioned mismatches between the adopted labeling for Italian and English data, tests are tailored to the subset of roles (i.e. targets and frame elements) that have the same label in both languages. The tested models are derived from the application of different ranking policies (e.g. e_length vs. $simpleprob$) as well as in the adoption of the post-processing phase ($+PP$ in table 1). The accuracy is evaluated independently over all semantic elements or just on roles (*FE only*). In this latter case, we simply neglect the targets in the accuracy computation. The baseline refers to the output of the basic algorithm defined in Section 2.2. It relies only on the Moses translations and refers to the best solution obtained through a direct look-up in the Moses PT tables.

Table 1. Accuracy of the role alignment task over the Gold Standard

Model	Perfect Matching (FE only)	Partial Matching (FE only)	Token Precision	Token Recall	Token F1
baseline	66.88% (28,37%)	72,78% (41,13%)	0.78 (0.59)	0,29 (0.14)	0.43 (0.23)
e_length	72,02% (39,48%)	90,98% (80,50%)	0.75 (0.71)	0.88 (0.85)	0.81 (0.78)
simpleprob	71,69% (38,77%)	**91,09**% (**80,73**%)	0.74 (0.70)	0.88 (0.85)	0.80 (0.77)
i_length	69,51% (34,04%)	89,56% (77,42%)	0.73 (0.69)	0.89 (0.86)	0.80 (0.77)
e_length (+PP)	**73,28**% (**42,20**%)	**89,94**% (**78,25**%)	**0.84** (**0.81**)	0.84 (0.81)	**0.84** (**0.81**)
simpleprob (+PP)	**73,28**% (**42,20**%)	89,83% (78,01%)	0.84 (0.80)	0.84 (0.81)	0.84 (0.81)
i_length (+PP)	70,92% (37,12%)	88,36% (74,82%)	0.82 (0.80)	**0.84** (**0.81**)	0.83 (0.79)

Table 1 reports the accuracy of perfect and partial matchings. Notice how the perfect matching corresponds to the usual SRL evaluation as applied to the labeling of the Italian test corpus: perfect matches here corresponds either to perfect boundary recognition and role classification. The last columns in Table 1 measure the gap in accuracy between *Perfect* and *Partial Matches*. Higher values in F1 suggest that tokens violating predicate and role boundaries are fewer.

As shown in table 1, the best model (i.e. $e\_length + PP$) achieves perfect matching for 42% of the Frame Elements (excluding target words) and 73% of all roles in the test sentences. Results for partial matching, according to the same approach reach percentage of respectively 78,25% and 89,94%. This shows that the proposed approach are almost everywhere able to find the correct core of individual semantic elements. Only few tokens violate boundaries, but most of the FE semantics is preserved. This is confirmed by the evaluation of tokens retrieval (see last three columns in Table 1), as a 81% of F1 is achieved only on the transfer of FEs. Notice how all the models are well above the baseline, obtained by relying just on Moses phrase translation pairs. This is particularly noticeable on FEs: notice that this is mainly due to the fact that targets are usually expressed by shorter segments, in general verbs, for which the higher frequencies in the Europarl allow Moses to produce more accurate translations. This is unfortunately no longer true for semantic roles, for which the baseline performs quite poorly, about 28% perfectly matched roles, with F1=0.23 at the token level.

4 Conclusions

Complex models for semantic cross-lingual transfer of Framenet information require highly performant parser and complex model optimization. In this paper a light, yet robust, semantic transfer method has been presented aiming to produce large scale frame semantic annotations over bilingual corpora. Although no direct comparison was made possible with respect to previous work (basically, for major differences in the adopted languages, measures and representations), the obtained results appear superior to previously proposed methods. A public distribution of the aligned material is foreseen for stimulating further comparative analysis. The adoption of unsupervised techniques for sentence selection as well as the poorer requirements of the semantic transfer approach

here proposed imply a larger applicability with more space for improvements. First of all, the approach is open to improvement through further grammatical analysis of the proposed alignments: chunking and parsing can be still applied to refine possibly wrong solutions and increase the token-level precision. Moreover, better statistical modeling of alignment preferences (through joint bayesian models) should be investigated to further improve the boundary detection step. The presented methodology has been currently applied to extend to current English-Italian gold standard of [8]. An existing SRL system, described in [16,17,18], has been used to annotate data outside the gold standard, i.e. about 20,831 sentences. As a result about 17,765 among the analysed sentences have been annotated in Italian with two or more roles. A relevant open issue is thus the evaluation of its impact on the learning of the current SVM-based SRL system for Italian. If the potential advantages in adopting a large scale (but noisy) training set with respect to smaller high-quality gold standards could be assessed, this would definitively open new perspectives on the use of bilingual corpora for a semi-supervised approach to SRL training.

Acknowledgments. The authors are thankful to the FBK group for granting the access to the gold standard developed by Emanuele Pianta and Sara Tonelli at FBK.

References

1. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The Berkeley FrameNet project. In: Proc. of COLING-ACL 1998, pp. 86–90 (1998)
2. Fillmore, C.J.: Frames and the semantics of understanding. Quaderni di Semantica 4(2), 222–254 (1985)
3. Palmer, M., Gildea, D., Kingsbury, P.: The Proposition Bank: an Annotated Corpus of Semantic Roles. Computational Linguistics 31(1), 71–106 (2005)
4. Gildea, D., Jurafsky, D.: Automatic Labeling of Semantic Roles. Computational Linguistics 28(3), 245–288 (2002)
5. Carreras, X., Màrquez, L.: Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In: Proc. of CoNLL 2005, Ann Arbor, Michigan, pp. 152–164 (2005)
6. Padïo, S.: Cross-lingual annotation projection models for role-semantic information. PhD Thesis, Dissertation, Universität des Saarlandes, Saarbrücken, Germany (2007)
7. Padïo, S., Pitel, G.: Annotation prïecise du francais en sïemantique de roles par projection cross-linguistique. In: Proc. of TALN 2007, Toulouse, France (2007)
8. Tonelli, S., Pianta, E.: Frame information transfer from english to italian. In: Proc. of LREC Conference, Marrakech, Marocco (2008)
9. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open source toolkit for statistical machine translation. In: Annual Meeting of the Association for Computational Linguistics (ACL), Demonstration Session, Prague, Czech Republic (2007)
10. De Cao, D., Croce, D., Pennacchiotti, M., Basili, R.: Combining word sense and usage for modeling frame semantics. In: Proc. of The Symposium on Semantics in Systems for Text Processing (STEP 2008), Venice, Italy, September 22-24 (2008)
11. Roberto, B., De Cao, D., Pennacchiotti, M., Croce, D., Roth, M.: Automatic induction of framenet lexical units. In: Proc. of the 12th International Conference on Empirical Methods for NLP (EMNLP 2008), Honolulu, USA (2008)

12. Heyer, L., Kruglyak, S., Yooseph, S.: Exploring expression data: Identification and analysis of coexpressed genes. Genome Research (9), 1106–1115 (1999)
13. Landauer, T., Dumais, S.: A solution to plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. Psychological Review 104, 211–240 (1997)
14. Koehn, P., Hoang, H.: Factored translation models. In: Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic, pp. 868–876 (2007)
15. Koehn, P.: Europarl: A parallel corpus for statistical machine translation. In: Proc. of the MT Summit, Phuket, Thailand (2005)
16. Moschitti, A.: Making Tree Kernels Practical for Natural Language Learning. In: Proc. of EACL 2006, pp. 113–120 (2006)
17. Moschitti, A., Pighin, D., Basili, R.: Tree Kernels for Semantic Role Labeling. Computational Linguistics Special Issue on Semantic Role Labeling (3), 245–288 (2008)
18. Coppola, B., Moschitti, A., Pighin, D.: Generalized Framework for Syntax-based Relation Mining. In: Proceedings of the IEEE International Conference on Data Mining (ICDM 2008), Pisa, Italy (2008)

A Parallel Corpus Labeled Using Open and Restricted Domain Ontologies*

E. Boldrini, S. Ferrández, R. Izquierdo, D. Tomás, and J.L. Vicedo

Natural Language Processing and Information Systems Group
Department of Software and Computing Systems
University of Alicante, Spain
{ebolrini,sferrandez,ruben,dtomas,vicedo}@dlsi.ua.es

Abstract. The analysis and creation of annotated corpus is fundamental for implementing natural language processing solutions based on machine learning. In this paper we present a parallel corpus of 4500 questions in Spanish and English on the touristic domain, obtained from real users. With the aim of training a question answering system, the questions were labeled with the expected answer type, according to two different ontologies. The first one is an open domain ontology based on Sekine's Extended Named Entity Hierarchy, while the second one is a restricted domain ontology, specific for the touristic field. Due to the use of two ontologies with different characteristics, we had to solve many problematic cases and adjusted our annotation thinking on the characteristics of each one. We present the analysis of the domain coverage of these ontologies and the results of the inter-annotator agreement. Finally we use a question classification system to evaluate the labeling of the corpus.

1 Introduction

A corpus is a collection of written or transcribed texts created or selected using clearly defined criteria. It is a selection of natural language texts that are representative of the state of the language or of a special variety of it. Corpus annotation is a difficult task due to the ambiguities of natural language. As a consequence, annotation is time-consuming, but is extremely useful for natural language processing tasks based on machine learning, such as word sense disambiguation, named entity recognition or parsing.

Question answering (QA) is the task that, given a collection of documents (that can be a local collection or the World Wide Web), retrieves the answers to queries in natural language. The purpose of this work is the development of a corpus for training a question classification system. Question classification is one of the tasks carried out in a QA system. It assigns a class or category to the searched answer. The answer extraction process depends on this classification,

* This research has been partially funded by the Spanish Government under project CICyT number TIC2003-07158-C04-01 and by the European Commission under FP6 project QALL-ME number 033860.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 346–356, 2009.

as different strategies may be used depending on the question type detected. Consequently, the overall performance of the system depends directly on question classification.

We have developed a parallel corpus of 4500 questions in English and Spanish, that has been employed in the QALL-ME European project.[1] Every question in this corpus has been labeled with its expected answer type (EAT), defined in the literature as "the class of object sought by the question". We labeled these questions using two different ontologies. The first one is Sekine's [15], an ontology suitable for open domain question answering systems, like those traditionally presented in conferences such as TREC[2] and CLEF[3]. The second one is a restricted ontology on the touristic domain, that has been created *ad hoc* for the QALL-ME project [12]. The aim of this double annotation is to allow training question classification systems in both open, and restricted domains.

In this paper, we present the different features of the ontologies, and we also propose the solution we adopted for the most problematic cases of annotation. Moreover, in order to test the coherence of the corpus, we examine the inter-annotator agreement and the performance of a question classification system trained on the corpus.

The rest of the paper is organized as follows: in Section 2 related work is presented. Sections 3 and 4 present a detailed description of the corpus and the ontologies we have employed. Afterwards, in Section 5 we explain the annotation process and we examine the most problematic cases. Section 6 presents the evaluation of the annotation of the EAT in the corpus. Finally, Section 7 depicts conclusions and future work proposals.

2 Related Work

There is a wide range of QA systems that apply machine learning techniques based on corpus, covering different stages of the question answering task. In [13], they developed a corpus of question-answer pairs called KM database. Each pair of the KM represents a trivia question and its answer, such as the trivia card game. The question-answer pairs were filtered to get only questions and answers that are similar to the ones presented in the TREC task. Using this corpus they automatically collected a set of text patterns employed for answer extraction.

In [16], they present a QA system around a noisy-channel architecture which exploited both a language model for answers, and a transformation model for

[1] QALL-ME is an EU-funded project which aims to establish a shared infrastructure for multilingual and multimodal question answering in the tourism domain. The QALL-ME system (`http://qallme.fbk.eu/`) allows users to pose natural language questions in several languages (both in textual and speech modality) using a variety of input devices (e.g. mobile phones), and returns a list of specific answers formatted in the most appropriate modality, ranging from small texts, maps, videos, and pictures.

[2] http://trec.nist.gov

[3] http://www.clef-campaign.org

answer/question terms. In order to apply the learning mechanisms, they first created a large training corpus of question-answer pairs with a broad lexical coverage. They collected FAQ pages and extracted a total of one million question-answer pairs. After that, they employed this training corpus in the query analysis, and in the answer extraction modules.

In the work presented in [1], they developed a system that used a collection of approximately 30,000 question-answer pairs for training. The corpus was obtained from more than 270 FAQ files on different subjects in the *FAQFinder* project [4]. They used this corpus to automatically learn phrase features for classifying questions into different types, and to generate candidate query transformations.

Finally, [3] proposed another approach based on machine learning. Starting from a large collection of answered questions, the algorithms described learned lexical correlations between questions and answers.

Nowadays, there is also a wide range of research projects focused on the touristic domain, and more specifically on the creation of restricted domain ontologies; the main objective is to investigate complex language technologies and web technologies to improve information searching and accessing in this data-rich area. In the following paragraphs we present some of the most relevant.

The first is The *Harmonise* ontology, developed during the *Harmonise* project[4], and then extended to new subdomains in the project of Harmonise Trans-European Network for tourism (Harmo-TEN). The aim of the two related projects was to provide an open mediation service for travel, and tourism information exchange within the tourism industry members.

It is also important to mention the*Hi-Touch Ontology* that is an IST/CRAFT European program, which aimed to develop Semantic Web methodologies and tools for intra-European sustainable tourism. The *Hi-Touch ontology* was developed mainly by Mondeca, using the "Thesaurus on Tourism and Leisure Activities" (World Tourism Organization, 2001) as an official source for its terminology. Moreover the ontology focuses on tourism products and customers' expectations. Its usage can ensure the consistency of categorization of tourism resources managed on different databases, and enhances searches among numerous tourism products by providing semantic query functionalities.

The *eTourism Semantic Web portal* was developed by Digital Enterprise Research Institute[5]; it consisted of a search interface, where information retrieval was based on the semantic data to allow better queries. In order to provide vocabulary for annotations, and obtain agreement on a common specification language for sharing semantics an ontology was used. It mainly covers accommodation and activities, including also the necessary infrastructure for the activities.

TAGA[6] is an agent framework for simulating the global travel market on the Web. In *TAGA*, all travel service providers can sell their services on the Web forming a travel market; travel agents can help customers to buy the travel

[4] http://www.cepis-harmonise.org/harmonise/php/
[5] http://e-tourism.deri.at/
[6] http://taga.sourceforge.net/

package from the Web travel market according to the customers' preferences. *TAGA* defines the domain ontologies to be used in simulations.

The BMBF funded project-German Text Exploitation and Search System *(GETESS)*[7], aimed at developing an intelligent Web tool for information retrieval in the tourism domain. *GETESS* enables natural language description of search queries through navigation in a domain-specific ontology, and presents the results in an understandable form. The *GETESS* ontology [8] contains 1043 concepts and 201 relations and provides bilingual terms (English and German) for each concept. It is the central service for text mining, storage, and query of semantic content by determining which facts may be extracted form texts, which database schema must be used to store these facts and what information is made available at the semantic level [17].

None of the corpus presented here, offer the double annotation that we describe in this work, and that allows us to train a question answering system in both open, and restricted domains.

3 Description of the Corpus

The object of our study is the corpus created for the QALL-ME project, composed by 4500 questions in Spanish, with an English parallel version about the touristic domain. It is the result of the collection of many sentences recorded by a large number of speakers. Every speaker has performed 30 questions based on 15 real scenarios and for each of them, two questions are generated. This collection of questions has been created to have a sample of real natural language, and for this reason speakers had to think about real needs and real situations. Every speaker is given a list of scenarios to be able to formulate the spoken queries to the system, using the telephone and then they will read a written question for the same scenario, that is composed by the following items:

1. **Sub Domain** identifies the context in which the query has to be posed; it could be "cinema", "restaurants", "events", etc.
2. **Desired Output** identifies the kind of information that you would like to obtain from the system (eg. How to get the cinema, the telephone number of a restaurant, the cost of a ticket, ...).
3. **Mandatory Items** are list of items. The speaker has to include all of them.
4. **Optional Items** are list of items: the speaker can put none, some, or all of them in the question.

We could define a realistic scenario as the set of instructions that allow a speaker to formulate useful questions, without providing too many suggestions related to each query; in our case, questions have been recorded, and then transcribed using the free tool Transcriber[8].

After the transcription process, the corpus has been translated into English, taking into account the main aim of the translation that is the simulation of

[7] http://www.getess.de/goal.html

[8] http://trans.sourceforge.net/en/presentation.php

situations in which a person is visiting a city and asks for touristic information in a natural way. The only elements that we did not translate are named entities.

After the translation, the corpus has been annotated to detect the speech acts that refer to the different communication purposes. According to [2], when we speak we are doing something with our words. In fact, the term "speech act" is the synonym of illocutionary act; when a minister joins two people in marriage he says: "*I now pronounce you husband and wife*". As we can understand, the words pronounced by the minister have effect in the reality. Thus, the queries of our corpus has been created thinking about a real need, and each of them should have the purpose to generate an effect; in our case the effect is the answer. For this reason, we group the queries in request and non request. The first group can be direct or indirect and the second can be greetings, thanks, asserts or other. The corpus contains queries such as: *"At what time can I watch the movie el Ilusionista at the Ábaco 3D cinema?"*, *"What is the price of a double room at Vista Blanca hotel?"*, etc.

After having realized the aforementioned steps, the EAT has been annotated using Sekine's ontology that is open domain; as a consequence it provides a description of the world, but we will focus on the part of the world we are working with. Moreover, we would like to explain that we annotated our corpus according to the definitions of the labels provided by Sekine, adapting them to the needs of the project. We could use this corpus in open domain QA systems such as the ones that participate in competitions like TREC [19], CLEF [6], etc.

Finally, the last step consisted in annotating the corpus using the restricted domain QALL-ME ontology that is specific, and complete for the touristic area and created *ad hoc* for the needs of the project.

4 Description of the Two Ontologies

The Sekine's Extended Named Entity Hierarchy originates from the first Named Entity set defined by MUC [7], the Named Entity set developed by IREX [14], and the Extended Named Entity hierarchy which contains approximately 150 NE types [15].

This ontology is divided into top level classes that are name, time, and numerical expressions. Starting from these three classes at the top of the Extended Named Entity Hierarchy we can find the others.

By contrast, the QALL-ME ontology [12] was created after a deep research on the previous ontologies (described in section 2) and, as a consequence, it borrows some concepts and structures from them. Regarding its coverage, it is similar to the *Harmonise* and *eTourism* ontologies because they all focus on static tourism information rather than dynamic. In figure 1, a part of the QALL-ME ontology, concerning cinema and movies, is shown.

The ontology provides a conceptualized description of the touristic domain. Moreover, it covers the most important aspects of the tourism industry, including tourism destinations, cities or events. It consists of 122 classes, 55 datatype

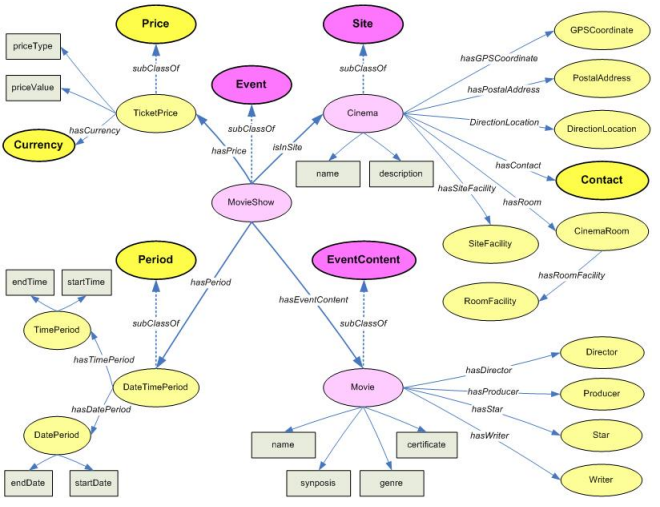


Fig. 1. Part of the QALL-ME ontology (cinema/movies)

properties and 52 object properties with the function of indicating the relationships among the 122 classes, divided into 15 top-level classes.

The structure of the QALL-ME ontology is similar to the *eTourism* ontology; in fact, both of them are written in the Web Ontology Language (OWL[9]), they can involve more complex classes and relationships, and support complex inferences.

4.1 Annotation

Tagging questions with their EAT needs an exhaustive definition of a hierarchy of possible answer types, where question EATs will be matched. We can find different general answer type taxonomies, employed for open-domain question answering, but they cannot be employed in specialized domains due to its high abstraction level.

The EAT of a question could be defined as the class of object sought by the question. In the QALL-ME project, we have to perform EAT tagging over a restricted domain modeled by an ontology. In fact, the QALL-ME system is considered a restricted domain question answering system, due to the fact that it fulfils the main characteristics of these kinds of systems [11].

The size of the corpus is limited, it contains a low redundancy level, and the domain of application is described and modeled with precision. Using the EAT classification, that is an essential process in QA, the annotator assigns a predefined class or category to the answer, and the subsequent extraction process

[9] OWL is a family of knowledge representation languages for authoring ontologies, and is endorsed by the World Wide Web Consortium.
http://www.w3.org/TR/owl-features/

depends on this previous classification. In order to create a coherent annotation, it is also fundamental to fix guidelines in order to properly annotate the corpus. One of the most important rule for our annotation is that we use ontology concepts (classes) as EAT as a default option.

Concepts in ontologies are organized hierarchically and, as a consequence, the most specific are included into the general ones. As an example, the concepts *fax*, *telephone* or *email* are part of the more general class *Contact*.

This structure causes the problem of deciding which is the best level to be used for EAT tagging, also because there is a wide range of ambiguous questions.

In general, we expect the annotation to be as much informative as possible, and therefore we will always assign the most specific concept of the ontology, when it is possible. However, we must pay attention because using very specific concepts may cause errors in the corpus when annotating more general questions. Moreover, there are cases in which a speaker asks for more that one thing, formulating a complex question. In this case we allow multiple EAT in the same question, and regarding the tagging purposes a tag for any of EAT in a question is added. Finally, when a query requires information not explicitly defined as a class but as a datatype property, question EAT will be expressed as a datatype concept where this datatype property takes values from.

5 Problems of Annotation

The aim of this section is to present a sample of each annotation difficulties we find out during the corpus annotations, and to present our solution.

As we mentioned in section 3, the corpus object of our study represents a sample of real language and, as a consequence, ambiguous questions are frequent. Moreover, annotated corpora should fulfil a fundamental requirement; they have to be coherent. As a consequence, when annotating general criteria need to be fixed. These decisions are essential, since they represent the pillar of the global annotation.

The main problem of our annotation is the coexistence of the two ontologies. In fact, Sekine's ontology is open domain, and this feature represents the first obstacle for annotators; the labeling of a restricted domain corpus with a general ontology can be a very complex process because Sekine describes the world in general and, as a consequence, we cannot find specific classes about the touristic domain, as for example the cuisine of a restaurant. We can only find out classes of the touristic domain that are very general and not specific for the domain we are analyzing. If we have a look at the Sekine's ontology, we can find the class *MONEY*, but nothing related with the price of a guest room in a hotel.

After having performed the annotation with Sekine's taxonomy, we had to start annotating using the ontology that it is extremely specific, and this characteristic represents another problem. In fact, the tendency of the annotator is to be as much specific as possible and this could generate the risk to be extremely specific, a negative attitude for the needs of the QALL-ME project. In other

words, the final system should provide the user as much information as possible, avoiding him to call many times in order to obtain the information he needs.

In the following paragraphs, we will provide an exhaustive description, and explanation of the criteria we adopted in order to propose a viable solution thinking on the needs of the QALL-ME project.

- *Tell me the address of the Meliá hotel in Alicante.*
  In this case we use the class *ADDRESS* for Sekine, and the class *Postal-Address* for the ontology. This is not a problematic query, but when it is compared with the following one, it could be ambiguous.
- *What is the street of the Heperia hotel in Alicante?*
  This is a different question, because, more specific than the previous example; if the user asks for the address, we use the class of the ontology *Postal Address*, while in this case we can not use the same class; the attribute *.street* has to be added in order to supply the information required by the user.
- *How can I get in touch with the Amerigo hotel?*
  This is a very general query, because the user could ask for the telephone number, the fax number or the email address of the hotel he is looking for. As a consequence, we do not have problems with Sekine because general is better for this ontology and we put the label ADDRESS. The problem we have to solve is to find the correct class that includes address, telephone, fax, mail website into the ontology; this class is *Contact*.
- *When does the pharmacy at calle Alfonso el Sabio opens?*
  In this question the speaker may want to know the day, the opening hours or both of them; for this reason we select *DateTimePeriod* in the QALL-ME ontology and *TIMEX* that are the two best classes for providing the user with all the information he needs.
- *Tell me the timetable of La Tagliatella restaurant*
  This question is also ambiguous. The timetable could be the time, the day or both of them; the solution is the same adopted for the aforementioned example. We select *TIMEX* for the Sekine's ontology, and *DateTimePeriod* for the QALL-ME ontology.
- *Tell me the name of a cinema, restaurant, etc.*
  We can not find a class specific for these queries in Sekine, and our option is to put the general class GOE-OTHER[10]; in the case of the annotation using the ontology we do not have problems.
- *Tell me the ticket price of the Panoramis cinema*
  This question can be interpreted in different ways; the fist one is that the speaker requires for the value, but it could ask for the type of the price to check if a discount is available or both options. In this case we are forced to select the class *MONEY* in Sekine and in the ontology to choose for *TicketPrice* that includes the attributes *.priceType* and *.priveValue*.

[10] GOE-OTHER is a class that indicates public facilities. School, Institution, Market, Museum, etc. are also included into this group. When the annotator cannot use one of these classes, it should select GOE-OTHER.

– *Does the price of the Bahía hotel include breakfast?*
We could see the breakfast as a facility or as a kind of price; we decided to see it as a kind of price that can be also the amount of money. For this reason we select *MONEY* in Sekine, and *GuestRoomPrice* for the annotation using the ontology.

6 Evaluation

In order to test the consistency of the labeled corpus, we performed two different evaluations. First, we employed the corpus to train and test a question classification system. Secondly, we calculated the kappa agreement between the assessors that labeled the corpus.

6.1 Black-Box Evaluation

In this first experiment, we performed a black-box type evaluation. We employed our corpus to train and test an SVM-based question classification system. Support Vector Machines (SVM) [18] have demonstrated to perform the state-of-the-art in the task of question classification [10]. In these experiments, we employed a lineal kernel and a *bag-of-words* representation of the feature space.

In order to evaluate the two sets of labels in English and Spanish, we carried out four different tests. We performed a 10-fold cross validation to test the system. Table 1 shows the results obtained.

Table 1. Question classification performance for English and Spanish

Language	Sekine	QALL-ME
English	95,18%	94,36%
Spanish	95,51%	95,04%

These results are considerably high for all the experiments. We can compare our results with those obtained with one of the most widely used corpus in the task of question classification, previously described in [9]. This English corpus consist in almost 5,500 questions for training and 500 for testing. These questions are labeled with a two level hierarchy of 6 coarse– and 50 fine–grained classes. In [20], they employed this corpus to train the same classifier that we used in our experiments, obtaining 85,8% precision for coarse-grained classes and 80,2% for fine–grained. When compared with these results, our corpus demonstrates to be a coherent and robust resource for the task of question classification.

6.2 Inter-annotator Agreement

The corpus developed in this work was labeled by two annotators. The kappa agreement obtained by these annotators in the ontology of Sekine was 0.87, while the agreement in the QALL-ME ontology was 0.89. These values were computed

according to [5], taken as equal for the coders the distribution of proportions over the categories.

In both cases, we obtained a substantial agreement. This agreement is higher for the QALL-ME ontology. This reflects the fact that the corpus was gathered thinking in the QALL-ME restricted domain ontology, and thus this labels can be naturally assigned to the questions in the corpus.

7 Conclusion and Future Work

In this paper, we have presented a corpus created under the QALL-ME project framework with the aim of training QA systems. The corpus consists of 4500 Spanish and English touristic domain questions which were annotated according to two different ontologies: an open domain, and a close domain ontology. Another contribution of our research is the presented solutions that focus on harmonizing the diferencies between the two ontologies in order to obtain a valid annotation. Thus, this corpus allows training a question answering system for both open and restricted domain purposes.

In order to evaluate the coherence of this resource, we have performed a double test and, on one hand, we have evaluated the inter-annotator agreement calculating the kappa measure. On the other hand, we performed the evaluation of the annotation using a question classification system. We have obtained considerably positive results for both test, demonstrating the coherence of the annotation process. Finally, as a future work proposal, our intention is to extend our work to other languages in order to train Cross-Lingual QA systems.

References

1. Agichtein, E., Lawrence, S., Gravano, L.: Learning search engine specific query transformations for question answering. In: Proceedings of the 10th World Wide Web Conference (WWW 10) (2001)
2. Austin, J.: How to do things with words. In: CPaperback, 2nd edn. Harvard University Press (2005)
3. Berger, A., Caruana, R., Cohn, D., Freitag, D., Mittal, V.: Bridging the lexical chasm: statistical approaches to answer-finding. Research and Development in Information Retrieval, 192–199 (2000)
4. Burke, R., Hammond, K., Kulyukin, V., Lytinen, S., Tomuro, N., Schoenberg, S.: Question answering from frequently-asked question files: Experiences with the faq finder system. AI Magazine 18(2), 57–66 (1997)
5. Fleiss, J.L.: Measuring nominal scale agreement among many raters. Psychological Bulletin 76(5), 378–382 (1971)
6. Giampiccolo, D., Forner, P., Herrera, J., Peñas, A., Ayache, C., Forascu, C., Jijkoun, V., Osenova, P., Rocha, P., Sacaleanu, B., Sutcliffe, R.F.E.: Overview of the clef 2007 multilingual question answering track. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152, pp. 200–236. Springer, Heidelberg (2008)
7. Grishman, R., Sundheim, B.: Message understanding conference- 6: A brief history. In: COLING, pp. 466–471 (1996)

8. Klettke, M., Bietz, M., Bruder, I., Heuer, A., Priebe, D., Neumann, G., Becker, M., Bedersdorfer, J., Uszkoreit, H., Maedche, A., Staab, S., Studer, R.: Getess - ontologien, objektrelationale datenbanken und textanalyse als bausteine einer semantischen suchmaschine. Datenbank-Spektrum 1, 14–24 (2001)
9. Li, X., Roth, D.: Learning question classifiers. In: Proceedings of the 19th international conference on Computational linguistics, Morristown, NJ, USA, pp. 1–7. Association for Computational Linguistics (2002)
10. Metzler, D., Croft, W.B.: Analysis of statistical question classification for fact-based questions. Information Retrieval 8(3), 481–504 (2005)
11. Mollá, D., Vicedo, J.L.: Question answering in restricted domains: An overview. Computational Linguistic 33(1), 41–61 (2008)
12. Ou, S., Pekar, V., Orasan, C., Spurk, C., Negri, M.: Development and alignment of a domain-specific ontology for question answering. In European Language Resources Association (ELRA) (ed.) Proceedings of the Sixth International Language Resources and Evaluation (LREC 2008), Marrakech, Morocco (May 2008)
13. Ravichandran, D., Ittycheriah, A., Roukos, S.: Automatic derivation of surface text patterns for a maximum entropy based question answering system. In: Proceedings of the HLT-NAACL Conference (2003)
14. Sekine, S., Isahara, H.: Irex: Ir and ie evaluation project in japanese. In: European Language Resources Association (ELRA) (ed.) Proceedings of the Sixth International Language Resources and Evaluation (LREC 2000), Athens, Greece (May-June 2000)
15. Sekine, S., Sudo, K., Nobata, C.: Extended named entity hierarchy. In: European Language Resources Association (ELRA) (ed.) Proceedings of the Sixth International Language Resources and Evaluation (LREC 2002), Las Palmas, Spain (March 2002)
16. Soricut, R., Brill, E.: Automatic question answering: Beyond the factoid. In: Proceedings of the HLT-NAACL Conference (2004)
17. Staab, S., Braun, C., Bruder, I., Düsterhöft, A., Heuer, A., Klettke, M., Neumann, G., Prager, B., Pretzel, J., Schnurr, H.-P., Studer, R., Uszkoreit, H., Wrenger, B.: Getess - searching the web exploiting german texts. In: Klusch, M., Shehory, O., Weiss, G. (eds.) CIA 1999. LNCS, vol. 1652, pp. 113–124. Springer, Heidelberg (1999)
18. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
19. Voorhees, E.M.: Overview of trec 2007. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152. Springer, Heidelberg (2008)
20. Zhang, D., Lee, W.S.: Question classification using support vector machines. In: SIGIR 2003: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, pp. 26–32. ACM, New York (2003)

Language Identification on the Web: Extending the Dictionary Method

Radim Řehůřek[1] and Milan Kolkus[2]

[1] Masaryk University in Brno
xrehurek@fi.muni.cz
[2] Seznam.cz, a.s.
milan.kolkus@firma.seznam.cz

Abstract. Automated language identification of written text is a well-established research domain that has received considerable attention in the past. By now, efficient and effective algorithms based on character n-grams are in use, mainly with identification based on Markov models or on character n-gram profiles. In this paper we investigate the limitations of these approaches when applied to real-world web pages. The challenges to be overcome include language identification on very short texts, correctly handling texts of unknown language and texts comprised of multiple languages. We propose and evaluate a new method, which constructs language models based on word relevance and addresses these limitations. We also extend our method to allow us to efficiently and automatically segment the input text into blocks of individual languages, in case of multiple-language documents.

1 Motivation

The amount of information available on the net is staggering and still growing at a fast pace. To make this information available, applications have sprung up to fill the void and gather, process and present Web information to the knowledge-hungry user. Unfortunatelly, documents on the Web have historically been created with human reader in mind, in formats such as HTML, and are not readily understandable by computers. Although XML and semantic markup (e.g. the `xml:lang` attribute, or the `<div lang="en">` construct) have been introduced to alleviate these problems, reality remains that many documents do not make use of metadata tags or, even worse, make use of them incorrectly and provide misleading information.

By not having metadata provided for us, or by deciding not to trust it, we are left with deducing information from the text itself. This is the domain of natural language processing (NLP) and text mining. This article deals with one aspect of text mining, namely telling which language (or languages) is a given Web page written in.

2 Related Work

A general paradigm in automated language identification is to create language models during a training phase and compare input document against these

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 357–368, 2009.

models during language identification. This places the task into the domain of *supervised learning* methods. Another consequence is that the set of target languages needs to be known beforehand, which makes language identification a *classification* problem.

A "common words" approach [1] is based on the observation that for each language, there is small class of words that carry little information but make up a large portion of any text. These are called *function words* or *stop words* and their presence is to be expected as word distribution follows Zipf's law.

In [2] it is noted that humans need surprisingly little in order to correctly identify a language. Interestingly, this is the case even if they are not proficient in that language or when the text snippet is quite short. This observation leads to a class of algorithms based on character (or even byte) n-grams, as opposed to more linguistically refined syntactic or semantic methods.

- A popular tool called *textcat* [3] constructs a ranking of the most frequent character n-grams for each language during the training phase and proclaims this ranking the *language model*. For classification, a ranking is constructed for the input document in the same fashion and is compared against each available language model. The closest model (in terms of ranking distances, see [3] for details) wins and is returned as the identified language.
- Another character n-gram approach pioneered by [2] computes likelihood of generating the observed character sequence explicitly, through use of higher order Markov models. Let S be a sequence which consists of n characters $(s_1, \ldots, s_n)$. Then the probability this sequence was generated by Markov model L of order k is given by

$$P(S \mid L) = p(s_1, \ldots, s_k \mid L) \prod_{i=k}^{n} p(s_{i+1} \mid s_{i-k+1} \ldots s_i, L),$$

  where the first factor is the initial state distribution and the conditional probability describes transitions. Training the language model consists of estimating these transition probabilities. Again, winner is the language with the best likelihood of generating the input text. It is observed that using character trigrams, i.e. Markov models of order 2, already gives optimal results and increasing the model order therefore cannot affect performace much. For a comparison of character trigrams to "common words", see [4].
- A related approach makes use of Shannon's information theory and compares language entropies [5]. Here Markov models are also estimated based on training data. In contrast to [2], all models of orders $0, \ldots, k$ are used and their relationship explicitly modeled. This allows the algorithm to fall back to lower order models in case of insufficient data through mechanism called *escape probabilities*. Decision function views input as a stream of characters and in accordance with information theory tries to predict the next character in the stream. Success of these predictions is measured by cross-entropy and the model with the lowest cross-entropy after having processed the whole stream wins. Because of its ties to information theory and language compression, this technique is sometimes called the *compression* technique.

Apart from individual algorithms, research into language recognition has also identified key factors which directly influence performance:

- **Size of training data.** Methods are evaluated based on how quickly their models converge, given differing sizes of training corpora. Note that more is not necessarily better here, as there is a risk of *overtraining* or *overfitting* the training data.
- **Size of input text.** Methods can be distinguished by how much text they need to be given in order to reliably identify its language. Amount of text can be roughly divided into *small* (a phrase, less than 30 characters or up to 5 words), *large* (a paragraph, more than 300 characters or 50 words) and *medium* (a sentence, in between).

3 Proposed Method

Motivation for Change
Summing up the previously mentioned articles, there are several reasons behind the success of language modelling via character n-grams:

- **Fast convergence.** Very small training corpora (in the order of hundreds of kilobytes of text) are required to learn the models. See e.g. [6] for a study on speed of model convergence for character bigrams and trigrams.
- **Robust.** As long as the overall letter distribution in input document follows that of training examples, problematic language phenomena such as neologisms (words newly introduced into the language), spelling errors, rare inflections or unknown words are handled gracefully.
- **Domain independent.** In [2] this approach was applied to a domain as distant as that of genetic sequence identification. Another often highlighted feature is that character n-gram methods do not require tokenization of the input, making them also suitable for Asian languages where tokenization is an interesting challenge in itself.

We implemented, for some time used and then evaluated an algorithm based on the compression technique [5]. We estimated all i-gram distributions for $i = 0, \ldots, n$ and then combined them through an *Expectation Maximization* (EM) smoothing algorithm on held-out data. We were interested in detecting nine European languages: French (*fr*), English (*en*), Italian (*it*), Spanish (*es*), Slovakian (*sk*), Czech (*cs*), Slovenian (*sl*) and Polish (*pl*). Although this method worked almost perfect on our test data, a number of real-world issues soon became apparent when applied to the task of Web page classification. The main problem was not insufficient accuracy of classification as such, but rather a shift in the formulation of the language identification problem:

- **No *unknown language* option.** All methods listed above based on entropy, Markov processes or n-gram profiles return the nearest, best-fitting language. They assume that a) the set of languages is complete, known and trained for beforehand and b) that the input text is in exactly one of them. While the first assumption can be dismissed as our own decision, the latter is unrealistic for the Web.

- **Multiple languages.** In an also rather frequent scenario, there are parts of the input document which are in different languages. This may stem from a page's logical structuring (menu, text body, copyright notices) but also from the nature of the text body itself. This moves the document away from any one model and the language models become mixed in undesired ways. As a result, the document may even be identified as a completely unrelated language not present in the input text at all. In our experience, multilingual documents somehow often ended up being marked as Slovenian.
- **Close languages (same language family).** As seen above, our language set includes Slovenian, Slovakian, Czech and Polish, which are all Slavic languages with considerable grammatical as well as lexical overlap. This is exacerbated by the fact that real texts on the Web often come in deaccented version, so that the trained models are unable to even theoretically take advantage of otherwise telling national characters ($\check{r}$ for Czech, $\acute{l}$ for Slovak etc.).

As a special case of the second point, there are many pages where letter distribution is heavily skewed by repetition of certain words or phrases. This includes discussion lists with `In reply to:` fields and so on. This problem does not come up in well-behaved corpora, but quickly becomes a nuisance when dealing with the Web.

To address the first two issues, we tried augmenting our implementation of the n-gram algorithm. We looked for a minimum threshold for each language that a document score has to exceed in order to be identified as that particular language, even if its score is the best available. Note that for language detection, document length is not an issue, as all models are evaluated on the same number of n-grams and the score numbers are thus directly comparable. For the fixed threshold to work, however, the scores need to be normalized to negate the effect of varying document lengths, as adding even one n-gram changes the order of magnitude of the probability scores.

Although we tried setting this threshold automatically, based on held-out training data, the results were not satisfactory. It appears that the per-character cross-entropies are dependent on the contents of text n-grams in a way that prohibits direct absolute comparison against any fixed threshold. In other words, it proved impossible to find a threshold that would allow us to tell "this best fitting language is in fact an error". We also tried learning a special *unknown language* model from a hotch-potch of documents in various random languages. This worked better and solved the Slovenian classification problem, but seems rather ad-hoc and theoretically unfounded.

To avoid headache of further complicating an already complex algorithm, we set out to try a different approach.

Dictionary Method

In [2], dictionary methods (i.e. methods based on words rather than characters) are discussed and dismissed, based on their best known representative, "common words", being too restrictive and only applicable to longer texts.

Going through the list of n-gram advantages, the benefits of broad domain independence, no required tokenization and fast model convergence will indeed have to go. Since our goal is to tell apart European (Latin character based) natural languages, the first two are not really a concern. The last one, small amount of training examples required, was perhaps an asset back when these methods were developed. In the present day, with Web as Corpus [7] projects and NLP advancements, fast indexing and retrieval techniques, the amount of available data is no longer a critical issue. The same cannot be said for runtime performance, as the same reason why there are many documents requires us to process them at increased speed. For these reasons we decided to revisit the dictionary method.

We take a qualitatively different approach to constructing the dictionary language models. Rather than looking for words that are common in a given language (called *function* or *stop* words), we note which words are *specific* for a language, or rather, *how specific* they are. The foundation of our algorithm is a relevance mapping

$$rel(word, language) : W \times L \mapsto \mathbb{R}$$

where W is a set of all words present in the training data and L the set of considered languages. We call the real-valued score of word $w \in W$ in a language $l \in L$ its *relevance*. In other words, the mapping is not binary as in the case of the "common words" approach, but rather a soft grading of words. Positive relevance hints at the word being indicative of the language, relevance around zero naturally corresponds to "no correlation" and negative values to "this word is indicative of *absence* of the language". We will call these *positive*, *near-zero* and *negative* evidence, respectively.

Naturally, the relevance mapping is constructed automatically from labeled training data. In contrast to character n-gram models, the convergence is much slower and significantly larger training corpura are required. We estimate the word relevance using reasoning detailed in [8]. Their idea, although developed for classifying documents into topics, can also be applied to our problem of language identification. Below we give a short overview of the main assumptions and steps behind derivation of the final relevance formula; for a more thorough discussion on various aspects, please see the original article [8].

We start by noting frequencies of words $w_1, w_2, \ldots, w_N$ within each language corpus and compare them to frequencies in a general, *background* corpus. In this context, a corpus C is simply a collection of D documents, $C = (d_1, d_2, \ldots, d_D)$. For each language $lang$, we have a corpus C_{lang} of documents only in that language, plus one general background corpus which represents a collection of documents of background language $lang_0$. This background language is ideally completely language neutral, or more realistically represents the distribution of all languages on the Web. To approximate $lang_0$, we consider the union of all language corpora to be the background corpus, $C_0 = \bigcup C_{lang}$. The *uncorrected observed frequency* of word w in language $lang$ is then

$$\bar{g}_{lang}(w) = \frac{TF(w, C_{lang})}{\#(C_{lang})}, \tag{1}$$

with $\#(C)$ being the total number of words in corpus C and TF the number of occurences of a word in a corpus, and

$$g_0(w) = \frac{TF(w, C_0)}{\#(C_0)} \tag{2}$$

for the background language.

From the assumption of languages being modelled as Bernoulli (word unigram) sources, the probability that a document d that contains f_i instances of word w_i was produced by language $lang$ is given by the multinomial

$$P(d \mid lang) = \binom{f_0 + f_1 + \cdots + f_N}{f_0, f_1, \ldots, f_N} \prod_{i=0}^{N} g_{lang}(w_i)^{f_i}. \tag{3}$$

To avoid singularities for zero frequencies, the *Jelinek-Mercer smoothing* correction is introduced

$$g_{lang}(w) = \alpha g_0(w) + (1 - \alpha)\bar{g}_{lang}(w) \tag{4}$$

for some small value of α, such as 0.1.

After substituting (4) into (3), we compute logarithm of probability ratio that a document was emitted by $lang$ rather the background language $lang_0$ by

$$log \frac{P(d \mid lang)}{P(d \mid lang_0)} = \sum_{i=0}^{N} f_i log \frac{\alpha g_0(w_i) + (1 - \alpha) g_{lang}(w_i)}{g_0(w_i)} \tag{5}$$

An interesting observation the authors present is that negative and near-zero evidence contributes very little to classification accuracy. In fact, according to [8], accuracy actually *improves* when near-zero and negative evidence is purposefully omitted. Translated to our language identification problem, we only pay attention to words that are highly indicative of the given language, disregarding near-zero and negative evidence entries. This has the pleasant side-effect of keeping the models reasonably sized, despite there being virtually tens of millions of possible words in each language relevance mapping. With this simplification and some minor mathematical tricks (see [8]) the formula becomes an elegant and manageable

$$\sum_{g_L(w_i) \ll g_{lang}(w_i)} f_i \cdot rel(w_i, lang), \tag{6}$$

where $rel(w, lang) = log(g_{lang}(w)) - log(g_0(w))$ is our desired relevance of word w in language $lang$. Put in words, the relevance of a word measures the orders of magnitude by which it is more frequent in the specific language corpus compared to the background corpus. This a surprisingly simple relationship, given we started only from the assumption of word independence (Bernoulli model). Another way to look at the formula is to realize that f_i corresponds to Term Frequency (TF) and $rel(w_i, lang)$ to a kind of Inverse Document Frequency (IDF) component, linking this result to the general framework of TF-IDF classifiers.

Obviously this is a sharp divergence from the idea of identifying languages by the most "common words".

With all pieces in place, how do we go on choosing which languages a sequence of words belongs to? According to the above derivation, we simply iterate over words that are distinctive of each language and sum their relevancies. We may compare this value to a threshold to immediately see if there was enough evidence to proclaim the document d as belonging to language *lang*. But to abstract from document length, we first divide this sum by the length of the document in words, that is, we take average of the individual word relevancies. The final decision function which identifies document d as coming from language *lang* is then

$$score(d, lang) = \frac{\sum_{g_L(w_i) \ll g_{lang}(w_i)} f_i \cdot rel(w_i, lang)}{\sum f_i} \geq t_{lang}. \tag{7}$$

The threshold t_{lang} is found, for each language separately, by optimizing an objective function on held-out data. One candidate for objective function is the F_1 measure, the generalized formula of which is

$$F_\beta = (1 + \beta^2) \frac{precision \cdot recall}{\beta^2 \cdot precision + recall}.$$

F_1 measure is popular in Information Retrieval and defines an equal trade-off between precision and recall. Other objective functions are possible, depending on the desired application of language identification. If the cost of not identifying the right language (*false negative*) is higher than cost of erroneously identifying an unwanted language (*false positive*), higher preference should be given to recall (e.g. via the F_2 measure) and vice versa. This effectively lowers (resp. raises) the estimated language threshold.

Contrary to results obtained from using thresholds for character n-gram method, detecting unknown language works quite reliably (see Evaluation section). Because some words may be indicative of several languages (such as the previously mentioned lexical intersection of Slavic languages), more than one language may be recognized, too.

Practical Considerations

As noted earlier, runtime performance of classification is important. Interpreting equation (7), the algorithm consists of tokenizing input text, averaging token relevancies and comparing this sum to a precomputed threshold. This can be done extremely fast, using any of the many commonly available data structures which map strings into numbers.

As for memory considerations, the mapping that needs to be stored is in fact very sparse, consisting of only those words which are distinctive for a language. In fact, the size of each language model when stored as *Patricia trie* [9] was in the tens of megabytes, which is comparable to size of our character pentagram models. This is not surprising as character pentagrams already come close in length to whole words.

We solved the practical question of obtaining large and representative language corpora by using Wikipedia dumps [10]. As research into Web corpora [7] rapidly progresses, it can be expected that assembling large text collections will become less and less of a problem in the future. It must be kept in mind however that common NLP techniques like stemming or lemmatization may not be applied, as these are dependent on language—the very thing we don't know and want to determine in the first place.

Evaluation

To evaluate our algorithm, we trained it on Wikipedia dumps [10] of the nine target languages. As a reminder, these are French (*fr*), English (*en*), Italian (*it*), Spanish (*es*), Slovakian (*sk*), Czech (*cs*), Slovenian (*sl*) and Polish (*pl*). To avoid overfitting the training data, we discarded duplicate sentences and only used each sentence once in our corpus. Sentences with non-Latin (mostly Asian and Arabic) characters were also ignored. Some data statistics can be seen in Table 1, where the number of unique sentences corresponds to the size of training data. In the same table we also give final model statistics. We put aside three thousand sentences of differing lengths for each language, to be used as test data. These were divided into *small*, *medium* and *large* sub-corpora (with texts of 2–5 words, 6–50 words and over 50 words, respectively), so that each subcorpus contained exactly 1,000 texts. We manually checked the test corpora and estimated that the ratio of erroneously labeled examples is

- about 10% for medium length documents (mostly municipality and proper name enumerations),
- about 20% for long texts (same reason, plus many texts are in fact English phrases such as song or book titles)
- and as much as 50–70% for the short texts.

Short texts are especially bad because they concentrate "sentences" consisting of formulas, location entries, article headings with a person's name and lifetime and so on. All of these often refer to foreign institutions and have no connection to the language of the main article. Final sizes of test corpora after removing these problematic texts are given in Table 1.

Table 1. Overall data and model statistics

Language code	**Dump size [GB]**	**No. unique sentences [k]**	**No. test documents** small	medium	large	**Dictionary model size [words]**
cs	4.8	2,926	814	907	814	551,126
de	39.4	27,010	461	916	762	944,450
en	208.3	74,926	448	980	998	548,649
es	18.9	10,848	520	891	742	318,423
fr	39.8	18,048	483	852	765	373,432
it	26.0	11,529	469	836	727	378,817
pl	18.0	10,157	286	878	784	799,180
sk	3.3	1,769	275	916	768	474,003
sl	2.8	1,472	249	916	795	288,442

Table 2. Evaluation on test data

	n-gram method			dictionary method		
language	**text size**			**text size**		
code	small	medium	large	small	medium	large
cs	81.9/75.2	96.8/96.0	100.0/100.0	64.9/84.0	85.2/96.9	97.8/99.6
pl	84.1/67.6	97.4/96.5	97.9/97.2	82.9/90.2	95.5/97.0	96.9/97.5
sk	81.6/77.7	97.7/96.9	99.3/99.0	57.8/82.9	71.4/96.6	87.6/96.7
sl	89.3/79.7	97.8/97.2	99.3/99.2	68.6/88.2	91.9/97.2	98.8/99.0
it	81.9/58.4	98.7/96.4	99.9/99,8	78.6/88.1	95.8/98.0	99.4/99.7
fr	80.1/52.9	98.3/97.3	99.8/99.6	82.7/88.7	98.4/99.0	99.5/99.6
de	85.2/73.6	98.8/98.1	99.0/98.4	85.7/91.8	98.2/99.6	98.8/99.2
es	81.5/61.6	99.0/98.1	100.0/99.9	73.2/86.4	94.3/98.9	99.3/99.8
en	81.4/51.7	99.4/98.2	99.8/99.1	86.1/91.6	99.2/99.7	99.8/99.4

Precision/recall on test data, in percent.

Classification results are summarised in Table 2, which also includes results of our implementation of the cross-entropy based character *n*-gram algorithm described earlier, on the same data. Recall is measured as the ratio of true positives to all available positives (including false negatives), precision is the number of true positives divided by the number of all positives returned (including false positives). Note that this gives more room for precision errors to the dictionary method, which can return multiple false positives for each document, unlike the *n*-gram method that returns at most one incorrect language per document.

Inspection of results reveals that the errors closely follow the data problems described above. On one hand this is vexing because it prohibits more exact evaluation. On the other hand it shows that despite the considerable amount of noise in training data (which obviously shares the same problems as the test data) and in face of contradictory information, the classifiers are able to generalize. However, we'd like to stress the fact that our goal here is not to discuss the absolute numbers, but rather to juxtapose and compare two language identification methods on the same dataset.

To confirm our hypothesis of poor data quality, we manually checked labels of all Czech and English test examples. The labeling error was about 1% for medium and large texts and about 40% for texts of small length. We expect this error to be similar for the seven remaining languages, too. We re-ran language identification experiments on the two manually pruned corpora, with results

Table 3. Evaluation on pruned test data

	n-gram method			dictionary method		
language	**text size**			**text size**		
code	small	medium	large	small	medium	large
cs	93.5/92.4	98.7/98.7	100.0/100.0	73.9/99.8	86.7/100.0	98.1/100.0
en	93.1/67.3	99.7/98.6	100.0/100.0	98.4/100.0	99.7/100.0	100.0/100.0

Precision/recall on pruned test data, in percent.

summarised in Table 3. Many short Czech documents are classified as both Czech and Slovak by the dictionary method, resulting in lower precision but still excellent recall.

We conclude that the numbers are sufficiently high (in fact, after discounting the test data noise, nearly optimal) for both algorithms. The main difference and actually the reason why we developed our dictionary method in the first place is the added value of being able to return a set of languages as identificaton result, including the elusive empty set.

4 Segmenting for Language

There is one item on our language identification wish-list that hasn't been covered yet: correct classification of documents that contain blocks from different languages. While our character n-gram method based on cross entropy returns a random language in this case, dictionary method returns an unknown language. Both results are wrong. A practical extension to either algorithm would ideally allow us to locate and identify all compact single-language blocks.

To our knowledge, the only attempt at language segmentation was made in [5]. The authors consider all possible combinations of language change at each character in the input text and measure the resulting entropy on such text blocks. Although they report brilliant results of 99.5 % accuracy on *character level*, the method misses the mark in terms of speed by several orders of magnitude. Even with advanced dynamic programming optimizations, it took tens of seconds to segment a text [5].

Here we describe and evaluate an algorithm that segments input text into monolingual blocks.

Let $S_{lang}(d) = (score(w_1, lang), \ldots, score(w_n, lang))$ be a sequence of individual unit scores (word relevancies or n-gram probabilities) for the n units in document d. We can view this sequence as a real-valued signal and use signal processing to smooth the signal, removing local extrema,

$$(smoothed)_i = fnc_{SIZE}(score(w_{i-SIZE}), \ldots, score(w_{i+SIZE})), \qquad (8)$$

for any language *lang*. We use *median* with sliding window size $SIZE = 2$ as the smoothing function while noting that there is a direct connection between the smoothing window size and robustness to short extra-lingual segments in text. These manifest themselves as sharp local valleys and correspond to proper nouns, typing errors and other text anomalies. Although strictly speaking they really are different from the surrounding text, our task is to identify coherent language blocks that are meaningful on discourse rather than token level.

Once we have smoothed signals for all available languages, we identify local minima in them. This gives us a first estimate of potential segment boundaries. The proposed segment boundaries are not final though—many of them correspond to local minima in between two segments of the same language. We remerge these back into a single segment. Note that in this way we prohibit having two consecutive segments of the same language, but we may still arrive

at segments with no language assigned to them. It is also possible to have a segment with more than one language. This means the text may have been written in either and is indistinguishable. This often occurs with shorter *cs/sk* passages and reflects real ambiguity of input.

Complexity of the whole procedure is linear in the number of words and languages, $O(|d| \times |L|)$.

Evaluation

To evaluate language segmentation, we constructed an artificial corpus. The corpus contains 1,000 documents, each one of them being a concatenation of 1 to 4 segments in different languages. The numbers were picked to somewhat mimick situation on the Web, with 4 languages in a single document as an extreme case. Language segments are pooled randomly from a collection of medium-length texts in that language (6 to 50 words).

We give this concatenation to our segmentation algorithm, with signal scores based on word relevancies, and mark down languages predicted for each token. This per-token evaluation records *success* each time a token was assigned precisely the one language that was expected, and $failure$ otherwise. Accuracy is then computed as $\#success/(\#success + \#failure)$. Note that assigning multiple languages or no language at all to a token always equals an error.

The algorithm misclassified 1,420 out of possible 49,943 words. This corresponds to 97,16% accuracy. In 603 cases, boundary was missed by one word, which is still an acceptable error for our purposes. Discounting these off-by-one boundary errors, accuracy climbs to 98,34%. Closer inspection of the 817 misses left shows that some of them come from English collocations like *grand theft auto* which are embedded inside non-English text segments and regrettably misclassified as English by the algorithm. The real accuracy is therefore probably slightly higher, depending on mode of application.

Although these results are lower than those reported in [5], the algorithm enjoys conceptual clarity and impressive runtime performance.

5 Conclusion

The article's main contribution is revisiting and re-evaluation of some of the assumptions made 15 years ago, when the domain of automated language identification was being shaped. It proposes a straightforward, fully automated method which learns a decision function from training data. The decision function is based on word relevancies and addresses some aching problems of popular character n-gram based methods, while retaining character n-gram's excellent accuracy and actually improving runtime efficiency. Another important benefit of using words instead of character n-grams is that the system is more open to human introspection, more predictable in ways of interpreting its results ("looking inside the box") or selectively changing its behaviour—something of considerable value in real systems.

A general segmentation algorithm is described which is based on the notion of language signal strength within the input document. The algorithm is evaluated to behave acceptably using word relevancies and solves the problem of language identification in multilingual documents.

Acknowledgements

This study has been partially supported by the grant LC536 of MŠMT ČR.

References

1. Ingle, N.: A Language Identification Table. Technical Translation International (1980)
2. Dunning, T.: Statistical Identification of Language (1994)
3. Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. In: Ann Arbor MI, pp. 161–175 (1994)
4. Grefenstette, G.: Comparing two language identification schemes. In: Proceedings of the 3rd International Conference on the Statistical Analysis of Textual Data (JADT 1995) (1995)
5. Teahan, W.: Text classification and segmentation using minimum cross-entropy. In: Proceeding of RIAO 2000, 6th International Conference Recherche d'Information Assistee par Ordinateur, Paris, FR, pp. 943–961 (2000)
6. Souter, C., Churcher, G., Hayes, J., Hughes, J., Johnson, S.: Natural Language Identification Using Corpus-Based Models. Hermes Journal of Linguistics 13, 183–203 (1994)
7. Kilgarriff, A.: Web as corpus. In: Proceedings of Corpus Linguistics 2001, pp. 342–344 (2001)
8. Kornai, A., et al.: Classifying the Hungarian Web. In: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics, Association for Computational Linguistics Morristown, NJ, USA, vol. 1, pp. 203–210 (2003)
9. Morrison, D.: PATRICIA – Practical Algorithm To Retrieve Information Coded in Alphanumeric. Journal of the ACM (JACM) 15(4), 514–534 (1968)
10. Wikimedia Foundation Project: Wikipedia Static HTML Dumps (June 2008), http://static.wikipedia.org/

Business Specific Online Information Extraction from German Websites

Yeong Su Lee and Michaela Geierhos

CIS, University of Munich, Germany

Abstract. This paper presents a system that uses the domain name of a German business website to locate its information pages (e.g. company profile, contact page, imprint) and then identifies business specific information. We therefore concentrate on the extraction of characteristic vocabulary like company names, addresses, contact details, CEOs, etc. Above all, we interpret the HTML structure of documents and analyze some contextual facts to transform the unstructured web pages into structured forms. Our approach is quite robust in variability of the DOM, upgradeable and keeps data up-to-date. The evaluation experiments show high efficiency of information access to the generated data. Hence, the developed technique is adaptive to non-German websites with slight language-specific modifications, and experimental results on real-life websites confirm the feasibility of the approach.

1 Introduction

With the expansion of the Web, the demand for targeted information extraction is continuously growing. There are many services on the Web providing industry sector information or performing job search tasks. For these purposes, the data used must be first manually collected and therefore features several sources of error, e.g. spelling mistakes, incomplete database entries, etc. Moreover, this process is extremely time-consuming and updating the data then requires a rollback of the full process. Automating these tasks will help to extract the business specific information quickly and maintain the data up-to-date.

The standard approach of business-related information retrieval disregards the relationship between the domain name and organization-specific content of a website, but concentrates on the structural aspect of company information [1]. Only a few studies restrict the information extraction task to certain domain names [2,3,4]. They extract company profiles by limiting their research on locating products and other features while analyzing the format of HTML tables for structured data and trying to find the phrase patterns for unstructured texts [2]. Others examine the presentation ontology for extracting organization-specific data such as contact details and product information concentrating on the differences in the presentation manner of formatted company profiles versus plain text profiles [3]. But company information extraction can also be extended to different resources and incorporates meta tags as well as plain texts and structured data [4].

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 369–381, 2009.

As the Web keeps evolving, of course, every new website will uncover new ways that people encode the information. That way, other scientists concentrate on linguistic analysis of web pages and disregard the main characteristic advantage of the HTML structure. They investigate, for example, information extraction techniques for company details and job offers on the Web. These methods consider the relevance of the domain name, but only exploit the local characteristics of the text [5]. They therefore process in two steps: first HTML stripping and then applying local grammars [6] (recursive transition networks) on plain texts to transform unstructured web pages into structured forms. Manually encoding morphosyntactic rules for extracting the information seems doomed to be a never-ending process, but evaluation experiments show high values of precision and recall.

Our starting point of a solution is the structured nature of data. In contrast to a general search scenario, company search can be seen as a slot-filling process. The indexing task is then to detect attribute-value pairs in the HTML documents and make them accessible. At this point, we are interested in the extraction of all organization-specific data being related to the website's domain name (secondary level domain). Obligatory elements, such as the company name combined with a highly restrictive domain vocabulary, make it possible to discover the logic of an information page that can then be integrated into a relational structure. As our studies during this research were limited to the German Web, the investigated language was German.

The paper is structured as follows. In the next section we introduce the concepts and terms used in the paper. Section 3 presents an overview of the system architecture. In Section 4 the analysis of the information page is further detailed and Section 5 evaluates the performance of the system and shows promising results of precision (99.1%) and recall (91.3%). The conclusion comments on practical implications of the given approach and the directions of future work.

2 Definition of Terms

Terms that are used throughout this paper in various contexts and that have a particular usage have to be clearly defined.

2.1 Business Specific Information

Business specific IE differs from the record extraction or entity recognition because the information must be examined with respect to the domain name and estimated how valuable it may be.

Definition 1 (Business specific information). *Business specific information contains the relational facts concerning the domain name.*

In order to illustrate what kind of information is relevant according to the domain name, one information page is shown in Figure 1. The left section contains the navigation bar, the right one a shopping cart and advertisements, and the center

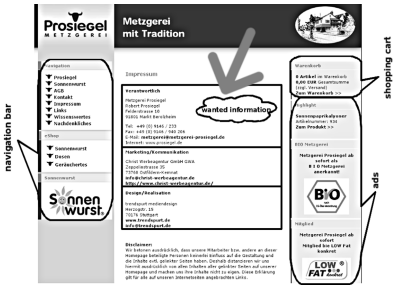


Fig. 1. Sample information page

Example of a company info form	
company name	Metzgerei Prosiegel
street	Felderstraße 10
zip code	91801
city	Markt Berolzheim
phone no.	(09146) 233
fax no.	(09146) 940206
email	metzgerei@metzgerei-prosiegel.de

Fig. 2. Business specific data of Fig. 1

is divided into three information records: The first contains the domain relevant information we are interested in. The second also appears somehow relevant but is about specialized marketing and the third names the web designer.

Definition 2 (Business specific IE). *Business specific information extraction is concerned with the automatic extraction of the relation between a domain name and an information set consisting of attribute-value pairs.*

2.2 Minimal Data Region

A group of data records that contains descriptions of a set of similar objects are typically presented in a particular region of a page (...) Such a region is called a ***data region****.* [7]

We can identify the region of the information bit with keywords or phrases heading the respective record. In our example (*cf.* Figure 1), the heading keyword for the relevant information is *"Verantwortlich" (responsible)*, for the marketing information it is *"Marketing/Kommunikation" (marketing/communication)*, and for the web designer record it is *"Design/Realisation" (design/ realization)*.

But we have to limit the data record containing information somehow focused on the domain name. In contrast to other approaches [7] we are not interested in locating data records of maximum length, we want to determine the "minimal data region" for an information bit (*cf.* Section 4).

Definition 3 (Minimal data region). *A minimal data region with respect to the business specific information is the smallest HTML tag region where most of the wanted information bits are located.*

2.3 Sublanguages on the Web

Definition 4 (Web sublanguage). *Sublanguages are specialized language subsets, which are distinguished by the special vocabulary and grammar from the general language [8,9]. With respect to the Web, a sublanguage is characterized by a certain number of phrases or a grammar and special vocabulary [10], e.g. "Impressum" (imprint).*

Table 1. Overview of attribute classes pertinent to business websites

Attribute Class	Quantity	Vocabulary
company name	99	*Anbieter, Firmenbezeichnung*
phone no.	25	*Fon, Tel, Tel + Fax*
fax no.	7	*Fax, Faxnummer, Telefax*
mobile no.	13	*mob, mobil, unterwegs*
email	16	*Mail, E-Mail, m@il*
CEO	23	*CEO, Geschäftsführer*
business owner	16	*Inh, Inhaber, owner*
contact person	10	*Ansprechpartner, Kontaktperson*
chairman	23	*chairman, Leiter, Vorsitzender*
management board	4	*Vorstand, Geschäftsführender Vorstand*
VAT ID	97	*UID, UST-ID-NR, Umsatzsteueridentnr.*
tax no.	25	*St. Nr., Steuernr, Umsatzsteuer Nr.*
register no.	22	*Handelsnr., Registernummer*
local court	28	*AG, Amtsgericht*
tax office	4	*FA, Finanzamt*

Web sublanguages occur on the home page of a website as well as on its information page. Regarding the home page we analyze the anchor texts that lead to the information page (*cf.* Figure 3). But the variety of organization-specific standard phrases (*frozen expressions*) that frequently emerge on information pages are clustered into *attribute classes* during the training step of our system. For instance, the class *"Provider"* contains about 140 specialized words and phrases (*attributes*), e.g. *"Anbieter i.S.d. TDG/MDStV" (Provider in terms of TDG/MDStV)* (*cf.* Table 1).

3 System Architecture

Figure 3 shows the elements of our system to extract business specific data from information pages of German websites. This process expects as input a set of URLs preclassified as business websites. The architecture is based on two interactive modules to establish a relational database storing company information and providing a query module:

A Localization of information pages on the Web
B Document analysis and information extraction
C Query processing

Our system `ACIET` (**A**utomatic **C**ompany **I**nformation **E**xtraction **T**ool) automates the extraction process of organization-specific information on the Web and works therefore in two steps:

In the first stage (**A**), a focused crawler is fed with URLs stored in a database and fetches the demanded websites. This step is performed by the "home page analyzer"[1]. Our system will follow the anchor tags leading to the information page and retrieve the document.

[1] For classification purposes, it can also extract the structural and textual features of a website by category. But at present we are only focused on the extraction process and suppose that our crawler input exclusively consists of business websites.

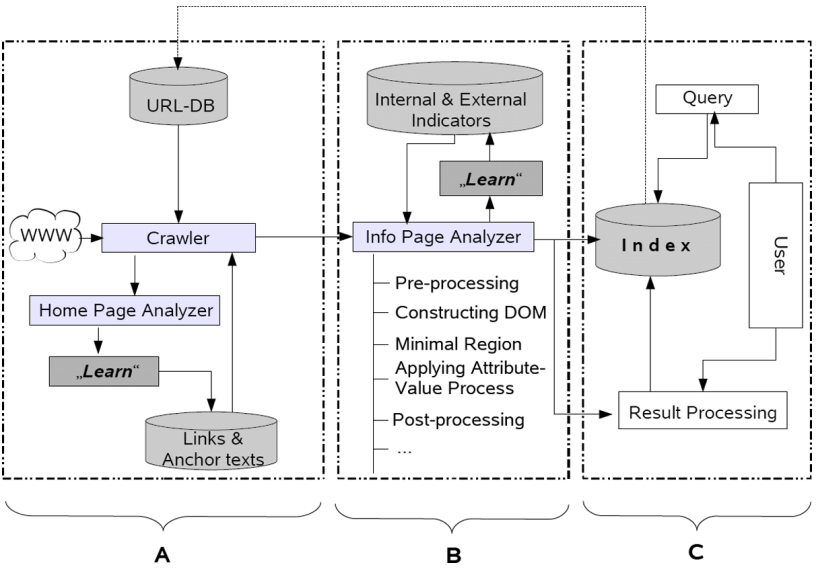


Fig. 3. Overview of the system architecture of *ACIET*

During the second stage (**B**), the information page is sent to a module called "info analyzer" to study the HTML content and extract the searched information bits. It thereby exploits the internal structure of named entities and uses sublanguage-specific contexts – attribute classes (*cf.* Section 2.3) to identify the attribute-value pairs. In difference to other systems the form filling process is fully automatized. From a document recognized as an information page by the system (part **A**) we extract all business specific information to fill a form that is presented in Table 2. For the transformation of the initial HTML-document into the form schema we need different operations shown in Figure 3 (part **B**).

An interaction by the user is provided in part **C** (*cf.* Figure 3). There, the user can query the database and supervise which information bit extracted by `ACIET` will be added to the index.

4 Information Page Analyzer

Given an information page, the preprocessing starts with analyzing the frame structure and existing javascript. Before creating an expressive DOM structure [7,11], the HTML file has to be validated and if necessary corrected. This step is done by the open source unix tool `tidy`. Now our system is able to locate the minimal data region (for more details see Section 4.1) surrounded by certain HTML tags containing the information record searched for. During a depth-first traversal of the DOM tree, the wanted subtree can be isolated according to the headings of the data record, e.g. *"Herausgeber" (publisher)*, *"Betreiber" (operator)* or *"Anbieter" (provider)*. Since we disregard domain name irrelevant

information, we will work further on with a pruned DOM tree. After identifying the minimal data region, all information bits relevant to the domain name are extracted by the attribute-value process (for more details see Section 4.2) with respect to external contexts and internal features. Our system considers about 20 attribute classes and searches their values on the information page of business websites [12]: *company name, address, phone and fax number, e-mail, CEO, management board, domain owner, contact person, register court, financial office, register number, value added tax number (VAT ID)*, etc.

4.1 Detecting the Minimal Data Region

As already shown in Figure 1, an imprint page contains lots of noisy and irrelevant data. In order to determine the minimal data region, we pursue three strategies:

1. Depth-first traversal of the DOM tree to locate the data region of the information bit searched for.
2. Isolation of subtrees containing information bits according to specified headings and pruning of the DOM tree by deleting domain name irrelevant data.[2]
3. Detecting the minimal data region with respect to predefined attribute classes (*"phone number"*, *"fax number"*, *"VAT ID"*).

This method works perfectly (see precision and recall in Table 2) and efficiently due to the minimal text length of the data region. That way, ambiguities arising by reason of multiple contexts are eliminated before they emerge.

4.2 Attribute-Value Process

Detecting the minimal data region limits the search areas in the DOM tree, but does not resolve any ambiguities. If we use, for example, a pattern-based approach to determine a phone number, the same regular expression can also match a fax number. Now we have to assign the correct values to the attributes according to close-by HTML content information provided by the DOM tree.

The recognition of person names causes similar problems: Searching for names on the DOM tree facilitates their localization because these strings are delimited by the HTML tags surrounding the entry. The internal structure of the person name will be characterized by a rule-based method, e.g. a non-left-recursive definite clause grammar. But to discover the person's role, we have to rely on the fact that names occur close to context words hinting on the corresponding attribute classes. That way, the named entity recognition can profit by the HTML structure which refines the search space. To distinguish the person's function, the "value" (person name) has to be extracted together with its "attribute"

[2] We are now able to delete all subtrees captioned by any negative heading (e.g. *"Design" (design), "Realisierung" (realization), "Umsetzung" (implementation), "Web-Hosting" (web hosting)*) from the document object model. That way, this pruning step isolates the business specific subtrees and even eliminates "negative-headed" regions of the tree nested in subtrees preceded by positive titles.

(attribute-value pair). All known attributes were collected during the training stage of our system and compiled into a trie. Moreover, unknown context words can also be correctly attributed by approximate matching with `agrep` [13].

The most remarkable advantage of the attribute-value process is the fact that for the named entity recognition, no large lexicon is required. Thus, the identification of person names is much faster than by a lexicon-based approach. How external and internal indicators work together to guarantee such a success will be discussed in the next section.

Internal and External Indicators for NER. *Internal evidence is derived from within the sequence of words that comprise the name. (...) By contrast, external evidence is the classificatory criteria provided by the context in which a name appears* [14]. Others experimented with several lexicon sizes and discovered that a large comprehensive lexicon cannot improve considerably the precision or recall of a NER system [15]. Hence, we also pursue this strategy and compile the internal and external indicators into the corresponding attribute classes. Some examples for external indicators obtained during the training phase are shown in Table 1. Moreover, the list of indicators is open-ended and managed within different files – a sublist per attribute class. There are two different types of internal indicators: vocabulary lists and regular expressions for digits like phone or fax number. With regard to company name recognition, we can benefit, for example, from 35 legal forms, 130 business types, 400 job titles, and some typical affixes of company names.

Creating an Expressive DOM Structure. Since the DOM tree does not reflect the fundamental characteristics of all HTML tags, we will cluster the HTML tags by their formatting function. We therefore divide the HTML tags in six groups: *character, heading, block, list, table, and image elements*. It is quite obvious that some tags within other tag regions might loose the differentiating property. That way, this deletion of HTML tags helps us to interpret the role of an HTML element within the whole DOM tree and to ignore pointless misplaced elements.

Recognition of Attribute-Value Pairs in Tables. About 70% of the information pages used during the training period encode business specific data in HTML tables. Since those tables totally differ in structure [16], their recognition will cause some problems if we always pursue the strategy to extract the value in the right context of the attribute. During the attribute-value process, we don't really have to recognize the table type (*cf.* Figure 4). Instead, we apply the attribute-value process directly to the table cells.

The extraction of attribute-value pairs in tables of type 1 and 3 seems trivial. If an instance for one of our predefined attribute classes is found, according to type 1, the cell in the next column will be scanned for the corresponding value of the attribute. For type 3, given an attribute separated by at least one delimiter the search for the value can be performed on a single column because both – attribute and value – are located together in the same cell.

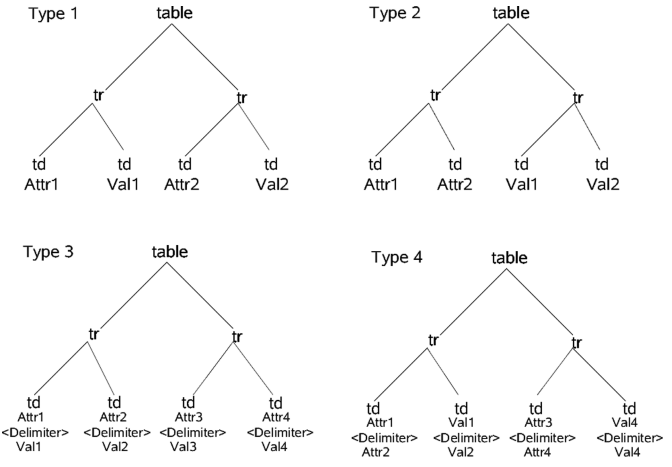


Fig. 4. Different types of HTML tables containing attribute-value pairs

However, we have to face a minor difficulty for type 2 and 4. The structure of type 2 shows that attributes and values are separated by the <tr>-tag and span over two lines. Therefore, the search algorithm has to be adapted to the new situation: After locating the first attribute, the cell in the next column is tested for values or further attributes. This recursive step will be repeated until the corresponding value is identified.

Type 4 is very complex in comparison to the other table structures. Since each cell contains several pieces of information separated by at least one delimiter, we will manage the data by a two-dimensional array. The algorithm

Pseudo-algorithm of the attribute-value process for table type 4

```
1. tds = Find tds; // Array of columns
2. anz_tds = Number of tds; // Number of columns
3. td_delimiters = break each column by <Delimiter>; // Array of Delimiters of columns
4. for ( i = 0; i <anz_tds - 1; i++ ) // for each column except the last
5.     anz_delimiters_td = Number of Delimiters of td_delimiters[i];
                                                  // Number of Delimiters of the concerned column
6.     for ( j = 0; j < anz_delimiters_td; j++ ) // For each text field
7.         next unless length td_delimiters[i][j]; // jump, if empty ist
8.         td_text = td_delimiters[i][j];
9.         for each Attribute_class of already classsified attribute classes
10.            if td_text is Element of concerned Attribute_class
11.               anz_delimiters_next_td = Number of Delimiters of td_delimiters[i+1];
                                                  // Number of Delimiters of  next column
12.               for ( k = 0; k <anz_delimiters_next_td; k++ ) // for the next column
13.                  next unless length td_delimiters[i+1][j];
14.                  if td_delimiters[i+1][k] is Wert_text of concerned Attribut class
15.                     Extract (td_text,  td_delimiters[i+1][k]);
16.                     Delete (td_text,  td_delimiters[i+1][k]);
17.                     Break for inner for-loop
18.               endif
19.            endfor
20.        endif
21.    endfor
22.  endfor
23. endfor
```

Fig. 5. Pseudo-algorithm to identify the attribute-value pairs in table type 4

therefore implemented is shown in Figure 5. One problem occurring quite often is that close-by cells do not contain the same number of delimiters. Thus, a complete scan of the cell divided by the delimiters is necessary and this step has to be repeated until the correct value can be assigned to the corresponding attribute.

Other Structures. Subtrees of the DOM other than HTML tables are also traversed by the attribute-value-process. After locating an attribute, the corresponding value has to be searched within the next HTML tag region or within the string containing an instance of the attribute class and at least one delimiter. Our system will limit the search area in the DOM tree by a pair of attributes and then go through the HTML content elements separated by tags or delimiters string by string. Moreover, the contextual information can also be used to extract company names from the HTML document. There is often some legal notification on the information page hinting on the domain operator, e.g. *"Publisher of this website is the (...)"* , *"assumes no liability"*, *"can not guarantee for the completeness"*, *"accepts no responsibility for the correctness and completeness"*.

4.3 Postprocessing

All extracted information bits are normalized afterwards to guarantee the data consistency. The normalization process affects the following attribute classes:

- company name, legal form, register number
- address: street, zip code, city
- contact: phone and fax number, email
- person name
- legal notification: tax number and VAT ID

The legal form (e.g. *"KG, GmbH, AG"*) within a recognized German company name usually indicates the register department not always correctly given. Hence, the coherence between recognized legal form and register department must be checked in order to assign the right department to the register number.

Spelling mistakes can also occur on informations pages and have to be corrected, e.g. *"Felderstra**ss**e"* must be *"Felderstra**ß**e"*. In order to get a uniform phone number, we have to delete all non-digits and the country code, match the longest area code provided by the lexicon and separate the number into the area code and direct outward dialing sequence. So a phone number like *+49 (0)9146/233* will be transformed to *"(09146) 233"*.

Person names often appear as uncapitalized sequences. In this case the uniform format can be reconsructed by the postprocessing.

With respect to the tax number and VAT ID, the postprocessing is indispensable. Not always information is given according to the standard scheme of the tax number and VAT ID. Given an external indicator *(attribute)* hinting on a VAT ID, our system will expect this number *(value)* to be a VAT ID.

But instead of a VAT ID, for example, the tax number follows: *"Umsatzsteuer-Identifikationsnummer gemäß 27a Umsatzsteuergesetz: DE 053-116-00763"*. The postprocessing step now allows our system to adjust its assumption: The given code *DE 053-116-00763* is not conform to a standardized VAT ID. So we replace the hyphen (-) by a slash (/) and get the valid scheme of a German tax number. During the evaluation scenario, our system correctly identified the tax number in 13 cases, although the local context refers to a VAT ID.

In Figure 2 we already showed an example of an automatically created company information form of the business website `www.prosiegel.de`. The values filled in these slots are normalized according to the above mentioned techniques.

5 Experimental Evaluation

To evaluate the quality of our system with regard to the recognition of information bits indicating business specific data, we designed a small, manually verified test corpus composed of approximately 150 SLDs (websites).

5.1 Test-Data Design

For creating this test base, our system[3] was fed with 924 SLDs picked up randomly by the focused crawler. Among these, 478 SLDs were determined to be appropriate candidates for company websites.[4] The evaluation process was then limited to every third SLD of the candidate set and these 159 SLDs were checked afterwards by visiting the sites with a web browser. As there existed several copies of some SLDs and others were no longer available on the Web, only 150 SLDs remained for test purposes.

5.2 Evaluation Results

Table 2 shows promising results of precision (99.1 % on average) and recall (91.3 % on average) considering the recognition of entities typically found in information pages of business websites. The experimental evaluation presented in this paper is limited to 16 information bits not counting those that have less than 10 instances on the test data.

5.3 Discussion

Needless to say, the evaluation results displayed in Table 2 show more lack of recall than precision. However, we want to discuss the reasons of it.

[3] For research and test purposes the prototype of our system is available at `http://www.cis.uni-muenchen.de/~yeong/ADDR_Finder/addr_finder_de_v12.html`

[4] This step was performed by an external tool – a classificator for business websites not described here.

Table 2. Evaluation results gained on the test SLDs

Extracted Type of Information	Total	Extracted	Correct	Precision	Recall
company name	150	134	129	96.3%	86.0%
street	150	149	147	98.6%	98.0%
zip code	150	150	150	100%	100%
city	150	150	150	100%	100%
phone no.	137	135	134	99.2%	97.8%
fax no.	125	124	124	100%	99.2%
mobile no.	13	13	13	100%	100%
email	126	124	124	100%	98.4%
VAT ID	73	72	72	100%	98.6%
tax no.	25	22	22	100%	88.0%
CEO	39	28	28	100%	71.7%
business owner	24	21	21	100%	87.5%
responsible person	33	24	24	100%	72.7%
authorized person	12	11	11	100%	91.6%
local court	44	38	38	100%	86.3%
register no.	45	38	38	100%	84.4%
On average				99.1%	91.3%

Lack of Precision. 3 of totally 16 information bits vary in precision due to

- mismatches of company names in case of several business occurences
- mistakes in street names in case of missing internal indicators on the page[5]
- non-resolution of ellipsis in phone numbers[6]

Lack of Recall. 13 of totally 16 information bits vary in recall. The reasons for their incomplete or none-recognition are due to

- flash animations, javascript and images protecting the piece of information searched for.
- missing external indicators on information pages, e.g. *Tel., Fax, E-Mail*
- textual representations of phone numbers, e.g. *0700 TEATRON*
- informal specification of tax numbers, register numbers, etc.

These types of errors cause some malfunction in the system.

6 Conclusion

We presented an integrated platform to enable business specific information extraction on the Web. Though we also gave an overview on the localization of information pages on the Web, the main focus in this paper lies on document analysis and business specific information extraction. The core technique to automatically extract structured information is the attribute-value process and use

[5] For the URL `http://www.gestuet-schlossberg.de/ deutsch/impressum.php` our system located the street name *"Ridlerstraße 31 B"*, but it should actually be *"Zachow 5"* which is not matched by the grammar.

[6] A number like *02851/8000+6200* is then transformed to *(02851) 80006200*. But the deletion of "+" is not correct.

of internal and external indicators hinting on the demanded information. The evaluation on the test SLDs shows excellent results for the proposed approach.

Though the linguistic descriptors and the examples of business information pages refer to the German Web, the methods are generalizable for other languages easily applicable to other countries' websites. The system expects the national specific variation of the information format and corresponding internal and external indicators. The integrated file management system can facilitate the maintenance of these indicators.

Even though every new website will uncover new ways that people encode the information, the success of our extraction method will not be affected by changing HTML structures. Tests showed that variations in web content and DOM tree do not influence the attribute-value process. Since our system relies on linguistic resources (e.g. specialized vocabulary), exhaustive studies of context information and a weighted, local interpretation of the HML tags, we can present a quite robust application. Moreover, our system `ACIET` can be extended to integrate further text analysis tools which extract, for example, the activities of companies or their production processes.

References

1. Chang, C.H., Kayed, M., Girgis, M.R., Shaalan, K.F.: A survey of web information extraction systems. IEEE Transactions on Knowledge and Data Engineering 18, 1411–1428 (2006)
2. Krötzsch, S., Rösner, D.: Ontology based extraction of company profiles. In: Proceedings of the 2nd International Workshop on Databases, Documents, and Information Fusion, Karlsruhe, Germany (2002)
3. Labský, M., Svátek, V.: On the design and exploitation of presentation ontologies for information extraction. In: ESWC 2006 Workhshop on Mastering the Gap: From Information Extraction to Semantic Representation, Budva, Montenegro (2006)
4. Svátek, V., Berka, P., Kavalec, M., Kosek, J., Vavra, V.: Discovering company descriptions on the web by multiway analysis. In: New Trends in Intelligent Information Processing and Web Mining (IIPWM 2003), Zakopane, Poland. Advances in Soft Computing series. Springer, Heidelberg (2003)
5. Bsiri, S., Geierhos, M., Ringlstetter, C.: Structuring job search via local grammars. In: Advances in Natural Language Processing and Applications. Research in Computing Science (RCS), vol. 33, pp. 201–212 (2008)
6. Gross, M.: The Construction of Local Grammars. In: Roche, E., Schabès, Y. (eds.) Finite-State Language Processing. Language, Speech, and Communication, pp. 329–354. MIT Press, Cambridge (1997)
7. Liu, B., Grossman, R., Zhai, Y.: Mining data records in web pages. In: KDD 2003: Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining, Washington, D.C., USA, pp. 601–606 (2003)
8. Harris, Z.S.: Mathematical Structures of Language. Interscience Tracts in Pure and Applied Mathematics 21, 152–156 (1968)
9. Harris, Z.S.: Language and Information. Bampton Lectures in America 28, 33–56 (1988)
10. Grishman, R.: Adaptive information extraction and sublanguage analysis. In: Proceedings of Workshop on Adaptive Text Extraction and Mining at Seventeenth International Joint Conference on Artificial Intelligence, Seattle, USA (2001)

11. Liu, W., Meng, X., Meng, W.: Vision-based web data records extraction. In: Ninth International Workshop on the Web and Databases (WebDB 2006), Chicago, USA, pp. 20–25 (2006)
12. Zhu, J., Nie, Z., Wen, J.R., Zhang, B., Ma, W.Y.: Simultaneous record detection and attribute labeling in web data extraction. In: KDD 2006: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, Philadelphia, PA, USA, pp. 494–503 (2006)
13. Wu, S., Manber, U.: Agrep – a fast approximate pattern-matching tool. In: Proceedings USENIX Winter, Technical Conference, San Francisco, CA, USA, pp. 153–162 (1992)
14. McDonald, D.: Internal and external evidence in the identification and semantic categorization of proper names. In: Boguraev, B., Pustejovsky, J. (eds.) Corpus processing for lexical acquisition, pp. 21–39. MIT Press, Cambridge (1996)
15. Mikheev, A., Moens, M., Grover, C.: Named entity recognition without gazetteers. In: Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics, pp. 1–8 (1999)
16. Embley, D.W., Lopresti, D.P., Nagy, G.: Notes on contemporary table recognition. In: Bunke, H., Spitz, A.L. (eds.) DAS 2006. LNCS, vol. 3872, pp. 164–175. Springer, Heidelberg (2006)

Low-Cost Supervision for Multiple-Source Attribute Extraction

Joseph Reisinger[1,⋆] and Marius Paşca[2]

[1] University of Texas at Austin, Austin, Texas
joeraii@cs.utexas.edu
[2] Google Inc., Mountain View, California
mars@google.com

Abstract. Previous studies on extracting class attributes from unstructured text consider either Web documents or query logs as the source of textual data. Web search queries have been shown to yield attributes of higher quality. However, since many relevant attributes found in Web documents occur infrequently in query logs, Web documents remain an important source for extraction. In this paper, we introduce Bootstrapped Web Search (BWS) extraction, the first approach to extracting class attributes simultaneously from both sources. Extraction is guided by a small set of seed attributes and does not rely on further domain-specific knowledge. BWS is shown to improve extraction precision and also to improve attribute relevance across 40 test classes.

1 Introduction

Class attributes capture quantifiable properties (e.g., *hiking trails*, *entrance fee* and *elevation*), of given classes of instances (e.g., *NationalPark*), and thus potentially serve as a skeleton towards constructing large-scale knowledge bases automatically. Previous work on extracting class attributes from unstructured text consider either Web documents [1] or query logs [2] as the extraction source. In this paper, we develop *Bootstrapped Web Search* (BWS), a method for combining Web documents and query logs as textual data sources that may contain class attributes. Web documents have textual content of higher semantic quality, convey information directly in natural language rather than through sets of keywords, and contain more raw textual data. In contrast, search queries are usually ambiguous, short, keyword-based approximations of often-underspecified user information needs. Previous work has shown, however, that extraction from query logs yields significantly higher precision than extraction from Web documents [2].

BWS is a generic method for multiple-source class attribute extraction that allows for corpora with varying levels of extraction precision to be combined favorably. It requires no supervision other than a small set of seed attributes for each semantic class. We test this method by combining query log and Web document corpora, leveraging their unique strengths in order to improve coverage and precision. Using BWS, extracted attributes from classes pertaining to various domains of interest to Web search users yield accuracy exceeding current state of the art using either Web documents or search queries alone.

⋆ Contributions made during an internship at Google.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 382–393, 2009.

2 Bootstrapping Web Extraction

Multiple-source extraction is outlined in Section 2.1 and applied to a corpus of *relevant* textual documents constructed from top Web search results, as described in Section 2.2.

2.1 Combining Multiple Data Sources

Significant previous work has been done on attribute extraction across a wide variety of data sources, e.g. news reports, query logs and Web documents. If extraction from such domains yields high precision results, intuitively it should be possible to obtain even more accurate attributes using a combination of data sources. Such a procedure may also help mitigate the particular biases inherent to each of the source text distributions.

We consider the general problem of extracting and ranking a set of candidate attributes $\mathcal{A}$ using a ranked list of (possibly noisy) seed attributes $\mathcal{S}$. The only assumption placed on $\mathcal{S}$ is that better attributes occur earlier than spurious attributes on average, e.g. the first five attributes are assumed to have higher average precision than the first 100. $\mathcal{S}$ is then used to rank a set of candidate attributes $\mathcal{A}$ extracted from a different source. Although the new supervision targets $\mathcal{S}$ are inherently noisy, there is nevertheless significant benefit to increasing the total amount of supervision available.

Naive approaches that use fixed number of attributes from $\mathcal{S}$ are not optimal because the increased noise lowers the average precision of the highest ranked attributes. In order to leverage noisy supervision, we introduce a simple one-parameter smoothing procedure that builds on the assumption that a seed's rank is proportional to its precision. The candidate attributes are ranked accordingly, weighting attributes higher when they are more similar to a large number of the most precise seed attributes. Intuitively, a candidate attribute $a \in \mathcal{A}$ should be ranked highly overall if it has a high average similarity to the most precise seed attributes. That is, an attribute's rank should take into account the average ranks of the seeds to which it is most similar.

The specific ranking algorithm used in this paper is detailed in Figure 1. Given $\mathcal{A}$, the unordered set of extracted candidate attributes for class C, and $\mathcal{S}$, the set of (noisy)

Require: $\alpha \in (0, \infty)$, $\mathcal{A}$ is a set of unranked candidate attributes, and δ is a vector of ranks at which to calculate ρ.

$\delta_0 \leftarrow 0$
$\mathcal{A}_{\text{ranked}} \leftarrow []$
for i in $1..|\delta|$ **do**
  calculate $\rho_a(\delta_i)$ for all $a \in \mathcal{A}$
  for k in $1..\frac{(\delta_i - \delta_{i-1})}{\alpha}$ **do**
    $a_{\text{new}} \leftarrow \arg\max_{a \in \mathcal{A}} [\rho_a(\delta_i)]$
    $\mathcal{A} \leftarrow \mathcal{A} \backslash a_{\text{new}}$
    $\mathcal{A}_{\text{ranked}} \leftarrow \mathcal{A}_{\text{ranked}} \cup a_{\text{new}}$
  end for
end for

Fig. 1. Bootstrapped extraction procedure

ranked seed attributes for class C extracted from a different source, each attribute $a \in \mathcal{A}$ is assigned a class-specific *extraction profile*,

$$\rho_a(s) \stackrel{\text{def}}{=} \frac{1}{s} \sum_{i=0}^{s} \text{Sim}(a, \mathcal{S}_i),$$

equal to its average similarity function Sim over the top $s < |\mathcal{S}|$ total seeds. Given a set of candidate attributes and their associated extraction profiles, a ranked list $\mathcal{A}_{\text{ranked}}$ is constructed incrementally. At each step $s < |\mathcal{S}|$, the top α attributes ranked by their average similarity to the top s seeds, $\rho_a(s)$, are added to $\mathcal{A}_{\text{ranked}}$, and removed from $\mathcal{A}$.

It is possible to take $s \in \mathbb{N}, s < |\mathcal{S}|$, however each evaluation of $\rho_a(s)$ requires $O(s)$ similarity calculations, and hence $O(|\mathcal{A}||\mathcal{S}|^2)$ operations would be required to rank all candidate attributes. Although $|\mathcal{S}|$ is effectively bounded by the number of relevant seed attributes, this computation can be time-consuming in practice. Thus, for efficiency we calculate ρ_a at a discrete set of *seed levels*, $\delta = [5, 10, 20, 40, 60, 100]$, where, e.g. $\delta_2 = 10$. $\mathcal{A}_{\text{ranked}}$ is then constructed recursively, and at each step the top $(\delta_s - \delta_{s-1})/\alpha$ attributes ranked by $\rho(\delta_s)$ are added to $\mathcal{A}_{\text{ranked}}$ (skipping duplicates). The parameter α controls how strongly attribute ranking should prefer to emphasize the number of seeds over the list rank; as $\alpha \to \infty$, more seeds are used on average to score each attribute. Intuitively, this method works because each seed level has a different peak where the best results are obtained. By adjusting α, we can construct a list of attributes with high precision across all ranks.

2.2 Extraction from Relevant Web Documents

As a specific application of combining multiple textual data sources for attribute extraction, we use attributes extracted from query logs in [3] as noisy supervision for finding attributes in *relevant* Web documents. Web documents relevant to a particular class are found by performing search queries for each instance of that class and collecting the top documents returned. This approach is novel and is hypothesized to yield better attributes than untargeted documents precisely because of the increase in relevancy. There are three main phases: 1) document acquisition and noun-phrase extraction (Section 2.2), 2) context vector generation (Section 2.2) and 3) attribute ranking (Section 2.2). Figure 2 summarizes the approach taken, using the class *Actor* as an example.

Noun Phrase Extraction. Relevant documents for a class C are obtained by collecting the top D documents returned from performing a Web search for each instance $I \in C$, where the query is taken simply as the instance name, e.g. Jet Li and Mel Gibson become search queries in Figure 2.

Non-textual data is removed from the document and part-of-speech tagging is run on the resulting plain text. All base noun phrases occurring in the top D documents for each instance search query $I \in C$ are extracted along with n-gram contexts and used as candidate attributes for the class C.

Base noun phrases are identified heuristically as sequences of one or more words tagged as noun (NN) or proper noun (NNS), and may contain leading adjectives (JJ; e.g. *physical appearance*) or "of" (e.g. *date of birth*). In order to find generic attributes,

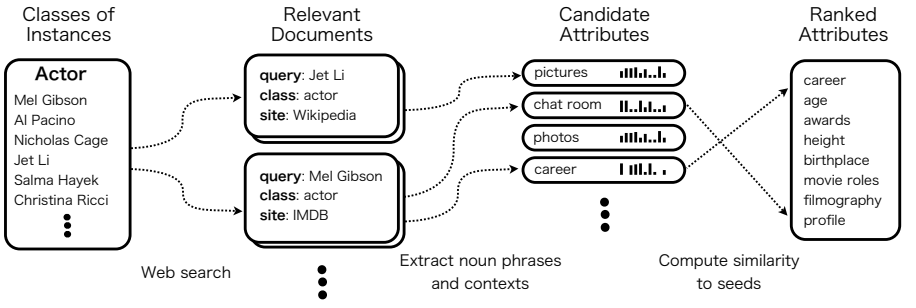


Fig. 2. Overview of the Web-search seed-based extraction procedure

modifiers within base noun phrases are iteratively discarded, and the resulting phrases are added to the pool of extracted attributes (e.g. *physical appearance* would also generate *appearance*). In addition, candidate attributes are also filtered if they occur fewer than T_{min} times in the document corpus.

Context Vector Generation. Each time a noun phrase is extracted, a corresponding *pattern* is generated, consisting of the n-gram context to the immediate left (pattern prefix) and right (pattern postfix) of the noun phrase, stopping at the end or beginning of the sentence. In all results reported in this paper, 3-gram patterns (i.e. three words to the left and right of the noun phrase) are used.

Each extracted pattern is used as a basis for generating more generic patterns by taking the original extracted pattern and replacing occurrences of words tagged NNP (proper noun) and CD (number) with generic symbols. All combinations of possible replacements are added. Furthermore, if the instance that generated the document result is found within the pattern it is replaced with a generic instance symbol. Such replacement results in higher-coverage patterns that may relate the extracted noun phrases to the instance. Finally, to improve targeting, only patterns containing at least one of the part-of-speech tags POS, IN and VB are retained; all other patterns are discarded.

All attributes (both candidate and seed) can now be represented as *context vectors*, capturing the distribution over n-gram contexts. Context vectors are computed separately for each class and simply contain counts of the number of times context X was extracted for attribute A in any Web document associated with class C.

Attribute Ranking. Once context vectors based on pattern counts are generated for each seed and candidate attribute, distance can be computed using various measures of similarity. In this paper we employ *Jaccard's coefficient* [4],

$$\text{Jac}(X_a, X_s) \stackrel{\text{def}}{=} \frac{|X_a \cap X_s|}{|X_a \cup X_s|},$$

where X_a is the set of contexts for the candidate attribute, X_s is the set of contexts for the seed attribute, and $|\cdot|$ denotes set cardinality. Jaccard's coefficient measures

the ratio of shared context vectors to total context vectors covered by the two attributes. Candidate attributes for each class are then ranked by their average similarity to the seed attributes for that class. Jaccard's coefficient is simple to compute and yields attribute rankings that are robust to small changes in the underlying extraction method.

3 Experimental Setup

Section 3.1 introduces the five extraction methods compared in this paper, Section 3.2 gives the parameter settings, Section 3.3 gives an overview of the class instances underlying our attribute extraction results, Section 3.4 describes the query logs and Web document corpora used in this study and Section 3.5 discusses our evaluation method.

3.1 Experimental Runs

In this paper, we compare five different attribute extraction systems, covering several combinations of extraction methods and corpora:

1. WD (Web Document based extraction) – This method uses a fixed set of linguistically motivated surface patterns (e.g. "the A of I" or "I's A" for instance I and attribute A) combined with a pre-specified set of instance classes to extract attributes from a Web-based textual corpus [3]. Attributes for a class C are extracted whenever a specific pattern is matched with an instance $I \in C$.
2. QP (Query logs using Patterns) – This method uses the same procedure as WD, but is applied to query logs instead of Web documents, yielding higher precision in practice [2].
3. QL (Query Logs using seeds) – A set of 5 manually specified *seed* attributes for each class are used to automatically extract patterns (syntactic contexts) that contain a seed and an instance from the same class. These patterns are then used to find other candidate attributes, i.e. non-seed noun-phrases that match the extracted patterns. These candidate attributes are then ranked based on their similarity across all contextual patterns (as discussed in Section 2.2). This method produces significantly more precise attributes than QP [3].
4. WS (Web Search extraction) – The first novel method proposed in this paper; attributes are extracted from relevant documents returned from search queries (see Section 2.2).
5. BWS (Bootstrapped Web Search extraction) – The second method proposed in this paper; it uses the top 100 high-precision attributes extracted with QL as additional supervision for WS, as outlined in Section 2.1.

3.2 Extraction Parameters

In all results reported here, the BWS smoothing parameter α is set $\alpha = 0.75$, the seed levels δ specifying when $\rho_a(\cdot)$ is calculated are set at $\delta = [5, 10, 20, 40, 60, 100]$, the number of documents returned for each instance query $D = 200$, and the noun-phrase filtering threshold T_{min} is set to 20.

3.3 Classes of Instances

Extraction specificity is controlled via a set of *instance clusters* (corresponding to semantic classes) for which we wish to obtain attributes. Obtaining such collections has been studied extensively in previous work [5,6,7]. In this paper we use 40 classes chosen manually to have broad coverage.

One possible source of bias that we do not attempt to control for in this paper might be termed *instance class bias*. Such bias can manifest itself in several ways, for example when the instance coverage is too narrow to fully contain the target class (e.g. *VideoGame* is skewed significantly towards recent games) or more subtly when only a few instances in the class are common and the rest are uncommon, e.g. as in the case of *Religion*.

3.4 Textual Data Sources

Our Web documents corpus is procured by retrieving the top 200 search results for each instance using Google and removing all non-html documents. Non-textual elements (e.g. html, javascript) are also removed, and part-of-speech tagging is performed using the TnT tagger [8]. The total (compressed) size of the corpus is over 16GB. The query log data is taken from a random sample of anonymized English queries submitted to Google. The sample contains approximately 50 million queries and frequency counts.

3.5 Evaluation Methodology

During evaluation, each candidate attribute extracted for a class is hand-labeled as one of three categories: vital, okay and wrong (cf. [9]). *Vital* attributes should appear in any complete list of attributes for the target class; *okay* attributes are useful but non-essential; *wrong* attributes are incorrect. These categories are converted into numerical scores in order to calculate the overall precision of attributes extracted for that class (vital=1.0, okay=0.5 and wrong=0.0). Although time-consuming, in order to reduce the effects of human error and bias all labels are checked by a second independent judge.

4 Evaluation Results

Our main results are two-fold. First, extraction from top Web search results yields higher attribute precision than fixed-pattern Web extraction, but has lower precision than extraction from queries, confirming similar results using untargeted Web corpora [3]. Second, the noisy attributes extracted from query logs can be used as additional seed targets for Web search extraction, yielding better precision than either method individually.

4.1 Precision

A plot of average precision over all classes for the five extraction methods is given in the upper left-hand graph in Figure 3 and breakdowns over a sample of the 40 classes are shown in the remaining five graphs. Overall these results show that the novel Web extraction method using relevant documents (WS) yields higher precision attributes than

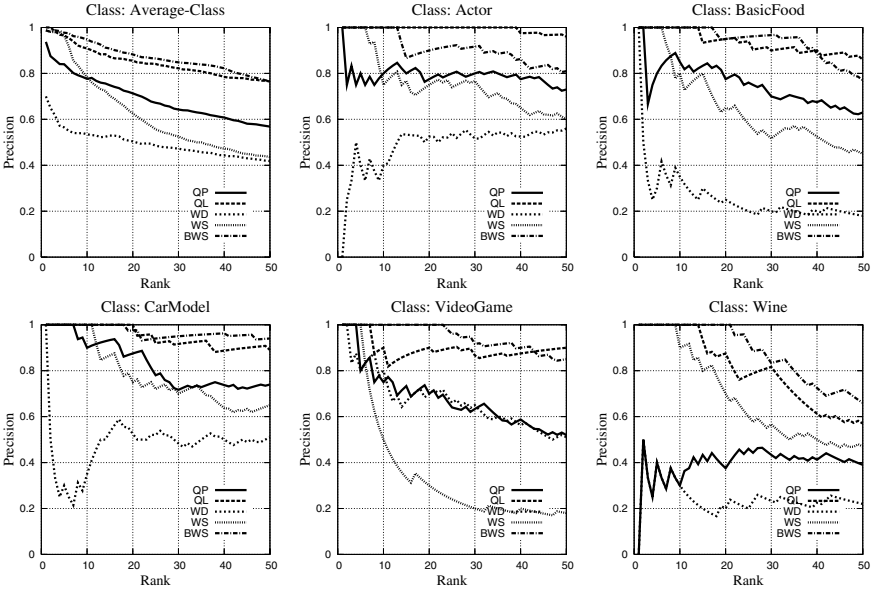


Fig. 3. Average precision plot over all 40 classes and individual precision plots for 8 specific classes

Table 1. Comparative precision of attributes, as well as recall of vital attributes relative to QL, measured at ranks 5, 10, 20 and 50 in the ranked lists of attributes extracted by various methods

Method	Precision (%)				Relative Recall (%)			
	5	10	20	50	5	10	20	50
QP	76.3	72.1	64.3	53.1	5.3	11.2	19.2	31.3
QL	96.5	90.9	85.6	76.5	11.6	23.5	44.3	100
WD	56.5	53.5	50.4	41.8	2.1	3.5	6.7	11.9
WS	96.5	78.3	62.5	43.6	9.3	11.6	13.7	17.8
BWS	**97.8**	**94.8**	**88.3**	**76.5**	9.3	21.0	41.8	76.1

standard extraction from Web documents (WD), but still significantly underperforms the query-log based extraction (QL), due to the increased noise inherent to web pages. However, BWS yields higher precision than QL at most ranks in the ranked lists of attributes, gaining up to a 42% reduction in error (at rank 10 in Table 1).

4.2 Analysis of Extracted Attributes

The top 10 attributes extracted using BWS are shown in Table 2. For comparison, attributes that also returned among the top 10 results using QL are italicized. Important novel attributes include atomic number for *ChemicalElem* and exchange rates for *Currency*.

Table 2. Top ten attributes extracted using BWS on a subset of the 40 evaluation classes; attributes also found in the top 10 in QL are shown in italics

Class	Attributes
Actor	career, *age,* awards, *height, weight, birthplace,* life, date of birth, filmography, profile
AircraftModel	weight, *length,* fuel capacity, wing span, *history, specifications, photographs,* fuel consumption, cost, price
BasicFood	*size, color, taste,* calories, *nutrition, nutritional value*, *allergies, nutritional information*, ingredients, *nutrients*
CarModel	transmission, *acceleration,* top speed, gearbox, gas mileage, owners manual, transmission problems, engine type, mpg, reliability
ChemicalElem	*symbol,* atomic number, mass, *classification,* atomic structure, freezing point, discovery date, number, physical properties, atomic weight
Company	*headquarters, chairman,* location, *ceo,* stock price, company profile, corporate office, president, parent company, stock quote
Country	*president, area,* population, *flag, economy, religion, climate, geography,* culture, currency
Currency	*country,* exchange rates, *symbol,* purchasing power, currency *converter,* currency conversion, currency exchange, subunit, sign, abbreviation
Drug	side effects, *dosage, price,* color, withdrawal symptoms, mechanism of action, mechanism, *dangers,* overdose, dose
Empire	ruler, *size,* collapse, founding, *location, definition, chronology, downfall,* kings, end
Hurricane	damage, strength, *date,* death toll, *track, destruction,* wind speed, *statistics,* history, storm surge
Movie	*director,* cast, producer, genre, *crew, synopsis,* official site, release date, script, *actors*
Painter	paintings, *biography,* birthplace, works, *artwork, bibliography, autobiography quotations, quotes, biographies*
Religion	*gods,* beliefs, message, *origin, teachings, principles, practices,* influences, doctrines, *tenets*
River	*tributaries,* location, *mouth, length, source, headwaters, depth, width, origin, gradient*
SearchEngine	quality, speed, market share, number of users, reliability, number, mission statement, phone book, algorithms, video search
SoccerClub	league, *titles,* head coach, *capacity, managers,* official website, *chairman,* official site, *tours,* flags
TerroristGroup	attacks, *leader, goals, meaning, leadership, website, photos, definition,* members, organization
Treaty	countries, *date, ratification,* clauses, *purpose, definition, summary, cons, members, provisions*
University	alumni, *dean,* research areas, number of students, department of psychology, career center, department, number, location, logo
VideoGame	platform, genre, price, *creator,* official site, website, system requirements, concept art, cheats, *reviews*
Wine	vintage, *style, color, taste,* wine reviews, *cost, style* of wine, wine ratings, fermentation, aging
WorldWarBattle	date, result, *location,* combatants, *images, importance, summary, timeline,* casualties, survivors

Table 3. Top 10 examples of attributes found in the top 50 results from BWS, but not in the top 100 for QL on a subset of the 40 classes

Class	Attributes
Actor	politics, sexuality, movie roles, official website, favorites, life story, marriages, cause, autobiography, height
AircraftModel	airplanes, seat map, length, seat configuration, maiden flight, facts, landing gear, price, production, cockpit
BasicFood	photo, origin, picture, carbohydrates, taste, calories, carbs, fiber, glycemic index, nutrition facts
CarModel	alternator, gas tank, clutch, fuel gauge, performance parts, recalls, fuel pump, headlights, modifications, ignition switch
Company	stock price, offices, general counsel, organizational structure, board of directors, code, founder, corporate social responsibility, company profile, operations
Country	weather, climate, land use, ethnic groups, people, precipitation, terrain, state, culture, current events
Drug	adverse effects, symptoms, abuse, price, recall, definition, volume, dangers, overdose, impotence
Empire	culture, beginning, government, kings, religions, flag, architecture, rulers, definition, trade routes
Movie	author, script, main characters, plot, posters, actors, clips, climax, release date, screenplay
Religion	holy days, origin, diet, basics, rituals, world, tenets, signs, cosmology, hierarchy
University	campus map, mascot, online courses, board of trustees, job openings, library catalog, faculty, homepage, dean, library
VideoGame	reviews, instruction manual, guide, review, similar games, prices, updates, system requirements, platform, world map
Wine	region, flavor, body, characteristics, franc, health benefits, vintage, vintage chart, year, red blends

Table 4. List of extracted attributes that appear in the top 50 results from QL, but not in BWS on a subset of the 40 classes

Class	Attributes
Actor	photogallery, life biography, vital statistics, ethnic background, b— pics, birthdate, hair colour, chat room, hairdos, movie list
AircraftModel	roll out, air new zealand, seat plan, best seats, seating layout, seating chart, drawing, interior photos, for sale, cross section
BasicFood	mercury level, nutritional data, calorie count, serotonin, growth stages, vitamin k, uric acid, tryptophan, selective breeding, fat grams
CarModel	gt specs, stalling, electrical, cooling system, tsb, overheating, radio removal, consumer report, maintenance schedule, towing capacity
Company	contact us, marketing strategies, tsunami donation, corporate address, stock performance, financial ratios, hoovers, 2004 annual report, company overview, store locator
Country	gnp, population distribution, landforms, physical features, national sport, national bird, national symbols, goverment, population pyramid, royal family
Drug	pediatric dosing, health benefits, annual sales, therapeutic index, teratogenicity, generic equivalent, contraindications, halflife, structural formula, recreational use
Empire	weaknesses, accomplishments, economics, fashion, atlas, inventions, contributions, photos, world map, picture
Movie	colouring pages, historical accuracy, free music downloads, soundboards, film location, cast list, character names, plot line, free clipart, behind the scenes
Religion	tenants, supreme being, holy cities, creation story, holy sites, rites of passage, early history, historical events, demographics, sacred symbols
University	graduation 2005, hillel, webcam, school colors, t - shirts, speech pathology, department of english, tuition costs, desktop wallpaper, campus photos
VideoGame	serial key, desktop theme, minimum spec, nocd crack, pc walkthrough, game walk through, walk thru, cheats for pc, cheat sheet, download demo
Wine	sartori, pilgrim, poached pears, fat bastard, mark west, acacia, toasted head, tommasi, attar, shelf life

The top 10 attributes extracted using BWS not found in the top 100 query log results are listed for a representative sample of the 40 classes in Table 3. Likewise, Table 4 shows the top 10 attributes extracted in QL but not found in the top 50 BWS.

As a tractable alternative to manually enumerating all attributes for each target class, attribute coverage was computed relative to a manually collected reference set, consisting of those attributes labeled as *vital* in the top 50 QLresults (Figure 1). BWS shares 65% of its top 50 attributes with the top 50 from its seed source QL. However, it shares over 76% of QL's vital attributes, indicating that the filtering taking place is biased towards more relevant attributes. A total of 13% of the attributes generated are completely novel, not found anywhere within the top 100 query log results. This result indicates that a major benefit of BWS is not only its ability to extract novel attributes, but also its ability to re-rank attributes already found by QL, giving incorrect attributes lower ranks.

4.3 Domain Specificity

Web domains that consistently produce good attributes for one class typically do not yield useful attributes for any other classes. Even websites like `wikipedia.org` do not consistently yield good attributes across multiple classes. Of the 231 unique domains that generated only "good" attributes in the top 50 results, only 9 of them occur in more than class (7 occurring in exactly two classes and the remaining 2 occurring in 4). Thus, a site's attribute quality is highly dependent on the class being considered. This result highlights the difficulty of implementing generic attribute extraction methods that yield high precision across many classes, and sheds insight into why WS outperforms WD.

5 Related Work

Class instance extraction is necessary as an intermediate step towards automatically building knowledge bases, and is in particular important for relation and attribute extraction [10,11]. Instances have been extracted by iterating using a few seed extraction rules [12], by mining named entities from comparable news articles [13] or multilingual corpora [14], and by bootstrapping [15]. Experimental results from [16] indicate that named entity recognizers can boost the performance of weakly supervised extraction of class instances, but only for a few coarse-grained types such as *Person* and only if they are simpler to recognize in text [16]. By adding the five best items extracted from 1700 text documents to the seed set after each iteration, 1000 semantic lexicon entries are collected after 200 iterations in [17]. In [18], handcrafted extraction patterns are applied to a collection of 60 million Web documents to extract instances of the classes *Company* and *Country*. The output type of such previous studies consists of sets of class instances given seed instances, rather class attributes given seed attributes.

Significant previous literature exists in attribute extraction. In [19], attributes and other knowledge is acquired from Web users who explicitly specify it by hand. In contrast, we collect attributes automatically from unstructured text. Several studies [1,3,20,21,22] attempt to acquire attributes from Web documents. In [1], handcrafted lexico-syntactic patterns are applied to unstructured text within a small collection of Web documents. The resulting attributes are evaluated through a notion of question answerability, wherein an attribute is judged to be valid if a question can be formulated about it. Our evaluation is stricter: many attributes, such as *long term uses* and *users* for the class *Drug*, are marked as wrong in our evaluation, although they would easily pass the question answerability test (e.g., *"What are the long term uses of Prilosec?"*) used in [1]. In [3], target classes are similarly specified as sets of representative instances and attributes are collected by applying hand-written patterns to unstructured text. In [20], a general-purpose search engine is used to identify potentially useful Web documents for attributes extraction (similar to WD). Unlike our approach, [20] perform extraction on structured text and report significantly lower performance (49% accuracy at rank 50 vs. 76% for our system).

The method introduced in [21] acquires attributes as well as associated values from structured text within Web documents. Queries are submitted to general-purpose search engines and the top search resulting content is analyzed. Besides using structured rather than unstructured text, there are a few key differences. First, the search queries issued are based on hand-written templates using the label of the target class, e.g., *"lists of movies"* for the class *Movie*. Our simple, instance-based search queries constitute a more flexible approach. Second, another set of hand-written patterns is employed to filter out spurious candidate attributes. In our approach, a single scoring function is applied to all candidate attributes with no post-filtering, potentially increasing coverage.

Finally, attributes can be considered relations between objects in the given class, and also between objects from other classes. Thus extracting attributes allows for gauging existing methods for extracting semantic relations [18,23] and also for acquiring relations themselves.

6 Conclusion

Extracting attributes from documents on the Web is difficult due to the presence of noise, however such sources are significantly more content rich than other, more high-precision attribute sources such as query logs. In this paper we develop a conceptually simple seed-based method for leveraging high-precision low-coverage attribute sources (e.g. query logs) in order to improve extraction from high-coverage low-precision sources such as Web documents. This approach yields significantly higher precision than previous methods (both Web and query-log based) and furthermore improves coverage by mitigating the "search bias" inherent in query-log based extraction.

References

1. Tokunaga, K., Kazama, J., Torisawa, K.: Automatic discovery of attribute words from web documents. In: Dale, R., Wong, K.-F., Su, J., Kwong, O.Y. (eds.) IJCNLP 2005. LNCS, vol. 3651, pp. 106–118. Springer, Heidelberg (2005)
2. Paşca, M.: Organizing and searching the World Wide Web of facts - step two: Harnessing the wisdom of the crowds. In: Proceedings of the 16th World Wide Web Conference (WWW 2007), Banff, Canada, pp. 101–110 (2007)
3. Paşca, M., Van Durme, B., Garera, N.: The role of documents vs. queries in extracting class attributes from text. In: Proceedings of the 16th International Conference on Information and Knowledge Management (CIKM 2007), Lisbon, Portugal, pp. 485–494 (2007)
4. Lee, L.: Measures of distributional similarity. In: Proceedings of the 37th Annual Meeting of the Association of Computational Linguistics (ACL 1999), College Park, Maryland, pp. 25–32 (1999)
5. Agichtein, E., Gravano, L.: Snowball: Extracting relations from large plaintext collections. In: Proceedings of the 5th ACM International Conference on Digital Libraries (DL 2000), San Antonio, Texas, pp. 85–94 (2000)
6. Lin, D., Pantel, P.: Concept discovery from text. In: Proceedings of the 19th International Conference on Computational linguistics (COLING 2002), Taipei, Taiwan, pp. 1–7 (2002)
7. Shinzato, K., Torisawa, K.: Acquiring hyponymy relations from Web documents. In: Proceedings of the 2004 Human Language Technology Conference (HLT-NAACL 2004), Boston, Massachusetts, pp. 73–80 (2004)
8. Brants, T.: TnT - a statistical part of speech tagger. In: Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP 2000), Seattle, Washington, pp. 224–231 (2000)
9. Voorhees, E.: Evaluating answers to definition questions. In: Proceedings of the 2003 Human Language Technology Conference (HLT-NAACL 2003), Edmonton, Canada, pp. 109–111 (2003)
10. Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., Slattery, S.: Learning to construct knowledge bases from the world wide web. Artificial Intelligence 118, 69–113 (2000)
11. Schubert, L.: Turing's dream and the knowledge challenge. In: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006), Boston, Massachusetts (2006)
12. Collins, M., Singer, Y.: Unsupervised models for named entity classification. In: Proceedings of the 1999 Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC 1999), College Park, Maryland, pp. 189–196 (1999)

13. Shinyama, Y., Sekine, S.: Named entity discovery using comparable news articles. In: Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004), Geneva, Switzerland, pp. 848–853 (2004)
14. Klementiev, A., Roth, D.: Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006), Sydney, Australia, pp. 817–824 (2006)
15. Riloff, E., Jones, R.: Learning dictionaries for information extraction by multi-level bootstrapping. In: Proceedings of the 16th National Conference on Artificial Intelligence (AAAI 1999), Orlando, Florida, pp. 474–479 (1999)
16. Feldman, R., Rosenfeld, B.: Boosting unsupervised relation extraction by using NER. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP-ACL 2006), Sydney, Australia, pp. 473–481 (2006)
17. Thelen, M., Riloff, E.: A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2002), Philadelphia, Pennsylvania, pp. 214–221 (2002)
18. Cafarella, M., Downey, D., Soderland, S., Etzioni, O.: KnowItNow: Fast, scalable information extraction from the Web. In: Proceedings of the Human Language Technology Conference (HLT-EMNLP 2005), Vancouver, Canada, pp. 563–570 (2005)
19. Chklovski, T., Gil, Y.: An analysis of knowledge collected from volunteer contributors. In: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005), Pittsburgh, Pennsylvania, pp. 564–571 (2005)
20. Ravi, S., Paşca, M.: Using structured text for large-scale attribute extraction. In: Proceedings of the 17th International Conference on Information and Knowledge Management (CIKM 2008), Napa Valley, California, pp. 1183–1192 (2008)
21. Yoshinaga, N., Torisawa, K.: Open-domain attribute-value acquisition from semi-structured texts. In: Proceedings of the 6th International Semantic Web Conference (ISWC 2007), Workshop on Text to Knowledge: The Lexicon/Ontology Interface (OntoLex 2007), Busan, South Korea, pp. 55–66 (2007)
22. Probst, K., Ghani, R., Krema, M., Fano, A., Liu, Y.: Semi-supervised learning of attribute-value pairs from product descriptions. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, India, pp. 2838–2843 (2007)
23. Pantel, P., Pennacchiotti, M.: Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006), Sydney, Australia, pp. 113–120 (2006)

An Integrated Architecture for Processing Business Documents in Turkish

Serif Adali[1], A. Coskun Sonmez[2], and Mehmet Gokturk[3]

[1] Istanbul Technical University, Department of Computer Engineering, 34469, Istanbul, Turkey
serifadali@yahoo.com
[2] Yildiz Technical University, Department of Computer Engineering, 34349, Istanbul, Turkey
acsonmez@ce.yildiz.edu.tr
[3] Gebze Institute of Technology, Department of Computer Engineering, 41400, Kocaeli, Turkey
gokturk@gyte.edu.tr

Abstract. This paper covers the first research activity in the field of automatic processing of business documents in Turkish. In contrast to traditional information extraction systems which process input text as a linear sequence of words and focus on semantic aspects, proposed approach doesn't ignore document layout information and benefits hints provided by layout analysis. In addition, approach not only checks relations of entities across document for verifying its integrity, but also verifies extracted information against real word data (e.g. customer database). This rule-based approach uses a morphological analyzer for Turkish, a lexicon integrated domain ontology, a document layout analyzer, an extraction ontology and a template mining module. Based on extraction ontology, conceptual sentence analysis increases portability which requires only domain concepts when compared to information extraction systems that rely on large set of linguistic patterns.

1 Introduction

Even though there has been an on-going effort for eliminating free-formatted text documents for almost thirty years, certain forms of communication continue to be unstructured such as fax and e-mail, where free-formatted text needs to be interpreted in order to understand its meaning and extract necessary information. Proposed integrated architecture has been tested for processing business documents in Turkish. The test data has been created based on sample documents provided by a local bank, by generating fictitious customer data (e.g. names, account number). Test documents are assumed to be optically recognized (OCR) and free of spell checking errors. Input document is treated as a combination of document model and event concept, where entities within document are cross-related to each other, and document can be correctly understood if certain clues are provided by the document layout analysis. Proposed integrated architecture combines document layout analysis, ontology-based

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 394–405, 2009.

information extraction and morphological analyzer for Turkish. Information Extraction (IE) is generally defined as a form of natural language processing, in which certain types of information must be recognized and extracted from text. Within the last three decades, the field of IE has gained a lot of importance mainly fostered by the Message Understanding Conferences (MUCs) started in 1987. Main IE tasks can be listed as extracting named entities, extracting pre-specified events (scenario) and extracting relations between entities and events. In contrast to traditional (IE) systems including Message Understanding Conference (MUC) evaluations, proposed approach doesn't ignore document layout information, where traditional systems process input text as a linear sequence of words, mainly focusing on semantic aspects, causing loss of valuable hints about input document. Based on location of an entity in a document, document layout analyzer selects correct tag for a named entiy. This study also stresses the importance of ontology-based IE, where it's defined as explicit specifications of concepts to provide information extraction systems with machine-readable information by representing the domain knowledge in a formal way [1]. Although natural language is highly connected to real world knowledge, currently parsing approaches don't make use of ontology very effectively; instead they depend on rule-based or statistical parsers. In this study extraction ontology is used for extracting domain specific events, while domain ontology is used for named entity recognition. Morphological analysis is significantly complex when compare to languages like English. It returns thematic structure of the sentence, tense, aspects of an event and modality for deeper understanding. Sentence structure of Turkish is highly flexible caused by morphological inflections. Considering this feature, based on the extraction ontology conceptual sentence analyzer focuses on locating domain specific concepts regardless of their location in a sentence. This method requires only domain concepts when compared to IE approaches which rely on large set of linguistic patterns, which seems to be its biggest advantage.

2 Turkish Language

Turkish language is the sixth most widely spoken language in the world spread over a large geographical area in Europe and Asia. It belongs to Altaic branch of Ural-Altaic family of languages with free-constituent order, agglutinative morphology, and head-final sentence structure. Information Extraction for Turkish is a developing research field, where currently only available Ph.D. thesis is published by Tür [2]. He worked on a statistical information extraction system for Turkish, where he applied statistical methods using both lexical and morphological information to basic tasks including word segmentation, vowel restoration, sentence segmentation, topic segmentation and name tagging**.** The following sentences demonstrate free constituent order in a Turkish sentence. All sentences mean "This boy went to school yesterday." ("*dün*"="yesterday", "*bu çocuk*"="this boy", "*gitti*"="went", "*okula*"="to school")

- *Dün / bu çocuk/ okula/ gitti.*
- *Dün / okula / bu çocuk / gitti.*
- *Bu çocuk / okula / dün / gitti.*
- *Okula / dün / bu çocuk / gitti.*

In contrast to languages like English, where there is a very small number of possible word forms with a given root word, languages like Turkish and Finnish have very productive agglutinative morphology, which makes it possible to produce thousands of forms. As seen in Table 1, new words have been generated by adding affixes to root words "*acı*" ("*acı*"="pain") and "*tanı*" ("*tanı*"="to know"). Morphology and syntax of Turkish (Ural-Altaic language) is significantly different when compared to English (Indo-European Language). Turkish language is characterized as a head-final language where the modifier/specifier always precedes the modified/specified. This characteristic also affects the word order of the sentences which can be described as subject-object-verb (SOV).

Table 1. Agglutinative morphology

Root Word	Generated Words	Meaning in English
"acı"	*"acı-mak"*	"to feel pain" (simple)
	"acı-n-mak"	"to grieve"(reflexive)
	"acı-n-dır-mak"	"to cause to grieve" (causative)
	"acı-n-dır-ıl-mak"	"to be made to grieve" (passive)
"tanı"	*"tanı-mak"*	"to know" (simple)
	"tanı-ş-mak"	"to know one another" (reciprocal)
	"tanı-ş-tır-mak"	"to introduce" (causative)
	"tanı-ş-tır-ıl-mak"	"to be introduced" (passive)

3 Integrated Architecture

Generally the process of IE starts by extracting individual facts from text documents through local text analysis, followed by integration of these facts for producing larger or new facts through inference. Finally the facts are integrated and translated into the required output format. Most early IE systems perform full syntactic analysis before looking for scenario patterns. In principle full sentence analyzers should be able to use global constraints to resolve local ambiguities, however full sentence parsers end up making poor local decisions. In addition, full sentence parsers are relatively expensive of computing time since they have a large space to search, therefore most IE approaches focus on relations only relevant to a specific scenario where correctly determining other relations may be a waste of time [3,4]. Document processing starts by tokenization. Input document (e.g. money transfer message) is tokenized and stored in an array structure. Row and column index of each token is stored to keep track of documents block boundaries and templates that are made of. Document blocks can be formed of either single or multiple lines. For instance, in a money transfer message, a fax header is a single line block (Fax-header = date→time→customer-name→phone-number→page-number), where branch details block is a multi-line block formed of a bank name followed by a branch name and a city name in separate lines, respectively.

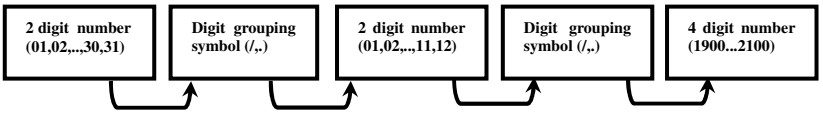


Fig. 1. Sample template for detecting date format "dd.mm.yyyy"

Named entity recognition starts by tagging each token in the input document by assigning base data types such as date, time and phone number before querying lexicon tables. For instance the system is supposed to tag the following date "23.11.2008". In the knowledge base, date format "dd.mm.yyyy" (day-month-year) is defined using the rule shown in Figure 1. If a token doesn't belong to any base data types, approach queries it in lexicon tables. (Banking domain lexicon tables: customer names, banks, city names, branch names, etc.).

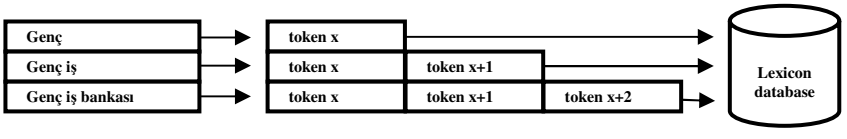


Fig. 2. Incremental lexicon search

As shown in Figure 2, a bank name *"Genç iş bankası"* is formed of there consecutive tokens (*"Genç"* ="Young", *"iş"*="Business" , *"bankası"*= "Bank"). Starting with token "*Genç*" system creates incremental concatenations of these tokens and queries each concatenated string in lexicon tables. Based on the lexicon integrated domain ontology approach picks correct interpretation *"Genç iş bankası***"** ("Young Business Bank"), which refers to a bank name, where first two concatenated strings don't fit any of the extraction concept fields.

3.1 Document Modeling and Layout Analysis

In order to process a broad range of documents, an IE system should be able to classify input documents. In this research documents are classified based on both structural and textual features. Test data consists of free form text documents (single-page business letters, emails) having identifiable layout characteristics. A document class is defined as a set of documents characterized by similarity of expressions, style, and page layout. Document layout analysis starts by grouping primitive elements of the input document into higher-level objects such as paragraphs and headings. Several research activities have been performed on recognizing document structure. Conway proposed a syntactic approach for deducing the logical structure of printed documents from their physical layout [5]. Hui et al. proposed a system where the contents of each fax message are recognized and stored according to their logical and layout formats in order to reduce the amount of storage required [6]. Klink et al. worked on document structure analysis based on layout and textual features. They used both layout (geometrical) features as well as textual features of a given document [7]. In proposed architecture, template matching technique is used to match blocks of an input

document with predefined document block templates. Similarly Ding et al. worked on template mining for extracting citation information from digital documents [8]. Matching an input document with all possible document block templates is a time consuming process, therefore computational complexity is reduced by hierarchical template matching. Proposed approach benefits a combination of Klink's and Ding's works. Template mining is used to for detecting document blocks, while document layout analysis is used for named entity recognition based on location of entities in a document.

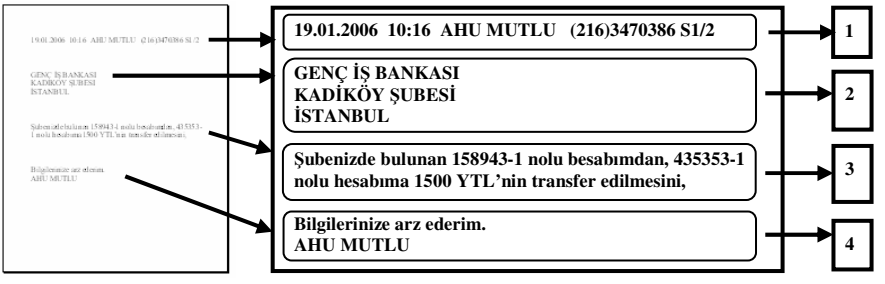


Fig. 3. Document blocks of a scanned (OCR) sample money transfer message

In Figure 3, a sample money transfer message in Turkish is grouped into main document blocks. Template matching method is applied to document blocks 1 (fax header), 2 (bank branch details) and 4 (sincerely yours block), where extraction ontology extracts money transfer event in block 3. Sample templates for detecting fax header block are shown in Table 2.

Table 2. Sample document block templates

Fax-Header = date→time→customer-name→phone-number→page-number
Fax-Header = date→time→phone-number →customer-name→page-number
Fax-Header = customer-name→phone-number→date→time→page-number

Sample document is formed of a single line fax header followed by a multi-line bank branch details block, followed by a free form text block containing money transfer details, finally followed by an ending block (e.g. sincerely yours block). A named entity (e.g. a customer name) may appear in different blocks in a document, and based on document layout analysis, system decides what these entities actually refer to. For instance, a customer name that appears in the header block refers to sender of the fax, where a customer name in the free form text block is most probably a parameter of a transfer action (e.g. receiver in money transfer scenario). System tries to predict an unknown named entity type based on document block templates. Assume that system locates customer name "*AHU MUTLU*" in a fax header (Table 3), but can't find it in available lexicon tables. System marks this customer as an unknown entity (Open Class). For predicting the correct tag of the unknown entity, approach finds the closest matching block template, by calculating the number of

Table 3. Unknown customer name captured in fax header template

Data	Detected Data Types	Closest Matching Block Template Fields
19.02.2006	Date	Date
10:12	Time	Time
AHU MUTLU	*Unknown*	*Customer Name*
(212) 347-0386	Fax Number	Fax Number
S1/2	Page Number	Page Number

matching token types surrounding the unknown entity. According to the closest matching template shown in Table 3, "*AHU MUTLU*" is considered as a customer name and added to the customers table automatically.

Experimental results show that, proposed approach assigns correct tags to open class entities based on document block templates with a success ratio of 99%. In addition to the document block templates, rule-base is powered by alternative formats of some of the base data types such as Page No ={P1 , P.01, S1/2, S.1, PAGE 1, PAGE.001}, Time = {09:32, 09.32 AM}, Fax/Tel= {90-212-3374782, 90 212 3374782, (212) 3374782}, Date= {23 OCT 08, OCT. 23 2008, 23.10.2008, 23/10/2008, EKI 23 2008}. Similar to this approach, Cardie worked on a case-based approach focusing on knowledge acquisition for domain specific sentence analysis, where it simultaneously learns part of speech and concept activation knowledge for all open class words in a corpus [9]. A commercial IE system ATRANS (Lytinen and Gershman) uses script approach for automatic processing of money transfer messages between banks [10]. Messages processed in ATRANS are structured telex message, where proposed architecture deals with both free-form and structured messages.

3.2 Domain and Extraction Ontologies

Extraction ontology is used for modeling domain specific events, while lexicon integrated domain ontology (build on Wordnet) is used for name entity recognition. Partial view of the domain ontology is shown in Figure 4. Several researchers focus on ontology-based IE. Yıldız and Miksch worked on an information extraction system which tries to automate the extraction-rule generation process, which currently seems to be the main obstacle to portable and scalable IE systems [11]. McDowell and Cafarella worked on an automatic and domain-independent ontology-driven IE system called OntoSyphon. It uses ontology to specify web searches that identify possible semantic instances, relations, and taxonomic information [12]**.** Temizsoy and Cicekci worked on an ontology-based approach for parsing Turkish sentences using morphological marks of Turkish which generally denote semantic properties [13]. Embley presented an approach for extracting and structuring information from data-rich unstructured documents using extraction ontology, where it parses the ontology to generate recognition rules for constants and context keywords for extracting structural and constraint information [14]. Wee et al. worked on generic information extraction architecture for financial applications. They have chosen a directed graph structure, a domain ontology and a frame representation respectively [15].

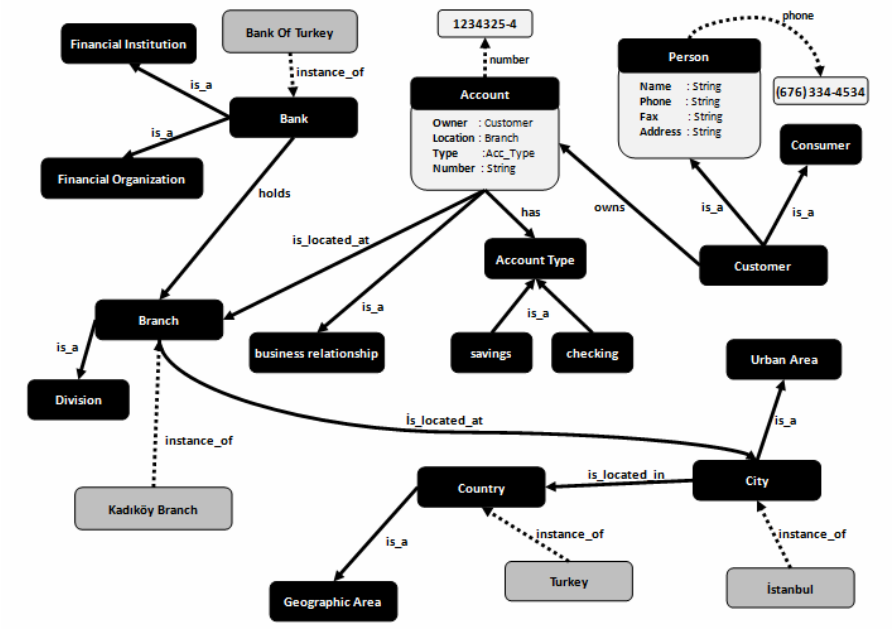


Fig. 4. Partial view of the domain ontology

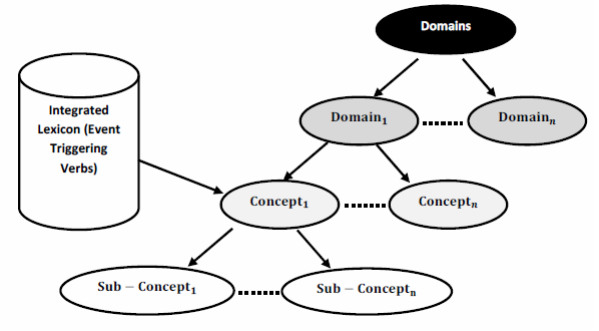


Fig. 5. Concept hierarchy

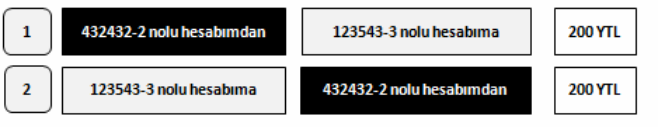
Wu et al. worked on domain event extraction and representation with domain ontology. The experimental results of their work demonstrate that the automatic event extraction and ontology acquisition can be good resources for text categorization and further information processing [1]. Proposed architecture aims to create an event extraction mechanism with minimum definition by using domain concepts, and reuse these concepts for creating a portable and scalable IE system. For extracting a domain specific scenario, approach locates concept triggering key-verbs and concept objects marked by the morphological analyzer and picks the best suiting concept in the extraction ontology. Turkish has free constituent order in a sentence causing numerous variations of linguistic patterns, where using sub-concepts minimizes the

Table 4. A sample money transfer concept in detail

Sub-Concept	Definition
Source	*"432432-2 nolu hesabımdan"* ("from my account with number 432432-2")
Destination	"123543-3 nolu hesabıma" ("to my account with number 123543-3")
Amount	*"200 YTL"* ("200 Turkish lira")

definitions required to capture concepts regardless of their location in a sentence. Concept hierarchy is shown in Figure 5. For instance "Money Transfer" is a concept in "Financial Documents" domain where source account, destination account and money amount are sub-concepts of the money transfer concept.

Simply there are six different ways to locate three sub-concepts of the money transfer concept, where sample sentences below (1,2) displays how sub-concepts can appear in different locations in a sentence. Experimental results demonstrate that conceptual sentence analyzer is significantly usefully especially when number of sub-concepts gets higher.



A sample sub-concept diagram generated by the concept editor is shown in Figure 6. Each sub-concept is defined by using a base data type, a lexical type or a combination of both. Source account in sample sentences shown above (1,2) fits the sub-concept in Figure 6 where property Account-Number = Account Number ("*432432-2*") +NoDef (*"nolu"*) and Source-Or-Destination= SourceAccPointer (*"hesabımdan"* - ablative). Based on extraction concepts, proposed approach breaks down the free-form text block into sub-concepts for minimizing computational complexity. As shown in Figure 6, Source-Account is the main node having properties such as location, type and account number. There are several ways to define a source account concept, defined as sub-concept variants. For instance, a source account sub-concept variant might be formed of Account Number + Type + Owner ("from my savings account with number 432432-5") or Account Number + Owner ("from my account with number 432432-5").

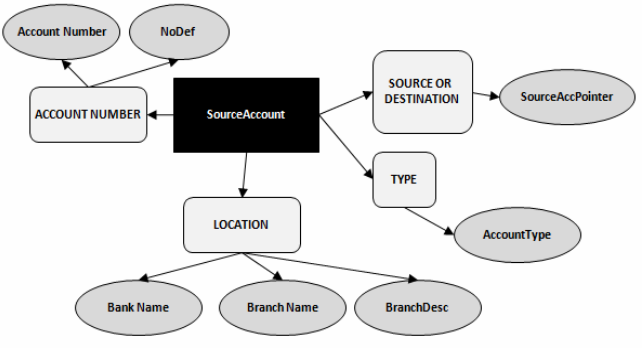


Fig. 6. Source account sub-concept designed using concept editor

3.3 Deeper Linguistic Analysis

Turkish is an agglutinative language where a sequence of inflectional and derivational morphemes gets affixed to a root [16]. It has a right dependent structure like Japanese where each determiner or modifier word appears before the word it modifies or determines (Figure 7). Head-final and right dependent structure of Turkish is useful especially for detecting noun-phrases, where these phrases (constituent) correspond to sub-concepts defined in the extraction ontology. Considering the following phrase "*Kadıköy şubenizdeki hesabımdan*" ("From my account at Kadıköy branch") "*Kadıköy şubenizdeki*" means "at Kadıköy branch" and "*hesabımdan"* means "from my account"). As shown in Figure 7, word "*Kadıköy*" determines word "*şubenizdeki*", and word "*şubenizdeki*" modifies word "*hesabımdan*".

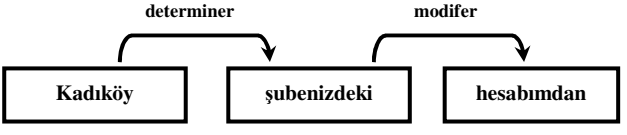


Fig. 7. Dependency relations between words in Turkish

In proposed architecture ontological knowledge and morphological analysis results are used together for domain filtering. Considering the sentence "*Adam para aktardı.*" ("The man transferred money."), "*Adam*" not only means "man" but also means "my island" in Turkish, where interpretation "The man transferred money." is valid since event denoted by "*aktar*" ("transfer") only accepts a person as an agent (in money transfer domain), which eliminates the other interpretation "My island transferred money". Table 5 shows how ontological knowledge and morphological analysis results are used together for domain filtering. Approach uses root of each word returned by the morphological analyzer for querying domain ontology. For instance word "*adam*" ("man") is queried in the ontology and tagged as person, where word "*ada*" ("island") is tagged as land.

Table 5. Transfer concept details

Word	Morph. Analysis	Definition	Type
adam	[root, adam] - [noun]	("man")	PERSON
adam	[root, ada] - [noun]	("my island")	LAND
para	[root, para] - [noun]	("money")	MEDIUM OF EXCHANGE
aktardı	[root, aktar] - [verb]	("transfer")	MOVE, DISPLACE

Sample analyzer results shown in Table 6 contain morphological details. For instance "*A3sg*" refers to 3.Singular, "*Pnon*" refers to Pronoun, "*Adj*" refers to adjective, etc. Considering tokens 3-6 (Table 6) where all tokens have the same root "*hesap*"("account") with different affixes, where "*hesaptan*" means "from the account", "*hesabımdan*" means "from my account", "*hesabından*" means "from your

Table 6. Morphologically analyzed tokens

No	Word	Morphological Analysis Results
1	Mumaralı	(numara+Noun+ A3sg+ Pnon+ Nom^DB+ Adj+ With)
2	Nolu	(no+Noun+ A3sg+ Pnon+ Nom^DB+ Adj+ With)
3	Hesaptan	(hesap+Noun+ A3sg+ Pnon+ Abl)
4	Hesabımdan	(hesap+Noun+ A3sg+ P1sg+ Abl)
5	Hesabından	(hesap+Noun+ A3sg+ P3sg+ Abl)
6	Hesabımızdan	(hesap+Noun+ A3sg+ P1pl+ Abl)

account", and "*hesabımızdan*" means "from our account". For money transfer messages this analysis is useful to distinguish between source and destination accounts, and to detect owner of an account.

4 Experimental Results

Approach is assigned a variety of scores by comparing its response to the answer keys. Let *NKey* be the total number of filled slots in the answer key, *NResponse* be the total number of filled slots, and *NCorrect* be the number of correctly filled slots. The primary scores precision and recall are shown below.

$$\text{Precision}=\frac{\text{NCorrect}}{\text{NResponse}} \qquad \text{Recall}=\frac{\text{NCorrect}}{\text{NKey}} \qquad \text{F-Score}=\frac{(2*\text{precision}*\text{recall})}{(\text{precision}+\text{recall})} \tag{1}$$

An additional score which is called *F-Score* is used as combined recall-precision score. Proposed architecture is tested for automatic processing of 2000 financial documents in Turkish. These documents include a random combination of 27 different financial scenarios, 6 fax header block type, 3 branch detail block type and 3 ending block type. System extracts information such as date, time, fax number, currency type, customer name, account number and money amount related to a specific scenario. Tests are performed on a system with Windows Vista operating system, Intel Duo Core 2.20 GHz cpu and 2048Mb ram. Average time for processing a single message is around 10 seconds. Approach is tested incrementally using 500, 1000 and 2000 documents respectively. It underwent three different tests for demonstrating how proposed architecture increases F-Score. First, the system is tested using a limited number of extraction concepts and document block templates. Event concepts and document block templates that can't be detected in each run are reported and added incrementally before testing next document set. Test results indicate that the system has to cover enough number of event concepts and document block templates, in order to obtain successful extraction results. Detailed test results are shown in Table 7. Tür's statistical work has an F-score 93% for tagging named entities (only person, location and organization) using lexical, contextual, morphological and tag model together. Contrast to proposed architecture, it doesn't deal with ontology based information extraction and document layout analysis.

Table 7. Performance comparison

Test	Incremental Performance Analysis	F-Score (500 documents)	F-Score (1000 documents)	F-Score (2000 documents)
1	Basic IE	86	89	90
2	Ontology Based IE	92	93	95
3	Ontology-Based IE & Document Layout Analysis	97	98	99

5 Conclusions

Proposed architecture relies on a document layout analyzer, a lexicon integrated domain ontology and an extraction ontology. Experimental results demonstrate that it works well when applied to well-defined, restricted domain applications. Conceptual sentence analyzer enables easy definition of extraction concepts and increases portability of the IE system. Currently, proposed architecture is being tested for automatic processing of frequent flyer retro claim (missing flight) emails. Test results for this domain is so far very close to banking domain results, which is successful in terms of F-score and portability. In addition, system queries relational database (e.g. A CRM system) for verifying extracted information by using key value(s) (e.g. customer name, fax number) defined inside a document model. This method not only verifies extracted information against real world data, but also verifies relations between entities and events. We believe that this study would be a good starting point for researchers interested in automatic document processing and information extraction in Turkish.

References

1. Wu, S., Tsai, T., Hsu, W.: Domain Event Extraction and Representation with Domain Ontology. In: Proceedings of the IJCAI 2003 Workshop on Information Integration on the Web, pp. 33–38 (2003)
2. Tür, G.: A Statistical Information Extraction System for Turkish. Ph.D. Thesis, Bilkent University, Department of Computer Engineering, Turkey (2000)
3. Gaizauskas, R., Wilks, Y.: Information Extraction: Beyond Document Retrieval. Computational Linguistics and Chinese Language Processing 3(2), 17–60 (1998)
4. Grishman, R.: Information Extraction: Techniques and Challenges. In: Pazienza, M.T. (ed.) SCIE 1997. LNCS, vol. 1299, pp. 10–27. Springer, Heidelberg (1997)
5. Conway, A.: A Syntactic Approach to Document Layout Recognition. In: Proceedings of the 2nd International Conference on document analysis and recognition, pp. 761–764 (1993)
6. Hui, S.C., Chan, K.Y., Leung, M.K., Qian, G.Y.: A Distributed Fax Messaging System. Journal of Network and Computer Applications 20(2), 171–190 (1997)
7. Klink, S., Andreas, D., Kieninger, T.: Document Structure Analysis Based on Layout and Text Analysis. In: Proceedings of International Workshop on Document Analysis Systems, pp. 99–111 (2000)
8. Ding, Y., Chowdhury, G., Foo, S.: Template Mining for the Extraction of Citation from Digital Documents. In: Proceedings of the 2nd Asian Digital Library Conference, pp. 47–62 (1999)

9. Cardie, C.: A Case-Based Approach to Knowledge Acquisition for Domain-Specific Sentence Analysis. In: Proceedings of the 11th National Conference on Artificial Intelligence, pp. 798–803 (1993)
10. Lytinen, S., Gershman, A.: ATRANS: Automatic Processing of Money Transfer Messages. In: Proceedings of the 5th National Conference of the American Association for Artificial Intelligence, pp. 93–99 (1993)
11. Yildiz, B., Miksch, S.: Motivating Ontology-driven Information Extraction. Indian Statistical Institute Platinum Jubilee Conference Series, pp. 45–53 (2007)
12. Mcdowell, L.K., Cafarella, M.: Ontology-driven Information Extraction with OntoSyphon. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 428–444. Springer, Heidelberg (2006)
13. Temizsoy, M., Cicekci, I.: An Ontology-based Approach to Parsing Turkish Sentences. In: Farwell, D., Gerber, L., Hovy, E. (eds.) AMTA 1998. LNCS, vol. 1529, pp. 124–135. Springer, Heidelberg (1998)
14. Embley David, W., Campbell Douglas, M., Smith Randy, D.: Ontology-based Extraction and Structuring of Information From Data-rich Unstructured Documents. In: Proceedings of the Conference on Information and Knowledge Management, pp. 52–59 (1998)
15. Wee, L.K.A., Tong, L.C., Tan, C.L.: A Generic Information Extraction Architecture for Financial Applications. International Journal of Expert Systems with Applications 16(4), 343–356 (1999)
16. Oflazer, K.: Two-level Description of Turkish Morphology: Literary and Linguistic Computing 9(2), 137–148 (1995)

Detecting Protein-Protein Interactions in Biomedical Texts Using a Parser and Linguistic Resources

Gerold Schneider, Kaarel Kaljurand, and Fabio Rinaldi

Institute of Computational Linguistics, University of Zurich, Switzerland
{gschneid,kalju,rinaldi}@ifi.uzh.ch

Abstract. We describe the task of automatically detecting interactions between proteins in biomedical literature. We use a syntactic parser, a corpus annotated for proteins, and manual decisions as training material.

After automatically parsing the GENIA corpus, which is manually annotated for proteins, all syntactic paths between proteins are extracted. These syntactic paths are manually disambiguated between meaningful paths and irrelevant paths. Meaningful paths are paths that express an interaction between the syntactically connected proteins, irrelevant paths are paths that do not convey any interaction.

The resource created by these manual decisions is used in two ways. First, words that appear frequently inside a meaningful paths are learnt using simple machine learning. Second, these resources are applied to the task of automatically detecting interactions between proteins in biomedical literature. We use the IntAct corpus as an application corpus.

After detecting proteins in the IntAct texts, we automatically parse them and classify the syntactic paths between them using the meaningful paths from the resource created on GENIA and addressing sparse data problems by shortening the paths based on the words frequently appearing inside the meaningful paths, so-called transparent words.

We conduct an evaluation showing that we achieve acceptable recall and good precision, and we discuss the importance of transparent words for the task.

1 Introduction

Scientific articles reporting results of biomedical studies are growing exponentially in number[1]. Publically available literature services such as Pubmed (http://pubmed.gov) already contain more than 17 million articles. Even for the expert it has become difficult to keep an overview of new results. Fully or partly automated systems that extract biological knowledge from text have thus become a necessity. Particularly, knowledge about protein-protein interactions

[1] This research is partially funded by the Swiss National Science Foundation (grant 100014-118396/1). Additional support is provided by Novartis Pharma AG, NITAS, Text Mining Services, CH-4002, Basel, Switzerland.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 406–417, 2009.

(PPI) is needed in biomedical and genetic research, as exemplified by the LLL genic interaction challenge [1] and the BioCreAtIvE challenge PPI track [2].

A number of methods have been applied to this task. Simple approaches classify two proteins as interacting when mentioned in the same sentence, or when their cooccurrence in an abstract is very frequent [3]. Such approaches often yield high recall at low precision and can be used as baselines for more involved approaches.

Other approaches apply handcrafted rules, for example regular expressions for surface searches [4], or syntactic patterns on automatically parsed corpora [5,6]. These approaches typically achieve high precision at the cost of recall.

Third, machine learning methods are increasingly used to construct a model from large annotated sources. To extract meaningful features for the model construction, dependency parsing is often used. [7] extract sentences in which two proteins and an interaction word co-occur. Their features include the interaction words and the parents of the proteins, according to the dependency analysis. [8] use a walk kernel which contains fragments of the paths between two proteins. Due to sparse data, the paths were partitioned into patterns, each consisting of two vertices and their intermediate edge (*vertex-walk*), and of two edges and their common vertex (*edge-walk*). While this alleviates the sparse data problem, it neglects that many semantic configurations are not local, they depend on considerably larger tree fragments. We suggest to use a single feature consisting of the entire path, but to reduce reduce sparseness by using very little lexical information and linguistic insights to shorten the paths.

[9] extends the approach of [8] by using a feature-based approach instead of a kernel, where e.g. each vertex-walk and each edge-walk is a feature, on the one hand a lexical feature containing words, on the other hand a syntactic feature containing tags. The lexical features are quite sparse due to Zipf's law.

The approach that we present in this paper is hybrid. It uses a large, partly annotated resource and manual annotations. It uses parsed data in order to obtain a suitable level of abstraction and reducing the number of manual annotation decisions, thus creating a new linguistic resource. It achieves higher precision than coocurrence methods because it uses stricter requirements. It achieves higher recall than handcrafted syntactic patterns because all syntactic connections that are observed in a large corpus are taken into consideration. Machine Learning methods and backoff techniques are applied to the linguistic resource thus created.

For training, we have used the GENIA corpus, to which we have manually added interaction information. Our approach shows a new application of the GENIA corpus. For the application phase, we use the IntAct corpus[10]. The IntAct corpus was devised to be used for the PPI task but has been underused so far.

The aim of our application is twofold. On the one hand, we use the IntAct data as a gold standard for evaluating the performance of our PPI algorithm, on the other hand we propose an algorithm that may help IntAct annotators by suggesting protein-protein interactions to them.

Our approach is also characterised by using linguistic insights and lightweight resources, allowing us to achieve good results despite using simple statistical methods and learning algorithms.

The paper is structured as follows. We summarise our term detection and grounding method in chapter 2. In chapter 3, we describe how we collect and annotate the syntactic data. In chapter 4, our application to the IntAct corpus is described. We give an evaluation in chapter 5 and conclude in chapter 6.

2 Term Detection and Grounding

Term detection and grounding is a necessary preliminary step to the detection of interaction. Our approach is described in detail in [11] and summarised here.

We compiled a term list of 1,685,126 terms based on the terms extracted from UniProtKB[2], NCBI[3], and PSI-MI[4], The term list contains the term name, the term ID, and the term type in each entry. In this list, 934,973 of the terms are multi-word units.

2.1 Automatic Term Detection

Using the described term list, we can annotate biomedical texts in a straightforward way. First, the sentences and tokens are detected in the input text. We use the LingPipe tokenizer and sentence splitter which have already been trained on biomedical corpora. The term detector matches the longest possible and non-overlapping sequences of tokens in each sentence, and in the case of success, assigns all the possible IDs (as found in the term list) to the annotated sequence. The annotator ignores certain common English function words (we use a list of about 50 stop words). Also, figure and table references are detected and ignored.

In order to account for possible orthographic differences between the terms in the term list and the token sequences in the text, a normalization step is included. We apply standard normalization rules, such as removing all characters that are neither alphanumeric nor space, normalising Greek letters and Roman numerals, convert to lower case, apply biomedical normalizations, such as removing the final 'p' if it follows a number, e.g. 'Pan1p' → 'Pan1'.

2.2 Automatic Term Grounding

A marked up term can be ambiguous. The most frequent reason is that the term can be assigned several IDs from a single type. This happens very often with UniProtKB terms and is e.g. due to the fact that the same protein occurs in many different species. Such protein names can be disambiguated in various ways. We have combined two methods: (1: 'IntAct' in table 1) remove all the IDs that do not reference a species ID specified in a given list of species IDs,

[2] http://www.uniprot.org
[3] http://www.ncbi.nlm.nih.gov/Taxonomy/
[4] http://psidev.sourceforge.net/mi/psi- mi.obo

Table 1. Evaluation of term detection and grounding on IntAct, measured against PubMed IDs. Two forms of disambiguation were applied: IntAct = species list from Intact data; span = species of neighbouring proteins must match

Diamb. method	Precision	Recall	F-Score
No disamb.	3%	73%	5%
IntAct	56%	73%	63%
span	3%	71%	6%
IntAct & span	57%	72%	64%

in this case the species found in the IntAct data; (2: 'span' in table 1) remove all IDs that do not agree with the IDs of the other protein names in the same textual span (e.g. sentence) with respect to the species IDs.

The 'span' method is motivated by the fact that according to the IntAct database, interacting proteins are usually from the same species: less than 2% of the listed interactions have different interacting species. Assuming that proteins that are mentioned in close proximity often constitute a mention of interaction, we used a simple disambiguation method: for every protein mention, the disambiguator removes every UniProtKB ID that references a species that is not among the species referenced by the IDs of the neighbouring protein mentions (we use same sentence as neighbourhood).

The disambiguation result is not always a single ID, but often just a reduced set of IDs. Also, it can happen that none of the IDs matches a listed species. In this case all the IDs are removed.

We evaluated the accuracy of our automatic protein name detection and grounding method on a corpus provided by the IntAct project[5]. This corpus contains a set of 6198 short textual snippets (of 1 to about 3 sentences), where each snippet is mapped to a PubMed identier (referring to the article the snippet originates from), and an IntAct interaction identifier (referring to the interaction that the snippet describes). In other words, each snippet is a textual evidence that has allowed the curator to record a new interaction in the IntAct knowledge base. By resolving an interaction ID, we can generate a set of IDs of interacting proteins and a set of species involved in the interaction, for the given snippet. Using the PubMed identifiers, we can generate the same information for each mentioned article. By comparing the sets of protein IDs reported by the IntAct corpus providers, and the sets of protein IDs proposed by our tool, we can calculate the precision and recall values, as given in table 1.

3 Collection and Annotation of Syntactic Data

3.1 Parsing and Tree Walks

The GENIA corpus has been manually annotated for biomedical terms and proteins. It consists of 2,000 abstracts, containing over 18,000 sentences. We parse

[5] `ftp://ftp.ebi.ac.uk/pub/databases/intact/current/various/data-mining/`

the GENIA corpus with a state-of-the art dependency parser which has been adapted to and evaluated on the biomedical domain [12,13].

Figure 1 shows the output of the parser for the sentence.

Significant amounts of Tom40 were also coprecipitated by anti - Tom20 and anti - Tom22.

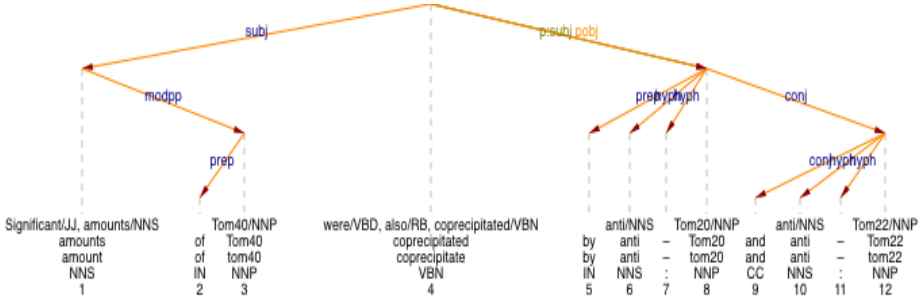


Fig. 1. Dependency parser output example

After parsing, we collect all syntactic connections that exist between all the terms as follows. For each term-coocurrence, i.e. two terms appearing in the same sentence, a collector traverses the tree from one term up to lowest common mother node, and down the second term, recording all intervening nodes. Such traversals have been used in many PPI applications [8], they are commonly called tree walks or paths. If one records all the information that an intermediate node contains, for example its lexical items and subnodes, the path would be extremely specific, which leads to sparse data and hence a recall problem for most applications. If one only records the grammatical role labels, the paths are too general, which leads to a precision problem for most applications. As a working assumption, we have recorded the lexical head lemma of the top node, and the grammatical labels plus prepositions connecting all intervening nodes. We have split the path into a left and a right half between the top node. The sentence in figure 1 contains 3 proteins: *Tom40*, *Tom20*, and *Tom22*. The path between *Tom40* and *Tom22* consists of the top node *coprecipitate*, the left path [*subj,modpp-of*] and the right path [*p:subj,conj*]. The path is treated as a single feature, unlike in most similar approaches, e.g. [8]. They use a a kernel with fragments of the of the paths between two proteins. Each pattern consists of two vertices and their intermediate edge (*vertex-walk*), and of two edges and their common vertex (*edge-walk*). While this alleviates the sparse data problem, it neglects that many semantic configurations are not local, they depend on considerably larger tree fragments.

We suggest to use a single feature consisting of the entire path, but using very little lexical information and linguistic insights to shorten the paths. in [9], each vertex-walk and each edge-walk leads to two features, on the one hand a lexical feature containing words, on the other hand a syntactic feature containing tags. The lexical features are quite sparse due to Zipf's law. There is a small

closed class of lexical items that is crucial to syntax [14,15], namely prepositions, which we have thus introduced into the path. But also the syntactic features are potentially sparser than what is linguistically meaningful, as they contain tags. A subject relation, for example, is mostly between a noun and a verb. Since there are 4 noun tags and almost a dozen verb tags, sparseness is inflated. Our paths are also shorter and less sparse than in many other representations, because the syntactic graphs that we use are based on chunks. We present linguistic insights that further allow us to reduce data sparseness by shortening paths in section 4.2.

3.2 Manual Annotation

Only a minority of the paths extracted by the method just introduced actually express a biomedical interaction. The path between *Tom20* and *Tom22* in our example, which consists of the top node *Tom20*, the empty left node, and the right node [*conj*], does not express any biomedical interaction, it does not state in any way that *Tom20* and *Tom22* interact. In order to apply the paths to the PPI task we need to classify paths into those expressing an interaction relation and those that do not. We have decided to classify manually.

Ideally, one should classify every individual co-occurrence of two terms in the entire corpus. Since we did not have the resources to conduct such a large-scale, token-based annotation, we have opted for a type-based annotation at the level of the extracted paths. If our working assumption that these paths are a useful level of abstraction holds, the annotation task offers a useful compromise in the trade-off between token-wise annotation and unsupervised machine learning. We have discarded singletons, i.e. paths only appearing once in GENIA, since they are too sparse and often arise from parsing errors. The frequency-ranked list of paths tails off sharply, indicating a Zipfian distribution, more than half of all paths are singletons.

A major advantage of annotating a large corpus over formulating hand-written patterns is that no instance is missed (except for very rare ones that happen to be absent from a large corpus). This insight has given rise to the methodology of corpus linguistics in descriptive linguistics.

We manually annotated the about 2500 paths appearing at least twice. Each decision, i.e. whether the target path expresses a relation or rather not, was based on at least three example sentences[6] containing the target path. During the annotation we observed that there are relatively few paths for which the example sentences suggested opposite decisions. We also observed that many paths express subset relations, for example *A is a B protein*, where *A* is a subset of *B*. We have decided to annotate these cases with a third class in addition to 'yes' and 'no', saving them for future ontology applications. Additionally, we observed that semantically lightweight nouns seem to play an important role for the decision: in order to test if a relation is expressed by an example sentence, it typically helps to paraphrase it, using the top node and the two terms. The sentence *A activates groups of B* essentially expresses that *A activates B*, or *A blocks activation of B*, or expresses that *A blocks B*, whereas *A activates C,*

[6] Except for paths that only appeared twice in GENIA.

which has a binding site for B does not express that *A activates B*. There is a large set of words like *group* and *activation*, for which we would like to use the term *transparent words*. We complied lists of them and extended it with a simple machine-learning approach described in section 3.3.

We noticed that there are some paths, especially relatively long ones, for which it is very difficult to decide. When asked for the decision if *D3* interacts with *CD28*, based on the following example sentence, we found it hard to decide and agreed to opt for 'no' in difficult cases.

Further, engagement inhibited progression through the cycle by inhibiting the production of D3, kinase cdk, and cdk6 when the cells were stimulated with CD28 and with anti-CD3 alone.

3.3 Learning Transparent Words from the Type-Based Annotation

Although we collected the paths based only on the head lemma of the top node and the labels of intervening nodes, we also kept record of all intervening words in order to be able to learn specific rules where necessary. All the words intervening inside a path are, for instance, candidates for being transparent words, as introduced in section 3.2. For each word appearing inside a path, we calculate a score which simply divides its frequency inside a path by its total frequency. Words above a threshold are treated as transparent in the application phase.

4 Application to IntAct

The paths that are extracted from GENIA can directly be used for PPI detection. We chose the IntAct data as a gold standard. Although we have constructed the paths in a way that aims to reduce sparseness, and although we have used a corpus-based annotation method instead of introspective creation of handcrafted rules, recall is very poor when the patterns are applied directly. The following are the main reasons why recall is low. In each subsection we also discuss how we have improved the situation.

4.1 Term Recognition and Upper Bound

The protein detectiontion and grounding algorithm which we use (section 2) has a recall of about 72%. Since any interaction involves two proteins, the performance for recognition and grounding of protein pairs can be expected to be about 50% recall. It is beyond the scope of this paper to improve term recognition performance, so we need to accept this upper bound for the PPI task[7].

4.2 Transparent Words

Sparse data problems could be reduced significantly by applying the transparent words resource that we have created. If no annotated path from GENIA exists, the following sparse data reduction methods are used as backoffs:

[7] Baseline 1 in the evaluation section calculates the exact upper bound.

– First, proteins occurring inside noun chunks are allowed to replace the head of the chunk if the head is an transparent word.
– Secondly (if still no path from GENIA exists), the relations for appositions, conjunctions and hyphens are cut.
– Third (if still no path from GENIA exists), parts of trees that are headed by an transparent word are cut.

In the example sentence 2 cutting conjunctions (second backoff) means that *portion of Tim54* appears at the same level as *Tim12*, cutting the transparent words *portion* and *all* (third backoff) means that *Tim54* appears at the same level as *Tim12*, and *Tim22* is only one PP-attachment (*modpp*) lower.

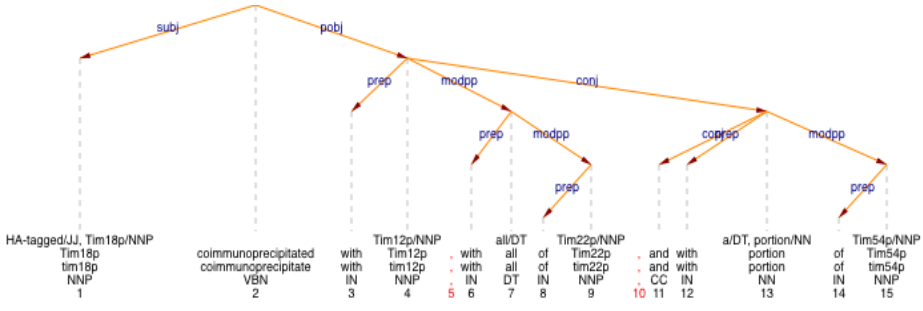


Fig. 2. Dependency parser output example

4.3 Surface Patterns to Address Tagging and Parsing Errors

Tagging and parsing errors are quite frequent, despite using taggers and parsers that are adapted to the domain. They are an important reason for the remaining sparseness. We have therefore developed surface patterns. They apply only at a backoff level, i.e. if no syntactic path is found in GENIA even after the syntactic backoffs.

There are three patterns. Each pattern consists of two proteins and a keyword:

– A verb B, e.g *A interacts-with B*
– noun A B, e.g. *association between A and B*
– A B noun, e.g. *A - B binding*

The distance between *A* and the keyword, as well as the distance between *B* and the keyword are restricted, as is typical for observation window-based approaches. These surface patterns typically achieve relatively good precision, but insufficient recall. If observation windows are very large, recall increases but precision drops off.

5 Evaluation

We have evaluated our approach, as well as an upper bound and a number of baselines, in order to measure the relative success of our approach. We have used

the first 1000 sentences of the IntAct data for the evaluation. We have mentioned that, given the performance of about our term recognition and grounding tool, the upper bound is about 50% recall. The term grounding tool sometimes delivers one UniProt ID and sometimes several UniProt IDs, on average 2.02 IDs. Since the ultimate aim of our approach is to deliver one and exactly one ID we speak of exact precision and recall if both proteins are given only one ID, and if there is an interaction, and of loose precision and recall if one or both of the proteins are given more than one ID by the grounding algorithm (i.e. if the UniProt ID could not be fully disambiguated), if one of the delivered IDs is correct for each protein, and there is an interaction. UniProt IDs are very fine-grained, including the organism in which the protein functions (ortholog). Since the task described in this paper is interaction detection rather than full term grounding disambiguation, we will mainly report loose precision and recall figures.

Our currently best system achieves 80.5% loose precision and 21.0% loose recall, and 59% exact precision at 15% exact recall. In order to assess the relative success that these performance figures mean, we will now compare them to a number of increasingly more advanced baselines.

5.1 Baselines

Baseline 1. Cooccurrence of two proteins in a sentence is a low baseline, one that heavily overgenerates. Precision of this baseline tells one how much one gets for free, while recall tells one how good one can maximally get (upper bound). We achieve 39.2% loose precision and 41.2% loose recall. Compared to this baseline, our best system has more than doubled precision at the cost of losing about half of the recall.

Baseline 2. In a purely syntactic approach, measuring all syntactically connected proteins lead to a second baseline. Since the parser we use does not always deliver analysis spanning the entire sentence, especially when sentences are complex, this is not a variant of baseline 1. We achieve 50.1% loose precision and 29.8% loose recall with baseline 2.

Baseline 3. In a purely "non-syntactic" surface based approach, observation windows are often used. We apply the surface patterns introduced in section 4.3, but no paths, and do not use information on transparent words. The window size is 5 words, which means that maximally 3 words may occur between the head (e.g. the verb) and the term. We achieve 78.6% loose precision and 11.4% loose recall with baseline 3.

Baseline 4. We extend baseline 3 by using the transparent words resource that we have created. This baseline is still purely surface-based and window size is 5, but transparent words are cut from the observation window, which means that remote words may move into the observation window if they are mainly separated by transparent words. We achieve 81.8% loose precision and 18.7% loose recall with baseline 4.

Table 2. Baselines compared to the system

Method	Description	Loose Precision	Loose Recall
Baseline 1	sentence cooccurrence	39.2%	41.2%
Baseline 2	syntactically conncected	50.1%	29.8%
Baseline 3	surface, no transparent words	78.6%	11.4%
Baseline 4	surface, transparent words	81.8%	18.7%
Best system	syntax, surfaces, transparent words	80.5%	21.0%

Best system. The currently best system achieves 80.5% precision and 21.0% loose recall. It uses syntactic patterns, surface patterns and the transparent words resource at both levels.

The step-wise improvements from the baselines to the currently best system are summarised in table 2. The performance of the last baseline, surface-based but using transparent words, is impressive. Adding the transparent words resource to the system increased performance more than the syntactic filter which the best system uses. The best system achieves 51% of the upper bound recall in baseline 1. Given gold-standard term information, the system therefore, all other parameters being equal, achieves a performance of 80.5% precision and 51% recall on the PPI detection task, which amounts to an F-score of 62.4%.

5.2 Breakdown of Results

We have broken down the precision results in table 3, which also quantifies the backoff method we use. If a syntactic method gives a decision, it is used, otherwise same chunk is applied. If that does not give a decision, the surface patterns are used. When available, the syntax-based method delivers the highest precision, but the surface method with the transparent words resource performs almost equally well. Absolute numbers are given in the third column.

We have tested a further syntactic backoff, which allows a path with a different top node to be used. Precision of this backoff was between 50 and 60%, lower than the surface backoff. We have also constrained the syntactic backoff using Wordnet, e.g. enforcing that the top node word for which a path was found in GENIA is similar to the candidate top node word, but precision did not increase.

Table 3. Breakdown of results

Backoff used	Loose Precision Percent	Loose Precision Count
Syntax	83.8%	62/74
Same chunk	75%	3/4
surface A verb B	76%	38/50
surface noun A B	82.4%	14/17
surface A B noun	66.7%	2/3
TOTAL	80.5%	120/149

6 Conclusions

We have created three new resources: annotated paths from the GENIA corpus, automatically learnt transparent words, and transparent words noted while annotating and testing. We have applied the resources to IntAct, both as a PPI task, and in order to develop an algorithm helping annotators.

We have evaluated our algorithm and performed better than all baselines. On the PPI task, our best system achieves 80.5% precision and 21% recall. 21% recall corresponds to 51% of the upper bound, if gold standard term recognition were used. We have based our path representations on linguistic insights. We use syntactic paths as features with very little lexical information (only the top node word and prepositions in PPs), and based on chunks, both of which lead to fewer sparse data problems. We have shown that transparent words, words with low semantic content, play an important role in allowing us to further reduce sparseness: we have cut transparent words and their nodes from our path representations. Further reducing sparseness by also excluding the top node word negatively affected performance.

References

1. Nedellec, C.: Learning language in logic – genic interaction extraction challenge. In: Proceedings of LLL 2005, pp. 31–37 (2006)
2. Krallinger, M., Leitner, F., Rodriguez-Penagos, C., Valencia, A.: Overview of the protein-protein interaction annotation extraction task of biocreative ii. Genome Biology 9 (suppl. 2) (2008)
3. Rebholz-Schuhmann, D., Kirsch, H., Arregui, M., Gaudan, S., Riethoven, M., Stoehr, P.: EBIMed – text crunching to gather facts for proteins from Medline. Bioinformatics 23(2), 237–244 (2006)
4. Giuliano, C., Lavelli, A., Romano, L.: Exploiting shallow linguistic information for relation extraction from biomedical literature. In: Proceedings of EACL 2006 (2006)
5. Rinaldi, F., Schneider, G., Kaljurand, K., Hess, M., Romacker, M.: An environment for relation mining over richly annotated corpora: the case of GENIA. BMC Bioinformatics 7(suppl. 3) (2006)
6. Fundel, K., Küffner, R., Zimmer, R.: RelEx – relation extraction extraction using dependency parse trees. Bioinformatics 23(3), 365–371 (2007)
7. Erkan, G., Ozgur, A., Radev, D.R.: Extracting interacting protein pairs and evidence sentences by using dependency parsing and machine learning techniques. In: Proceedings of BioCreAtIvE 2 (2007)
8. Kim, S., Yoon, J., Yang, J.: Kernel approaches for genic interaction extraction. Bioinformatics 9(10) (2008)
9. Landeghem, S.V., Saeys, Y., de Peer, Y.V.: Extracting protein-protein interactions from text using rich feature vectors and feature selection. In: Proceedings of the Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008), Turku, Finland (2008)

10. Kerrien, S., Alam-Faruque, Y., Aranda, B., Bancarz, I., Bridge, A., Derow, C., Dimmer, E., Feuermann, M., Friedrichsen, A., Huntley, R., Kohler, C., Khadake, J., Leroy, C., Liban, A., Lieftink, C., Montecchi-Palazzi, L., Orchard, S., Risse, J., Robbe, K., Roechert, B., Thorneycroft, D., Zhang, Y., Apweiler, R., Hermjakob, H.: Intact: open source resource for molecular interaction data. Nucleic Acids Res. (35 Database), D561–D565 (2006)
11. Kaljurand, K., Rinaldi, F., Kappeler, T., Schneider, G.: Detecting and grounding terms in biomedical literature. In: CICLing 2009, 10th International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico (2009)
12. Schneider, G., Kaljurand, K., Rinaldi, F., Kuhn, T.: Pro3Gres parser in the CoNLL domain adaptation shared task. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, Prague, pp. 1161–1165 (2007)
13. Haverinen, K., Ginter, F., Pyysalo, S., Salakoski, T.: Accurate conversion of dependency parses: targeting the stanford scheme. In: Proceedings of Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008), Turku, Finland (2008)
14. Collins, M., Brooks, J.: Prepositional attachment through a backed-off model. In: Proceedings of the Third Workshop on Very Large Corpora, Cambridge, MA (1995)
15. Collins, M.: Head-driven statistical models for natural language parsing. Computational Linguistics 29, 589–637 (2003)

Learning to Learn Biological Relations from a Small Training Set

Laura Alonso i Alemany and Santiago Bruno

NLP Group
Facultad de Matemática Astronomía y Física (FaMAF), UNC
Córdoba, Argentina

Abstract. In this paper we present different ways to improve a basic machine learning approach to identify relations between biological named entities as annotated in the Genia corpus.

The main difficulty with learning from the Genia event-annotated corpus is the small amount of examples that are available for each relation type. We compare different ways to address the data sparseness problem: using the corpus as the initial seed of a bootstrapping procedure, generalizing classes of relations via the Genia ontology and generalizing classes via clustering.

1 Introduction and Motivation

The huge amount of biomedical research papers available nowadays makes intelligent information access a necessity. In this paper we develop a method to assist information retrieval for highly focused information needs. In particular, we develop a module to detect relations between biological entities in free text. The work presented here is part of the MicroBio project[1], which aims to build a system for information retrieval of biomedical research papers. This module will be inserted in a wider system for information retrieval of biomedical research papers. This module will provide the capability of querying a database to find documents containing a particular relation between biological entities.

Given the importance of biomedical research, much effort has been devoted to the problem of biomedical information access. Very good results have been obtained for the recognition and classification of biological named entities (Bio-NER). In contrast, approaches to the problem of discovering relations between biological entities have been more rare and less successful, probably because the problem is more complex.

To begin with, there are less and smaller corpora annotated with relations, which constitutes a considerable bottleneck for supervised machine learning approaches to the problem. Unsupervised machine learning approaches do not suffer from lack of training data, but are normally incapable of detecting fine-grained distinctions in relations. As for symbolic approaches, they suffer from

[1] http://www.microbioamsud.net

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 418–429, 2009.

very small coverage, are not easily scalable or portable and are expensive in terms of human effort, which is why we do not consider them from the start.

We try to overcome the shortcomings of supervised and unsupervised methods by combining them in a bootstrapping approach. We use an annotated corpus as a starting seed to learn a high precision, low coverage tool to identify relations. This tool will be used to identify relations in unannotated corpora. Those instances of relations that the tool can reliably identify and classify will be incorporated to the seed annotated corpus, and the relation annotation tool will be learnt again from this bigger set of examples. This procedure will go on iteratively until no improvement is obtained.

Bootstrapping methods have been successfully applied to deal with deep Natural Language Processing problems involving meaning (e.g., Yarowsky 1995). However, one must be very careful in the design of the approach, since there are some crucial points that may make the whole method useless. Bootstrapping is especially sensitive to the starting seed of knowledge from which it departs: if the space of the problem is not well represented by this seed, it is very hard for bootstrapping methods to get to cover the whole space. The second crucial aspect of bootstrapping is how the starting seed is grown: if one is not careful enough, subsequent models can diverge from the targeted model represented by the starting seed, ending up in something rather different.

In this paper we focus on the first of these two crucial aspects. We present various experiments aimed at finding a good bootstrapping-oriented tool from the initial annotated corpus. The main objective is to introduce generalization in this annotated corpus, so that the starting model does not overfit the corpus, while retaining important distinctions. Generalization allows the application of the model to unseen examples, which makes it easier to grow the starting corpus with previously unseen examples. At the same time, generalization alleviates the severe data sparseness problem, and helps obtain more reliable models, based on more significant statistical evidence.

We apply different strategies to generalize the corpus: feature selection and generalizing the classes of relations via an ontolgy or via clustering. The latter strategy, generalizing relations via clustering, provides the best improvement in accuracy, at the cost of losing some useful distinctions.

Aiming at wide applicability, our model is based on shallow features of the annotated examples. Thus, the system relies on robust, easily available tools such as stemmers or part of speech taggers, and not on more sophisticated, error-prone tools like parsers. Since we are focusing on relation extraction, it will be assumed that named entities have already been identified in the corpus, for which reliable tools are available for the biomedical domain.

We are using Weka (Witten and Frank 2005) to carry out the experiments to find the adequate classifier. Weka provides a good choice of supervised and unsupervised learning methods for us to explore different alternatives. Most of all, we take advantage of the detailed information about accuracy.

The rest of the paper is organized as follows. In the following section we review some previous work on identification of relations between biological named

entities. Then we describe the annotation of the GENIA corpus and how we extracted examples therefrom in Section 3. We explain how we incorporated abstraction in the characterization of examples in Section 4. Section 5 describes the experiments we carried out and the results of evaluation. We finish with some conclusions and future work.

2 Related Work

In recent years, natural language processing in the biomedical domain has experimented more intense interactions with intensive natural language processing techniques like information extraction or text mining, leading to significant progress in the tasks related to named entity recognition and classification (BioNERC). A number of resources have been made publicly available as well.

Having achieved a good performance in BioNER, tasks building upon entities have come into focus. For some time now, researchers are trying to build systems that identify and classify relations between NEs, as a step forward to information extraction from biomedical text. Most of the work in this area is focussed on protein-protein interaction.

The problem of identifying relations between entities is a classical one in information extraction. For restricted domains, the problem can be solved with reasonable degrees of accuracy, that is why it can be expected that it works well for subdomains of the biomedical domain.

Different techniques have been applied to solve the problem of identifying interactions between NEs. Following the mainstream in NLP, machine learning methods are being widely used (e.g., Airola et al. 2008 uses a variety of corpora, Haddow 2008 a variety of features). To apply these methods to the biomedical domain, corpora annotated with interactions between NEs have been created.

In this paper we build a system that applies machine learning techniques learning from the event annotation in the GENIA corpus, described in Section 3.1. Other corpora annotated with relations in the biomedical domain are publicly available, most of them focusing on protein-protein interactions, like BioInfer (Pyysalo et al. 2007) or AIMed (Bunescu et al. 2005). The ITI-TXM corpora (Alex et al. 2008) are two corpora of full-text biomedical research, one of them annotated with PPI and the other with tissue expressions. In contrast, the GENIA corpus presents a variety of relations from the Gene Ontology.

Also following the trend in NLP, not only fully supervised methods are being used, but also weakly supervised or semi-supervised methods. For example, Bunescu and Mooney (2007) take some positive and negative examples of a given relation and extract more examples from the web that contain the NEs and words in negative or positive examples as further positive or negative examples, respectively. These automatically acquired examples are used as training for a classical machine learning approach to learn the relation.

In this paper we present the first step towards a bootstrapping approach similar to that of Bunescu and Mooney (2007), focussing on the best classifier to obtain for a given set of seed examples, in our case, the GENIA corpus.

Table 1. Number of instances in the GENIA corpus for each class of relations in the GENIA ontology. Nodes at the second highest level of the ontology are in boldface, the rest are nodes at the lowest level of the ontology (leaves).

Artificial_process	311	
Biological_process	9	
Cellular_process		
Cell_adhesion	122	
Cell_communication	30	295
Cell_differentiation	119	
Cell_recognition	4	
Cellular_physiological_process	20	
Physiological_process	456	
Binding	1421	
Localization	1	
Metabolism	8	
DNA_metabolism	4	
DNA_recombination	21	1948
Mutagenesis	20	
Gene_expression	1	
Protein_catabolism	1	
Protein_amino_acid_dephosphorylation	3	
Protein_amino_acid_phosphorylation	11	
Protein_processing	1	
Correlation	205	
Viral_life_cycle	35	163
Initiation_of_viral_infection	128	
Regulation	827	
Negative_regulation	493	2913
Positive_regulation	1593	
UNCLASSIFIED	15	

A weakly supervised approach has the advantage that it requires very few manually annotated examples, since it is capable of acquiring a big number of examples fully automatically. Thus, the number of acquired examples can be sufficient for an automatic inference approach to learn the target relations with more reliability than in purely supervised methods, which use only the small number of examples that are available in hand-annotated corpora. For an illustration of the number of examples available for each relation in a corpus as big as GENIA, see Table 1.

3 Examples to Learn from

3.1 GENIA Event Annotation

The GENIA corpus (Kim et al. 2008) contains 1000 abstracts, with 9372 sentences and 30411 annotated events. Events are defined as *"A change of the biological*

state, properties or location of a bio-molecule.", which makes them encompass a wider spectrum of textual phenomena than purely relations between biological entities. That is one of the reasons why we have restricted ourselves to a subset of these events, as we explain in the next section.

Each event is classified in one of the semantic classes described in the GENIA ontology. The distribution of classes in the corpus can be seen in Table 1. Some of the entities occur very infrequently, so the learning process would probably benefit from some kind of generalization, as we did for classes of events. This, however, is left for future work.

3.2 Candidates

We take into consideration only relations between two entities in the same sentence, thus discarding events where more or less than two entities were implied, or events where events were implied as arguments. This leaves us with 5859 instances out of a total of 30411 events annotated in GENIA.

3.3 Characterization of Examples

The basic set of information that we are using to characterize examples is:

- distance between the two entities (in number of words),
- lexical form by which each of the entities is represented in text,
- concept in the GENIA ontology to which each of the entities belongs,
- part of speech and lexical form of words in a window of two words to the left and two to the right of each of the entities involved in the relation.

These features are very varied, and many of them are sparse. To reduce data sparseness and to eliminate noise due to possible errors, a possible solution is to filter out features occurring few times. For example, if all features are taken into account, the number of features is as big as 11393, for 5859 instances. Considering only words that occur more than once in the context of a term leaves us with 8561 features, and considering words occurring more than twice leaves us with 7496, almost the half. On the other hand, considering only the lexical form and semantics of entities occurring more than once leaves us with 8806 features.

We carry out experiments with all words (in the tables describing experiments named *words as features, all words*), 11393 features), with words and entities ocurring more than once (*words as features, >1*, 5174 features) and more than twice (*words as features, >2*, 3208 attributes). We also carry out a set of experiments where we discard the lexical form of words in the context of occurrence of the entities in the relation (*words as features, no lex*, 5507 attributes). The most drastical reduction of features is achieved by using positions relative to terms as features, and words occurring in those positions as values (*words as values*). This approach reduces the number of features to 22, but it does not imply a reduction of the search space, complexity is just it is just represented in a different manner.

In the annotation of the GENIA corpus, much more information is available, but we do not exploit it to characterize examples because this information will not be available in other annotated corpora or even in automatically annotated text, and we want our system to run on any kind of text. For example, we do not exploit the "clue" feature, that characterizes the part of the text that is giving the hint about the relation. We expect that this crucial part of the text will be included within the larger set of features described above.

4 Incorporating Abstraction in the Learning Process

Since the GENIA corpus is very small, the training data to learn the classifier suffers from data sparseness. In order to alleviate this problem, we apply different kinds of generalization. First, we use the GENIA ontology, substituting each relation by a superclass. Second, we cluster relations and substitute each relation with the label of the cluster it belongs to. Finally, we applied automatic feature selection techniques to discard those features that were not discriminating, thus reducing dimensionality and eliminating noise. We describe these approaches in what follows.

The effects of generalization are comparable to those of smoothing: the problem of overfitting is reduced by reserving some probability mass for unseen events. In most common forms of smoothing, probability mass is explicitly reserved by adding a factor to the actual counts of seen events. By generalization, we are reserving probability mass because we assume that unseen events are included as instances of more general classes or in examples that are described by more general features.

4.1 Ontology

Looking for generalization, we substituted each relation by one of its ancestors in the GENIA ontology. We evaluated the impact of two different degrees of abstraction within the ontology. The number of instances of each relation at different levels of abstraction can be seen in Table 1. Without any abstraction, we have 26 classes, substituting each relation by the topmost-but-one level of abstraction in the ontology (level 2) leaves us with 8 classes.

Having less classes, distinctions are coarser-grained but we have more instances for each class, which will hopefully help to obtain a more reliable model.

4.2 Clustering

As an alternative to ontology-based generalization, we clustered relations to find empirically motivated groups. Then, these groups will be taken as classes, so that the task to be learned now will not be distinguishing the originally annotated classes, but these newly built clusters. This method is likely to produce a classifier

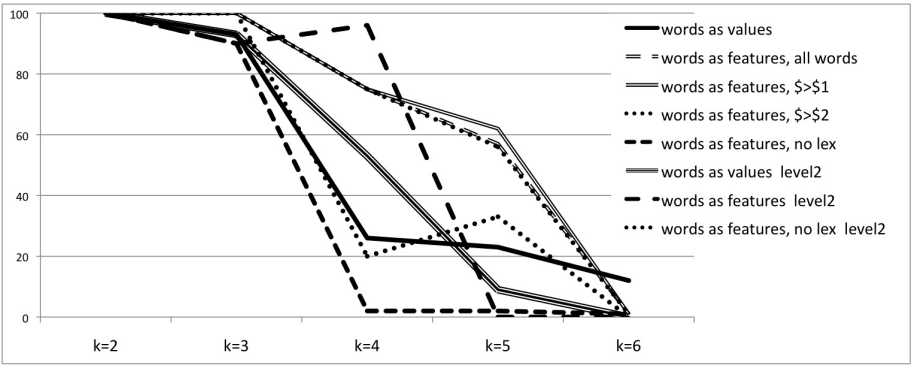


Fig. 1. Results of k-means clustering in our dataset, displaying the ratio of the total population that belongs to a class that has at least half of their population in a single cluster.

with very high accuracy, because one is using the very same features to create clusters and to discriminate their elements. The problem with this method is the interpretability of clusters: are the obtained clusters useful for the goal of information access? In this framework, a good clustering solution should present these properties[2]: capable of conveying fine-grained distinctions (that is, the more the clusters, the better), with high correlation with the original classes.

In order to have a quick, general idea of the goodness of various clustering solutions, we analyzed how compact the original GENIA classes were distributed in clusters. We prefer those solutions where most of the classes and most of the population is distributed among clusters so that, for each class, a single cluster holds more than half of the population belonging to that class.

We applied k-means clustering to our dataset. Figure 1 shows how different clustering solutions perform in concentrating most of the population of each individual class in the same cluster. We can see that the family of solutions that seems to achieve the best results in that aspect is *words as features, level 2*. At the leaf level of generalization, *words as features, >1* seems to perform best.

The question whether clusters in this solution are interpretable and will be useful for information access, this has to be carefully evaluated in an applicative environment. But in an initial overview, when we take a look at the content of each cluster of these solutions, they look quite uninterpretable, with most of the clusters having members of various classes, and classes being quite scattered across clusters.

Therefore, in this work we take clustering solutions as an upper bound of performance for a classifier with this dataset and the features that we have chosen, but we do not consider them as a real alternative for generalization.

[2] Besides the usual metrics for assessing the goodness of clustering solutions.

4.3 Feature Selection

We had a very big feature space, which makes even more acute the problem of data sparseness. As said before, we discarded features that did not occur more than twice in the corpus, which reduced the space to almost half of its original size. Even so, the space was still too big for the few examples we were dealing with, so we applied automatic feature selection techniques. These techniques try to find the subset of features that are most adequate for the classification problem at hand, because simply adding all possible features does not necessarily give the best results (Guyon and Elissee, 2003).

We explored different techniques to find the optimal subset of features, and we found no significant difference between them, therefore we carried out only the family of experiments with the feature selection methods that Weka offers by default: `CfsSubsetEval` as the evaluator, and `BestFirst` as the search strategy. `CfsSubsetEval` (Hall 1998) chooses subsets of features that are highly correlated with the target classification and have low intercorrelation between each other.

The original set of 7496 features is reduced to 26 without any generalization and to 44 with generalization via ontology.

5 Evaluation

We carried out various experiments in the GENIA event annotation corpus, trying to learn three different kinds of classifiers: a symbolic, rule-based classifier (Repeated Incremental Pruning to Produce Error Reduction (RIPPER), here named JRip after Weka's class), a Naive Bayes classifier (NB) and a support vector machine (Sequential Minimal Optimization, SMO), all of them as implemented in Weka. Learning different classifiers was useful evaluate different approaches to the problem and the representation of examples.

All classifiers were evaluated by ten-fold cross-validation on the GENIA event corpus. Results are displayed in Table 2, reporting the proportion of correctly classified instances and the κ coefficient to assess the reliability and reproducibility of results (Carletta 1996). For the purpose of extending the training set, reproducibility is a key issue, since we want to reproduce the classification learnt from the GENIA event corpus in other corpora.

Some experiments were not carried out because of lack of computational power, like those with JRip and *words as features* without any pruning. Some others were not carried out because of lack of interest, as those with SMO taking the results of clustering as classes, because these would surely yield very high values given the good results for the rest of classifiers, but, as discussed above, are uninteresting for us because clusters are uninterpretable.

The classifier performing best was SMO, obtaining κ values above .7, which indicates good reproducibility of results. Therefore, this classifier provides us with the reproducibility that we need to extend the training set with unannotated corpora. In addition, since kernel methods are very robust with respect to noisy data, this seems to be a very adequate approach for bootstrapping in unannotated corpora.

Table 2. Experiments with different generalizations, evaluated by 10-fold cross validation on the GENIA event corpus. Some experiments are missing due to lack of computational power or interest. Best results are highlighted in boldface.

	JRip		Naive Bayes		SMO	
	% correct	κ	% correct	κ	% correct	κ
no generalization						
words as values	58	.47	71	.65	**78**	**.73**
words as features, all words	–	–	55	.47	**78**	**.73**
words as features, >1	59	.49	55	.47	77	.72
words as features, >2	59	.49	55	.47	76	.71
words as features, no lex	54	.42	52	.43	70	.63
generalization via ontology - ancestors level 2						
words as values	78	.64	83	.73	**89**	**.82**
words as features, all words	–	–	69	.54	**89**	**.82**
words as features, >1	80	.67	69	.54	88	.81
words as features, >2	79	.66	69	.53	87	.80
words as features, no lex	76	.60	70	.53	85	.76
generalization via feature selection (words as features, all words)						
leaf level	51	.37	52	.40	54	.42
ancestors level 2	76	.55	72	.55	77	.61
generalization via clustering						
words as features, k-means 2	100	1.00	100	1.00		
words as features, k-means 3	88	.79	89	.82		
words as features, k-means 4	90	.86	89	.85		
words as features, k-means 5	92	.89	91	.88		
words as features, k-means 6	94	.93	93	.91		
generalization via clustering on ancestors level 2						
words as features, k-means 2	99	.99	99	.99		
words as features, k-means 3	99	.99	95	.92		
words as features, k-means 4	99	.98	97	.96		
words as features, k-means 5	98	.97	95	.94		
words as features, k-means 6	95	.93	93	.91		

Evaluation on the training data yielded 100% accuracy for SMO, which indicates that the problem is linearly separable with the chosen features, which in turn indicates that the features are adequate to represent the problem. Since the range of features to represent this kind of problem is very big (from letter-grams to dependency structures or ontology-based semantics), this result is very encouraging.

It is also interesting to note that eliminating infrequent features (the *words as features* > 1 and > 2 approaches) slightly worsens the performance of the classifier, on the contrary. This leads us to the conclusion that the accuracy

of classifiers can be increased by increasing the number of training examples, with no need to add sophisticated pre-processing of the texts. In this way, the bootstrapping approach provides us with exactly what we need.

We can see that there is no difference in performance for the approaches *words as values* and *words as features* for SMO. As said before, both approaches are equivalent in the information they represent, only the form is different. This difference in form produces small differences in performance for the other two classifiers. The Naive Bayes approach performs significantly better with the *words as values* approach. The rule-based classifier performs better with *words as features*, but requires a lot of computational resources to deal with the *all words* approach.

Using the ontology to generalize classes of events the accuracy of classifiers increases by 10 (SMO and Naive Bayes) to 20 percent (rule-based). This is mainly due to the fact that the classification is easier, since there is a smaller number of classes, with more examples to learn for each class, and classes with small population have been eliminated, as can be seen in Table 1. The reproducibility of results is also higher for this approach, reaching above $\kappa = .8$. Therefore, it seems advisable to apply first this level of generalization to acquire new instances by bootstrapping, and apply finer-grained classes only in a second phase.

Using clustering methods to generalize classes of events yields very good results with respect to classification. However, as said before, the classes obtained by clustering do not seem interpretable. This intuition is confirmed by the rules produced by the rule-based classifier, which are in general meaningless, for example:

```
a) (term2+2_lex__ >= 1)
       => cluster=cluster2 (903.0/0.0)
   => cluster=cluster1 (4956.0/0.0)

b) b.1) (term2+1_lex_. >= 1)
            => cluster=cluster2 (897.0/0.0)
   b.2) (term1-1_lex_of >= 1)
            and (term2-1_pos_DT <= 0)
            and (term2_sem_DNA_domain_or_region <= 0)
            => cluster=cluster3 (698.0/27.0)
```

Rule `a)` is the only rule in a k=2 clustering solution, and yields 100% accuracy, but clusters are not separated by any interpretable feature but by the fact that there is no word two positions to the right of the second term involved in the relation, that is, that the second term is close to the end of the sentence. Rules in `b)` is part of the rules in a k=4 clustering solution, and we can see that meaningless rules, like the one applying to the punctuation mark ".", `b.1)` still apply for most of the population and with the highest accuracy, while rules incorporating some interpretable features, like `b.2)` tend to do it negatively, that is, classifying examples that do not have a feature like "semantics of the second term is DNA_domain_or_region".

Finally, we can see that feature selection does not yield an improvement in performance for any of the classifiers or approaches. Probably, a wider variety of feature selection methods need to be tested to find a subset of features that does yield an improvement.

6 Conclusions and Future Work

We have presented the first phase of a procedure to learn a broad-coverage, portable tool to identify and classify relations between biological named entities. The procedure is based on a bootstrapping approach to enrich a seed of training examples with examples from unannotated corpus. In this first phase, we have carried out different experiments to learn a classifier that can reliably identify and classify relations with high reliability in unannotated text, so that they can be incorporated as training examples to learn a classifier with wider coverage and improved precision.

We have used the GENIA event annotated corpus as seed corpus to learn a classifier. Examples have been characterized with shallow features and biomedical Named Entities, which can be automatically identified in the huge amount of unannotated biomedical research papers that are publicly available.

We have found that a support vector machine can reliably classify relations learning from this small training set characterized with shallow features only. Reproducibility of the classification is assessed by values of the κ coefficient above .7 for fine-grained classes of relations and above .8 for coarser-grained classes of relations. Moreover, evaluation on the training data yielded 100% accuracy, which indicates that the problem is linearly separable with the chosen features, which in turn indicates that the features, even if they are shallow, are adequate to represent the problem.

Future work includes developing the second part of the approach, that is, applying this classifier to unannotated corpora and incorporating as training examples instances that can be reliably classified, thus iteratively enhancing the training corpus. We will also explore in depth more methods to generalize examples in order to avoid data sparseness: we will explore ontology-based generalization for entities, and also further experiments on feature selection.

References

[Airola, Pyysalo, Björne, P ahikkala, Ginter, and Salakoski 2008] Airola, A., Pyysalo, S., Björne, J., Pahikkala, T., Ginter, F., Salakoski, T.: A graph kernel for protein-protein interaction extraction. In: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing, Columbus, Ohio, June 2008, pp. 1–9 (2008)

[Alex, Grover, Haddow, Kabadjov, Klein, Matthews, Tobin, and Wang 2008] Alex, B., Grover, C., Haddow, B., Kabadjov, M., Klein, E., Matthews, M., Tobin, R., Wang, X.: The ITI TXM corpora: Tissue expressions and protein-protein interactions. In: Proceedings of the Workshop on Building and Evaluating Resources for Biomedical Text Mining at the 6th International Conference on Language Resources and Evaluation (LREC 2008), Marrakech, Morocco (2008)

[Bunescu and Mooney 2007] Bunescu, R., Mooney, R.: Learning to extract relations from the web using minimal supervision. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Prague, Czech Republic, pp. 576–583 (June 2007)

[Bunescu, Ge, Kate, Marcotte, Mooney, Ramani, and Wong 2005] Bunescu, R.C., Ge, R., Kate, R.J., Marcotte, E.M., Mooney, R.J., Ramani, A.K., Wong, Y.W.: Comparative experiments on learning information extractors for proteins and their interactions. Artif. Intell. Med. 33(2), 139–155 (2005)

[Carletta 1996] Carletta, J.: Assessing agreement on classification tasks: The Kappa statistic. Computational Linguistics 22(2), 249–254 (1996)

[Haddow 2008] Haddow, B.: Using automated feature optimisation to create an adaptable relation extraction system. In: Proceedings of BioNLP, Columbus, Ohio (2008)

[Hall 1998] Hall, M.A.: Correlation-based Feature Subset Selection for Machine Learning. Ph. D. thesis, University of Waikato, Hamilton, New Zealand (1998)

[Kim, Ohta, and Tsujii 2008] Kim, J.D., Ohta, T., Tsujii, J.: Corpus annotation for mining biomedical events from literature. BMC Bioinformatics 9(1) (2008)

[Pyysalo, Ginter, Heimonen, Björne, Boberg, Järvinen, and Salakoski 2007] Pyysalo, S., Ginter, F., Heimonen, J., Björne, J., Boberg, J., Järvinen, J., Salakoski, T.: Bioinfer: A corpus for information extraction in the biomedical domain. BMC Bioinformatics 8, 50 (2007)

[Witten and Frank 2005] Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)

[Yarowsky 1995] Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: ACL 1995, Cambridge, MA, pp. 189–196. ACL (1995)

Using a Bigram Event Model to Predict Causal Potential

Brandon Beamer and Roxana Girju

University of Illinois at Urbana-Champaign
{bbeamer,girju}@illinois.edu

Abstract. This paper addresses the problem of causal knowledge discovery. Using online screenplays, we generate a corpus of temporally ordered events. We then introduce a measure we call *causal potential* which is easily calculated with statistics gathered over the corpus and show that this measure is highly correlated with an event pair's tendency of encoding a causal relation. We suggest that causal potential can be used in systems whose task is to determine the existence of causality between temporally adjacent events, when critical context is either missing or unreliable. Moreover, we argue that our model should therefore be used as a baseline for standard supervised models which take into account contextual information.

1 Introduction

Automatic recognition and extraction of causal event sequences from text is a crucial task for many Computational Linguistics applications. It is a prerequisite in text coherence, entailment, question answering, and information retrieval (Goldman et al., 1999; Khoo et al., 2001; Girju, 2003). Put simply, it is a prerequisite to perform textual *reasoning*.

Whether two textual units (words, phrases or sentences) are in a causal relationship is largely dependent on context. But that is not to say that such pairs in general have no statistical tendencies when it comes to causality.

This paper describes a knowledge-poor unsupervised model relying on a statistical measure we call *causal potential*. Our focus is on event sequences as expressed by consecutive verbs in a discourse (event pairs). Event pairs with a high causal potential can be interpreted as being more likely to occur in causal contexts than events with low causal potential. This measure can then be used in more complex systems to gain causal intuitions in situations when context is scarce or unreliable. Therefore, we argue that our model should be used as a baseline by standard supervised models which take into account contextual information.

In this paper we evaluate our measure of causal potential and show that it correlates highly with human observances of causal events in text.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 430–441, 2009.

2 Previous Work

Despite its importance to Computational Linguistics, the task of causal knowledge extraction has not been tackled much in this field. Many of the early works on this topic (Khoo et al., 2001; Girju, 2003) focused on predefined lexico-syntactic constructions encoding causal relations. Girju (2003), for example presents a supervised knowledge-intensive system which relies on "noun – verb – noun" constructions to identify new noun–noun pairs encoding cause-effect (e.g., *Tsunamis cause tidal waves*). The verb is identified from a set of 60 causal WordNet verbs such as *cause, lead to, provoke*. Her system obtains a precision score of 73.91% and she proves the importance of the system in the task of question answering.

Chang and Choi (2006) improve over Girju's approach by employing a dependency parser which identifies patterns of the type "NP_cause cue NP_effect". They use a Naive Bayes classifier based on cue phrase and lexical pair (NP_cause–NP_effect) probabilities which are learned from an unannotated corpus of examples. Chang and Choi (2006) start with the classifier proposed by Girju (2003) and then use expectation-maximization to bootstrap the final classifier. The expectation and maximization steps are repeated until no improvement is obtained. The reported performance is 80% F-measure.

In this paper we present a knowledge-poor unsupervised approach to the identification of causal relations between temporally adjacent events denoted by verbs. Unlike previous attempts, our system does not rely on any predefined lexico-syntactic pattern. Instead, events are identified only after part of speech tagging the text. In particular we introduce a measure called *causal potential* which relies on statistics gathered over a large unannotated corpus and show that this measure is highly correlated with human judgments.

3 Our Notion of Causality

As mentioned earlier, determining if two events are in a causal relationship is no simple matter. The task of constructing a definition of causality that is both rational and fits our linguistic intuitions still eludes many fields including Linguistics and Philosophy.

The challenge of defining causality has been pursued by philosophers for a long time. While they have not yet resolved the issue, numerous schools of thought have emerged from their efforts (Sosa and Tooley, 1993). Most relevant to our goal of language processing and understanding is a consistent set of annotation guidelines which capture our perception of causality as expressed by language. Thus, we are interested in causal theories which provide the annotator with a relatively objective test, allowing her to judge the causal relation between two events without relying on intuitions which will vary significantly from annotator to annotator. Additionally, the test should also be easy to perform mentally, without needing detailed philosophical knowledge about causality. After reviewing various causality theories in philosophical literature, two of them lend themselves as possibilities: *counterfactual* theories and *manipulation* theories.

3.1 Counterfactual Tests of Causality

Counterfactual theories (see Menzies (2008) for an overview) examine causality via counterfactual statements. For example, the statement *Mary shooting John caused his death* has the counterfactual equivalent: *John would not have died (at that moment) had Mary not shot him.* Shibatani (1976) offers a rigorous counterfactual definition of causation:

Two events qualify as a causative situation if

(a) the relation between the two events is such that the speaker believes that the occurrence of one event, the 'caused event', has been realized at t_2, which is after t_1, the time of the 'causing event'; and if
(b) the relation between the causing and the caused event is such that the speaker believes that the occurrence of the caused event is wholly dependent on the occurrence of the causing event; the dependency of the two events here must be to the extent that it allows the speaker to entertain a counterfactual inference that the caused event would not have taken place at the particular time if the causing event had not taken place, provided that all else had remained the same.

Hence a counterfactual test for an annotator deciding if event A causes event B could be to ask herself the following questions/criteria:

(i) Did event A occur before (or simultaneously with) event B?
(ii) Is the occurrence of event B wholly dependent on the occurrence of event A?
(iii) Had event A not taken place, could one necessarily infer that event B would not have taken place?

If and only if a given situation satisfies all these constraints, the annotator would decide that the two events in question are causally related. A test like this satisfies our constraint that annotation tests should be easy, and should not require in-depth knowledge of works in philosophy and logic. However, it has a few problems. First, deciding if one event is "wholly dependent" on another is rather vague and possibly subjective. Second, assuming the annotator has a vague understanding of criterion (ii), criterion (iii) can produce false positives in cases where two events have a common cause. Consider the following example:

(1) *Mary shot John. John collapsed. John died.*

In this context, the statement "John's collapsing caused his death" is false, even though the counterfactual "Had John not collapsed, could one necessarily infer that John would not have died" is true. John collapsed because he was shot, hence had John not collapsed that would necessarily entail that he had not been shot, which would also entail that John would not have died. Having a good grasp on criterion (ii) can help an annotator avoid this pitfall, but *dependency* as it is used in the definition is not very well defined and this could lead annotators astray.

3.2 Manipulation Tests of Causality

Manipulation theories (see Woodward (2008) for an overview) examine causality via mental experiments where one manipulates one event and observes the behavior of another. A few examples of manipulation definitions of causality are:

(1) A causes B if control of A renders B controllable. A causal relation, then, is one that is invariant to interventions in A in the sense that if someone or something can alter the value of A the change in B follows in a predictable fashion. (Hoover, 1988)
(2) Z_1 is a cause of Z_2 is just a convenient way of saying that if you pick an action that controls Z_1, you will also have an action that controls Z_2. (Orcutt, 1952)

Most philosophers agree today that, insofar as the definition of causality is concerned, manipulation approaches are insufficient because it turns out to be quite impossible to discuss any notion of control in non-causal terms. Thus the definition becomes circular. This is not a problem for us though, as we are not seeking a rigorous philosophical definition of causality, but rather a relatively objective test for the purpose of linguistic annotation. In this capacity, manipulation theories turn out to be quite useful.

Thus, we can devise a new annotation test for causality. For example, an annotator deciding whether event A causes event B could ask herself the following questions instead. Answering *yes* to both would mean the two events are causally related:

(i) Does event A occur before (or simultaneously) with event B?
(ii) Keeping constant as many other states of affairs of the world in the given text context as possible, does modifying event A entail predictably modifying event B?

This annotation test is both simple to execute mentally and is relatively objective. Subjectivity would certainly arise in cases where the annotator is unaware of how certain things in the world work, but relying on the fact that many people share more or less the same baggage of commonsense knowledge this should not be a problem.

We word our annotation test in terms of the manipulation mindset because we believe the language is easier to understand. However, it is important to see that in practice both the counterfactual and the manipulation tests end up being mostly equivalent. The manipulation test instructs the annotator to modify event A and observe the behavior of B. The simplest and most extreme way to modify event A is to either add it to or remove it from the world. The outcome of adding the event is obvious since it corresponds to the situation described in the text. The only mental experiment the annotator ends up doing considers the outcome when the event is removed (i.e. the event does not happen). Hence, if the text describes event A preceeding event B, then since it is obvious that B happens when A happens given the context, the question is always "does B still happen when A does not?". This is a counterfactual test.

4 The Corpus

In this section we present the text collection employed for this research along with details about the annotation guidelines.

4.1 Corpus Construction and Processing

Determining a causal relationship between two events A and B necessarily entails first establishing that A temporally precedes B, as it is impossible for a future event to cause one in the present or past. Since poor temporal judgments will most certainly hinder the performance of any causal prediction model, it is crucial to first establish an accurate temporal ordering of any event sequence on which causal predictions are to be made. And since the state of the art in automatic temporal ordering of narrative events has not yet achieved an adequate level of accuracy (Mani et al., 2006; Verhagen et al., 2007; Chambers and Jurafsky, 2008; Bethard and Martin, 2008), one of our goals was to find a large source of online text describing events in an already temporally ordered fashion. We achieved this goal by utilizing online screenplays.

Table 1. Portion of a prototypical screenplay

INT. NORTH KOREAN CONSULATE - HALLWAY - DAY

Joy opens the bathroom door. Sam is standing there, grinning.

JOY
There are six bathrooms in this house, Sam.

SAM
(fanning the air)
But only one with a smoking section.

She quickly closes the door behind her. Sam laughs.

Figure 1 shows a small portion of a prototypical screenplay. Screenplays can be broken up into two major components: action and dialog. As shown in the figure, action and dialog are explicitely separated via the format of the document; usually lines that contain different kinds of information (e.g. actions, dialog, scene breaks) have their own indentation and/or capitalization.

Screenplays are very useful for our purposes for two main reasons: (1) scene breaks are clearly marked, and thus it is easy to detect breaks in event sequences and (2), actions are consistently written in present tense. Thus, identifying the temporal order of textual events becomes trivial since the temporal order and event story order are usually equivalent.

Our corpus was generated from 173 screenplays downloaded from the internet[1]. Camera instructions, character dialog, and other non-action text was removed and scene breaks were used to separate the remaining action text. The

[1] The screenplays were extracted primarily from joblo.com.

resulting corpus consists of 2,554,364 word tokens describing textual actions with very confident temporal ordering. We then part-of-speech (POS) tagged (Roth and Zelenko, 1998) the words in the text collection thus acquired to identify and extract events (verbs).

4.2 Corpus Annotation

According to the definitions provided in Section 3 we generated a set of annotation guidelines. These guidelines state to annotate an example as causal if and only if the following conditions hold:

(i) Each event must be represented in text as one or more consecutive verbs.
(ii) Each event must refer to separate distinct events in context.
(iii) The events must occur either at the same time or in the order in which they were written.
(iv) When as much about the world as possible is held constant, manipulating the existence or manner of the first event must *predictably* and *unavoidably* manipulate the existence or manner of the second event.

We used two annotators who received the guidelines prior to annotation. Then, for each event pair they were provided with consecutive sentences containing the pair throughout the corpus and were asked to annotate each example as *Yes, No*, or *ERR*. *Yes/No* capture causal/non-causal relations. *ERR* captures instances which are not valid. By *valid instances*, we mean bigram instances which actually describe two unique events occurring in order. Possible reasons of invalid instances include: (a) the events do not occur in temporal sequence, (b) the two verbs do not describe distinct events, (c) there is a verbal event (that the POS tagger missed) written between the two marked events (i.e. they are not orthographically adjacent), (d) one or both of the marked events are mistagged as verbs by the POS tagger.

Examples of invalid instances are shown in situations (2) and (3) below.

(2) *John* **looked** *at Mary with sore eyes. They* **broke** *up five years ago but John always wanted a second chance.*

While such a case would be *extremely* rare in the corpus, the events are in past tense breaking the assumed temporal ordering of events. Hence the bigram instance *look→break* would not be considered when calculating the observed causal frequency of the bigram in general.

(3) *The bomb exploded,* **sending** *John* **flying**.

This is a much more common case in the corpus. This is an error however because the two highlighted verbs do not represent two distinct events – it is not the case that a *flying* event occurred just after a *sending* event. Hence this bigram instance of *send→fly* would not be considered during evaluation.

The following examples show event bigram instances from the corpus coupled with our annotation judgments and explanations.

(4) *It* **explodes** *with incredible force,* **sending** *dead bodies in all directions.* – **CAUSAL**

`Explanation:` The explosion occurs before the sending event. And if one could control whether the explosion happens, could also predictably control whether the sending of dead bodies happens.

(5) *Rudy* **picks** *up the phone and* **dials** *a number.* – **CAUSAL**
`Explanation:` The critical state of affairs to be aware of here is Rudy's implied intent to dial the number. Keeping that and everything else possible constant, one could control whether the dialing event happens by controlling whether the picking-up event happens.

(6) *As he* **says** *this, he* **holds up** *his right arm.* – **NOT CAUSAL**
`Explanation:` The saying and the holding up events happen simultaneously (which is fine), but independent of each other.

5 Model Description

We model the ordering of consecutive events by calculating bigram frequencies. From our corpus, we extracted 328,035 event instances defined over 4,368 unique events and 120,004 unique bigrams (pairs of adjacent verbs - in the same sentence or consecutive sentences). Using only statistics calculated over bigram and event frequencies, we then calculate a measure we call *Causal Potential*.

The causal potential of any two events is a measure which gauges how likely these two events are to be in a causal relationship without prior knowledge of any context. Causal potential ($\mathcal{C}$) is calculated via the following formula:

$$\mathcal{C}(e_1, e_2) = \log\left(\frac{P(e_2|e_1)}{P(e_2)}\right) + \log\left(\frac{P(e_1 \rightarrow e_2)}{P(e_2 \rightarrow e_1)}\right) \quad (1)$$

Our arrow notation ($\rightarrow$) denotes bigrams. Hence $e_i \rightarrow e_j$ denotes the event bigram (e_i, e_j) and $P(e_i \rightarrow e_j)$ is simply the frequency of occurrence of $e_i \rightarrow e_j$ in the corpus. To help avoid 0-probabilities, we adopt a very simple smoothing policy whereby non-existent bigram counts are assigned a frequency of 1.

There are two main intuitions behind our causal potential $\mathcal{C}$. The first term comes from the notion of probabilistic causation which defines it in terms of the causal event's occurrence increasing the probability of the result event (Suppies, 1970; Mellor, 1995). Thus the first term has high values when $P(e_2|e_1) > P(e_2)$ and has low values when $P(e_2|e_1) < P(e_2)$. The second term comes from the basic assumption that causes must preceed effects and thus if two events occur often in causal situations, they should also occur often in temporal order. Hence the second term has high values when they occur more often in order, and has lower values the more often they occur out of order. Satisfying both of these intuitions results in high values of $\mathcal{C}$ while lacking in one or both of them lowers the value of $\mathcal{C}$.

Our measure of causal potential has a range of $(-\infty, \infty)$. However since we smooth our frequency counts these types of results will never occur in practice (as shown in Figure 1).

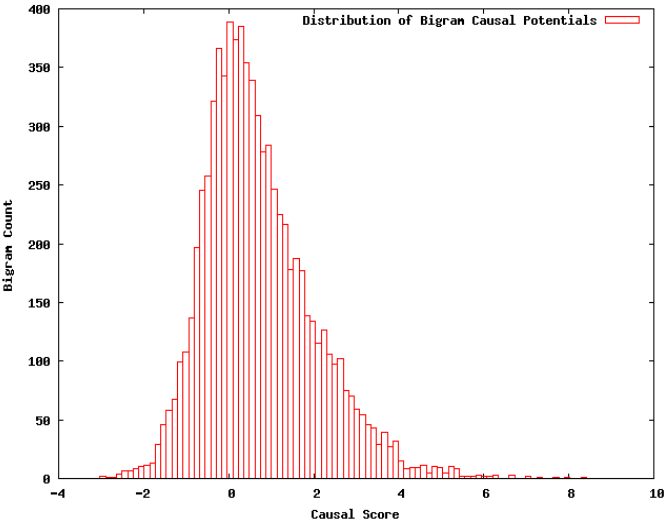


Fig. 1. Distribution of causal scores in screenplay corpus when $\mathcal{C}$ is calculated for each bigram in our corpus with frequency at least 5

6 Model Evaluation

Every consecutive event–event pair in the screenplay corpus was ranked by the system based on the causal potential model described above. Since our screenplay corpus is too large, we evaluated the causal potential on a sample of the corpus which was selected in the following way:

(a) All bigrams which had at least 5 instances were sorted by their scores ($\mathcal{C}$)
(b) We selected 90 bigrams from this sorted list: the top 30, the bottom 30, and 30 near the middle.
(c) For each of these bigrams we located all of its valid instances in the corpus.
(d) Each bigram instance was annotated in context according to the annotation test described in Section 3.2.
(e) From these annotations, Table 2 records the number of causal instances (column 2), the number of non-causal instances (column 3), and the number of invalid instances (column 4).
(f) *Observed causal frequency* (column 5) is simply column 2 divided by columns 2 and 3 added together.

Table 3 shows positive and negative examples of sentences corresponding to event pairs in Table 2. The average inter-annotator agreement was 85%.

We restricted our evaluation to only bigrams with at least 5 instances to ensure that observed causal frequencies were calculated over a decent number of instances (more than 5 instances is a rare situation among bigrams).

Observed causal frequency is an estimate of the probability that two events will be in a causal relationship without prior knowledge of context. Table 2

Table 2. Bigrams' Causal Potentials vs. Observed Causal Frequencies. From left to right the columns are: *event bigram*, *number of times bigram instance was observed to represent a causal relationship*, *number of times bigram instance was observed to not represent a causal relationship*, *number of times bigram instances were extracted in error*, *calculated observed causal frequency (the second column divided by the sum of the second and third columns)*, *calculated causal potential* ($\mathcal{C}$).

Bigram	# Causal	# Non-causal	Error	Causal Freq.	$\mathcal{C}$
send → reel	0	0	11	N/A	8.358
offer → refuse	7	0	0	1.00	8.012
send → fly	0	0	39	N/A	7.799
send → sprawl	0	0	11	N/A	7.389
swerve → avoid	2	0	4	1.00	7.137
wear → tailor	0	0	6	N/A	6.725
round → bend	0	0	8	N/A	6.661
give → peck	0	0	6	N/A	6.368
send → crash	2	0	7	1.00	6.238
leave → strand	0	0	7	N/A	6.201
put → gear	0	0	10	N/A	6.075
explode → send	11	0	0	1.00	5.992
lean → kiss	40	1	0	0.98	5.965
pick → dial	32	1	0	0.97	5.946
try → lock	0	0	26	N/A	5.857
sound → echo	2	0	4	1.00	5.791
stumble → fall	19	0	1	1.00	5.721
open → reveal	88	0	2	1.00	5.636
swing → connect	7	0	0	1.00	5.564
unlock → open	14	1	3	0.93	5.468
cry → sob	6	0	0	1.00	5.437
sit → nurse	0	7	0	0.00	5.427
seize → drag	8	0	0	1.00	5.399
nod → satisfy	3	0	4	1.00	5.397
hit → send	13	1	1	0.93	5.393
hear → creak	0	0	7	N/A	5.368
scan → spot	6	0	0	1.00	5.364
kick → send	11	0	0	1.00	5.357
raise → aim	8	0	1	1.00	5.323
aim → fire	0	4	3	0.00	5.308
play → go	1	10	2	0.09	0.011
see → shove	1	6	0	0.14	0.010
face → see	10	5	34	0.67	0.010
sit → hear	0	29	0	0.00	0.010
dress → get	1	3	6	0.25	0.010
pull → look	7	15	2	0.32	0.009
roll → fall	2	2	2	0.50	0.009
get → appear	2	6	0	0.33	0.007
slap → look	2	5	3	0.29	0.007
roll → look	2	22	3	0.08	0.007
put → face	0	3	5	0.00	0.006
face → run	0	1	7	0.00	0.002
look → loom	0	8	0	0.00	-0.000
smile → move	0	8	0	0.00	-0.000
play → hear	0	7	2	0.00	-0.001
point → stand	0	8	4	0.00	-0.001
see → fire	2	5	5	0.29	-0.001
punch → look	1	4	2	0.20	-0.001
sit → remain	0	12	2	0.00	-0.003
fall → kick	1	4	1	0.20	-0.003
change → see	3	1	0	0.75	-0.003
go → jump	1	6	1	0.14	-0.005
hang → enter	0	9	1	0.00	-0.005
lead → walk	9	7	1	0.56	-0.005
pull → face	3	4	7	0.43	-0.007
say → hold	2	6	3	0.33	-0.008
pull → emerge	4	4	1	0.50	-0.008
turn → lie	3	13	2	0.19	-0.008
throw → look	10	18	11	0.36	-0.008
look → slap	3	5	2	0.38	-0.010
give → stand	0	7	4	0.00	-1.994
jump → look	1	4	2	0.20	-2.012
sit → fall	0	6	2	0.00	-2.014
shake → sit	0	7	0	0.00	-2.120
look → stop	1	12	3	0.08	-2.026
pick → sit	2	5	3	0.29	-2.036
stare → see	3	6	1	0.33	-2.040
take → enter	0	10	2	0.00	-2.076
look → lock	3	1	2	0.75	-2.102
listen → sit	0	6	0	0.00	-2.106
find → turn	3	4	4	0.43	-2.115
lead → come	2	4	0	0.33	-2.149
take → appear	1	4	1	0.20	-2.155
come → wait	7	2	1	0.78	-2.193
see → open	4	15	12	0.21	-2.217
move → nod	3	3	2	0.50	-2.243
reveal → pull	1	6	3	0.14	-2.253
open → wait	0	5	1	0.00	-2.319
lean → sit	0	7	0	0.00	-2.332
stand → reveal	1	7	2	0.13	-2.334
wear → see	3	4	4	0.43	-2.341
pass → walk	0	7	1	0.00	-2.367
know → look	4	4	11	0.50	-2.447
look → play	0	8	2	0.00	-2.517
turn → read	1	4	1	0.20	-2.556
open → watch	0	5	1	0.00	-2.570
wear → come	0	5	2	0.00	-2.691
stare → stand	0	6	1	0.00	-2.720
reach → walk	3	3	1	0.50	-2.902
enter → open	6	4	2	0.60	-2.948

Table 3. Positive and negative examples of event pairs corresponding to those in Table 2

Examples	Annotation
offer→refuse	
A member of the crew enters carrying a tray on which there is a half-filled glass of liquor, which Sandro takes and <offers> to Anna. Anna positively <refuses> it, and the sailor leaves as Sandro sets the glass down on a shelf.	Yes
explode→send	
Sykes is just past the cars when they <explode> – <sending> hoods and door pane and glass flying in all directions.	Yes
hit→send	
Bits of metal fall, <hit> the fan and <are sent> clanging off into space	No
hit→send	
Brody recoils in horror as the beast rushes past, he spins the wheel and <hits> the throttle, <sending> the launch hard to port.	Yes
pull→look	
Carmen <pulls> Johnny to a stop, <looks> him in the eye.	Yes
pull→look	
A bus <pulls> up to the bus stop. The black woman <looks> down at her watch.	No
lead→walk	
Hilary <leads> the line of vets toward the large anti-Vietnam war rally. The group of vets <walk> as Forrest tries to take another picture.	Yes
lead→walk	
The soldier <leads> the German Prisoner away. Maximus and Marcus <continue walking> in silence for a beat.	No
jump→look	
She <jumps> through the narrow opening as Han and Chewbacca <look> on in amazement.	Yes
jump→look	
Startled, Sid and Beth <jump> back. They <look> at each other and laugh.	No
see→open	
She <can't> see anything. She <throws open> the closet door.	Yes
see→open	
As Epps holds there she <sees> the vent <opening>.	No

shows that bigrams with high causal potential very often have high observed causal frequencies, while bigrams with low causal potential scores very often have low observed causal frequencies.

We used Spearman's rank correlation coefficient to measure the degree of correlation between the observed causal frequency and the calculated causal potential. Spearman's rank correlation coefficient is defined as:

$$\rho = \frac{n(\sum x_i y_i) - (\sum x_i)(\sum y_i)}{\sqrt{n(\sum x_i^2) - (\sum x_i)^2}\sqrt{n(\sum y_i^2) - (\sum y_i)^2}} \tag{2}$$

where x_i and y_i are rankings of two lists. In our case x_i is the ranking of causal frequencies and y_i is the ranking of causal potentials. Spearman's rank correlation coefficient has a range of $[-1, 1]$. A coefficient of -1 corresponds to the two lists being perfectly uncorrelated (one is the reverse sort of the other), a coefficient of 1 corresponds to perfect correlation (the rankings of both lists are identical), and a coefficient of 0 for rankings being completely independent. The Spearman rank correlation coefficient between observed causal frequency and our measure of causal potential is $\rho = 0.497$.

7 Discussion

Our results show that our notion of causal potential is highly correlated with observed causal frequency; Spearman's rank correlation coefficient verifies that the observed ranking and the ranking predicted by our measure of causal potential are positively correlated. Bigrams which score very high have very high observed causal frequencies and those which score very low have very low observed causal frequencies.

Due to aforementioned issues of validity, some of the bigrams are not very reliable. These mostly correspond to bigrams in Table 2 which have a very low number of occurrences or a very high relative number of errors. This low occurrence rate/high error rate is the result of the system's simple approach to event extraction; in some cases, bigrams were not counted as they did not satisfy criterion (i) of our causal annotation test: the two events must occur in temporal sequence. While the nature of the corpus ensures event precedence most of the time, simple identification of part-of-speech tags is not enough to ensure event sequence. For example, some events occurring at the top of the list of causal potentials were: *send* → *sprawl* (c.f. *sends him sprawling*), and *realize/understand/believe* → *happen* (c.f. *realize/understand/believe what's happening*). The problem here is that simple part-of-speech patterns cannot capture the syntactic structure of the phrases and thus falsely extract numerous cases of subordinate clauses. The solution is to recognize structure in the corpus and extract accordingly. Such a solution is tractable and will be implemented in future work.

Another problem with our event extraction technique lies in the shortcomings of part-of-speech taggers. For example, here are a few bigrams extracted which also have a high causal potential: *wear* → *tailor* (c.f. *wears a tailored jacket*), and *leave* → *strand* (c.f. *leaves him stranded*). These are cases where a verb past participle is acting as a noun or adverb modifier. Similar problems can arise with verb gerund forms. In the future we will improve our tag patterns to account for the different verb forms instead of treating them all alike.

8 Conclusion

In this paper we described a knowledge-poor unsupervised causal event model which relies on a statistical measure we call *causal potential*. Causal potential can be easily calculated with simple statistics gathered from a corpus of temporally ordered event pairs. We have empirically shown that event pairs with a high causal potential are more likely to occur in causal contexts than events with a low causal potential and that events with a low causal potential are likely to not occur in causal contexts. This behavior lends our measure of causal potential to be used in cases where context is either absent or unreliable, to gain intuitions regarding the likelihood of two events to be causally related. Moreover, we argue that our model should therefore be used as a baseline for standard supervised models which take into account contextual information.

References

Bethard, S., Martin, J.: Learning semantic links from a corpus of parallel temporal and causal relations. In: The Human Language Technology (HLT) Conference, Short Papers, pp. 177–180 (2008)

Chambers, N., Jurafsky, D.: Unsupervised learning of narrative event chains. In: Human Language Technology Conference, pp. 789–797 (2008)

Chang, D.-S., Choi, K.-S.: Incremental cue phrase learning and bootstrapping method for causality extraction using cue phrase and word pair probabilities. Information Processing and Management 42(3), 662–678 (2006)

Girju, R.: Automatic detection of causal relations for question answering. In: The 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003), Workshop on Multilingual Summarization and Question Answering - Machine Learning and Beyond (2003)

Goldman, S., Graesser, A., Broek, P.: Narrative comprehension, causality, and coherence: Essays in Honor of Tom Trabasso. Erlbaum Associates, Mahwah (1999)

Hoover, K.: The New Classical Macroeconomics. Basil Blackwell, Oxford (1988)

Khoo, C., Myaeng, S., Oddy, R.: Using cause-effect relations in text to improve information retrieval precision. Information Processing and Management 37, 119–145 (2001)

Mani, I., Verhagen, M., Wellner, B., Lee, C.M., Pustejovsky, J.: Machine learning of temporal relations. In: The Association for Computer Linguistics (ACL) Conference (2006)

Mellor, D.H.: The Facts of Causation. Routledge (1995)

Menzies, P.: Counterfactual theories of causation. The Online Stanford Encyclopedia of Philosophy (2008)

Orcutt, G.: Actions, consequences, and causal relations. Review of Economies and Statistics 34, 305–313 (1952)

Roth, D., Zelenko, D.: Part of speech tagging using a network of linear separators. In: The Association for Computational Linguistics Conference (1998)

Shibatani, M.: The grammar of causative constructions: A conspectus. In: Shibatani, M. (ed.) Syntax and Semantics. The Grammar of Causative Constructions, vol. 6. Academic Press, London (1976)

Sosa, E., Tooley, M. (eds.): Causation (Oxford Readings in Philosophy). Oxford University Press, Oxford (1993)

Suppies, P.: A Probabilistic Theory of Causality. North-Holland Publishing Company, Amsterdam (1970)

Verhagen, M., Gaizauskas, R., Schilder, F., Hepple, M., Katz, G., Pustejovsky, J.: Semeval-2007 Task 15: Tempeval temporal relation identification. In: the Fourth International Workshop on Semantic Evaluations (SemEval 2007), pp. 75–80. Association for Computational Linguistics (2007)

Woodward, J.: Causation and manipulability. In: The Online Stanford Encyclopedia of Philosophy (2008)

Semantic-Based Temporal Text-Rule Mining

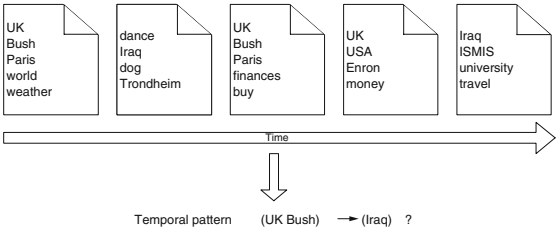
Kjetil Nørvåg* and Ole Kristian Fivelstad

Dept. of Computer Science, Norwegian University of Science and Technology
Trondheim, Norway
Kjetil.Norvag@idi.ntnu.no

Abstract. In many contexts today, documents are available in a number of versions. In addition to *explicit knowledge* that can be queried/searched in documents, these documents also contain *implicit knowledge* that can be found by text mining. In this paper we will study association rule mining of temporal document collections, and extend previous work within the area by 1) performing mining based on *semantics* as well as 2) studying the impact of appropriate techniques for ranking of rules.

1 Introduction

In many contexts today, documents are available in a number of versions. Examples include web newspapers and health records, where a number of timestamped document versions are available. In addition to *explicit knowledge* that can be queried/searched in documents, these documents also contain *implicit knowledge.* One category is inter-document knowledge that can be found by conventional text-mining techniques. However, with many versions available there is also the possibility of finding *inter-version knowledge*. An example of an application is given in the figure below, where a number of document versions are available, and where the aim is to find and/or verify temporal patterns:



In the example above, one possible temporal rule is the terms[1] *UK* and *Bush* appearing in one version means a high probability of *Iraq* to appear in one of the following versions.

How to mine association rules in temporal document collection has been previously described in [16]. In the previous work, the rule mining was performed on *words* extracted from the documents, and ranking of rules (in order to find the most interesting

* Corresponding author.

[1] A term can be a single word as well as multiword phrase.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 442–455, 2009.

ones) was based on traditional measures like support and confidence. However, based on the results it was evident that using simple words did not give satisfactory results, and that more appropriate measures were needed for rule ranking.

In this paper we extend the previous work by performing the temporal mining based on *semantics* as well as studying the impact of other techniques for ranking of rules. Thus the *main contributions* of this paper are 1) presenting the appropriate pre-processing for use of semantics in temporal rule mining, 2) studying the impact of additional techniques for ranking of rules, and 3) presenting some preliminary results from mining a web newspaper.

The organization of the rest of this paper is as follows. In Section 2 we give an overview of related work. In Section 3 we outline the assumed data model, rule mining process, and provide an introduction to our Temporal Text Mining (TTM) Testbench tool. In Section 4 we describe how to perform semantic-based pre-processing. In Section 5 we describe techniques that can increase quality of rule selection by considering semantic similarity. In Section 6 we describe experiments and results. Finally, in Section 7, we conclude the paper and outline issues for further work.

2 Related Work

Introduction to *data mining in general* can be found in many good text books, for example [4]. The largest amount of work in *text mining* have been in the areas of categorization, classification and clustering of documents, we refer to [3] for an overview of these area. Algorithms for mining association rules between words in text databases (if particular terms occur in a document, there is a high probability that certain other terms will occur in the same document) was presented by Holt and Chung in [6]. In their work, each document is viewed like a transaction, and each word being an item in the transaction. In [5] a more thorough overview of previous research in rule mining of text collections is given, with particular emphasis on the case when additional background information is available.

Much research has been performed on aspects related to temporal data mining, and a very good survey of temporal knowledge discovery paradigms and methods is given by Roddick and Spiliopoulou [17]. As will be described in more detail in the rest of the paper, of particular interest in the context of our work is research in intertransaction association rules. The first algorithms for finding intertransaction rules described in the literature, E-Apriori and EH-Apriori [13], are based on the Apriori algorithm. These are extensions of the Apriori algorithm, where EH-Apriori also includes hashing. A further development of intertransaction rules is the FITI algorithm [20], which is specifically designed for efficient mining intertransaction rules.

A general problem in mining association rules is the selection of interesting association rules within the overall, and possibly huge set of extracted rules. Some work in this are exist, either based on statistical methods [18] or by considering the selection of association rules as a classification task [8].

Related to our work is trend analysis in text databases, were the aim is to discover increasing/decreasing popularity of a set of terms [11,15]. A variant of temporal association rule mining is taking into account the exhibition periods of items [10].

3 Preliminaries

In this section we outline the underlying data model for our work, the rule mining process, and a description of the TTM Testbench tool.

3.1 Data Model

We will now outline the data model for temporal documents we use as context for our research. Note that *document* D_i is here used as a generic term, specific types of documents include web pages as well as document formats like MS Word and Adobe PDF. For these document types pre-processing will be employed in order to filter out the actual text from formatting information etc.

The document collection C_i on which we perform the rule mining are assumed to be (or can be converted to) an ordered list of documents $C = [D_1...D_n]$. A document in this context can be the one and only version of a document, or it can be a particular version of a document. Each document is timestamped with the time of creation, and is essentially a tuple containing a timestamp T and an ordered list of words, i.e., $D = (T, [w_1, ..., w_k])$. A word w_i is an element in the vocabulary set V, i.e., $w_i \in V$. There can be more than one occurrence of a particular word in a document version, i.e., it is possible that $w_i = w_j$.

3.2 Rule Mining Process

Mining association rules from a text collections can be described as a 3-step process consisting of 1) pre-processing, 2) the actual mining and 3) post-processing. In the pre-processing phase the documents are converted from external documents into some common representation, words are extracted (tokenization), and then various operations might be performed on the text aiming at increasing the quality of the results or reducing the running time of the mining process. Then the actual mining is performed, resulting in a number of association rules. The number of rules can be very high, and in the post-processing phase the system tries to determine which rules are most interesting, based on some measure. We will now describe the steps as performed in the previous word-centric approach. In Section 4 we will describe how to improve by using semantics.

Pre-Processing. In word-centric pre-processing the text of the documents is filtered and refined. In general the processing time increases with both number of words and size of vocabulary, so the aim of the pre-processing is to reduce both without significantly reducing the quality of the results.

The goal of text filtering is to remove words that can be assumed to not contribute to the generation of meaningful rules. One simple technique is stop-word removal, in which words occurring in a separate user-maintained stop-word list are removed from the text. In addition, words that are very frequently occurring can be removed.

In order to reduce the vocabulary size as well as increasing quality of the contributing terms, stemming can be performed. By employing stemming, a number of related words will be transformed into a common form (similar to the stem of the words).

Finally, term selection can be performed in order to reduce the number of terms. In this process, a subset of the k terms most important words in each document are selected. One such technique we have employed is using the k highest ranked terms based on the TF-IDF (term-frequency/inverse document frequency) weight of each. It should be noted that there is a danger of filtering out terms that could contribute to interesting rules when only a subset of the terms are used, so the value of k will be a tradeoff between quality and processing speed.

Rule Mining. Techniques for temporal rule mining can be classified into a number of categories [4]. As described in [16] the most appropriate in the context of temporal rule mining is *intertransaction association rules.* Using an appropriate algorithm for finding intertransaction association rules, we can find rules on the form "*car* at time 0 and *hotel* at time 1 implies *leasing* at time 4". As can be seen, these algorithms produce rules with items from different transactions given by a timestamp. In order to find intertransaction association rules, we employ a variant of the FITI algorithm [20].

Rule Post-Processing. From the potentially high number of rules created during the rule mining, a very important and challenging problem is to find those that are *interesting*. Traditionally, measures like *support* and *confidence* have been used. Unfortunately, these measures have been shown to be less useful in text mining. One particular aspect of rule mining in text is that a high support often means the rule is too obvious and thus less interesting. These rules are often a result of frequently occurring terms and can partly be removed by specifying the appropriate stop words. However, many will remain, and these can to a certain extent be removed by specifying a maximum support on the rules, i.e., the only resulting rules are those above a certain minimum support and less than a certain maximum support. In section 5 we will describe two approaches more suitable in our context.

3.3 The Temporal Text Mining Testbench

In order to help discovering inter-version knowledge as well as developing new techniques for this purpose, we have developed the *Temporal Text Mining (TTM) Testbench* tool. The TTM Testbench is a user-friendly application that provides powerful operators for rule mining in temporal document collections, as well as providing extensibility for other text mining techniques.

The TTM Testbench consists of two applications: one for converting a document collection into the TTM format (essentially XML files containing the text and additional metadata), and one for performing the actual mining (which in general will be performed a number of times for each document collection, with different operations and parameters). A number of operations are available in the TTM Testbench, each essentially working as part of a filtering/operator pipeline. Examples of operators include **ExtractTerms**, **RemoveStopWords**, **FilterTerms**, **ExtractConcepts** and **FITI**. Text mining on a collection is performed by choosing which operations should be performed, and let the system perform the selected operations and present the final result. The result of a rule mining process is a number of rules, for example:

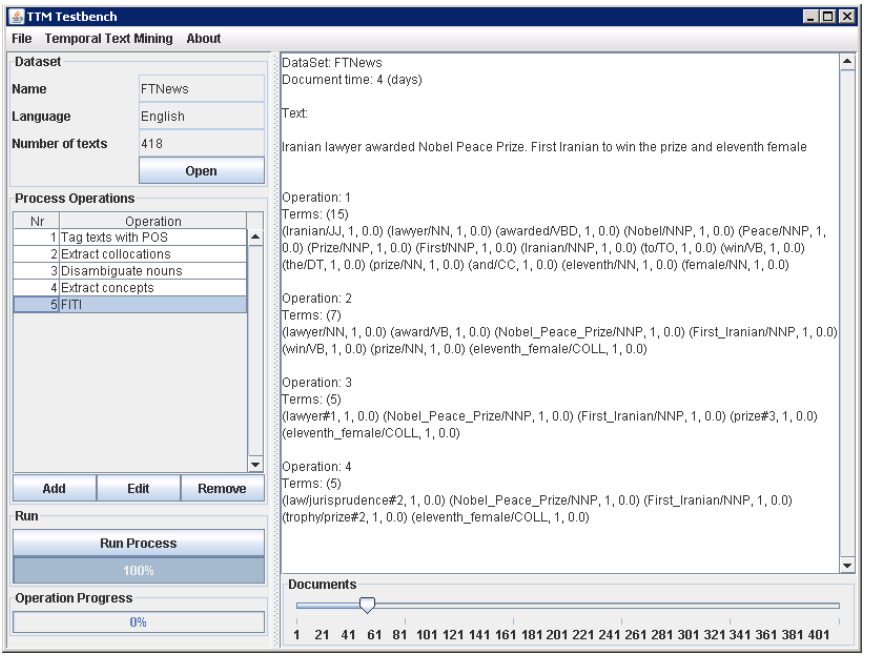


Fig. 1. Screenshot of TTM Testbench after performing operations on a document collection

Rule	Sup	Conf	Sim
(('attack' , 0) ('profits' , 1))→('bush' , 2)	0.11	1.0	0.3
...	...	...	...

The above rule says that if the word *attack* appears in a document version one day, and the word *profits* the day after, there is a high probability that the word *bush* will appear the third day (this is an actual example from mining a collection of Financial Times web pages). The last three columns give the *support*, *confidence*, and *semantic similarity* (to be described in more detail in Section 5.1) for the rule. Fig. 1 shows the TTM Testbench and the results after each text refinement operation, using the semantic operators which will be described in more detail below.

4 Semantic-Based Pre-processing

Performing the mining based on words extracted and refined as described in Section 3 did not achieve the desired quality. Factors contributing to the problem include those described above, i.e., feature dimensionality (i.e., vocabulary size) and feature sparsity, but also semantic aspects like synonyms (words having same or almost same meaning) and homonyms (words with same spelling but different meaning).

Considering semantics in the pre-processing phase could reduce the problems with synonyms and homonyms. In addition, by employing *concepts* instead of words in the rule mining process the dimensionality can be reduced, in addition to giving rules not found when not considering semantics. This is typically words that each have a low frequency but when represented as a common concepts could be important. An example

is the concept *vehicle* used instead of the words *bike car* and *lorry*. This considerably reduces dimensionality, in addition to giving rules containing these words higher support, and in that way increasing the probability that they will be found by the user, or detected automatically by the system.

We will in the following describe how semantic-based pre-processing and how it is integrated into the TTM Testbench. Note that we only consider semantics in the pre- and post-processing, while the mining is performed on semantic concepts in the same way as mining previously was performed on words.

The aim of the semantic-based pre-processing is twofold: find collocations (sequence of words or terms that occur together, for example *oil price*) and extract concepts (from single words or collocations). As will be described, this is performed in a multistep process involving: 1) part-of-speech tagging, 2) collocation extraction, 3) word-sense disambiguation (WSD), and 4) concept extraction.

For WSD and concept extraction we employ WordNet,[2] which essentially provides us with words and semantic relationships (for example hypernyms) between the words, and *synset*, which are words considered semantically equivalent (synonyms). For each word sense there is also a short description (gloss).

4.1 Part-of-Speech Tagging

Some word classes are more important than other in the mining process. In order to keep the number of participating terms as low as possible, it might be useful to filter out terms from only one or a few word classes from the text, for example nouns and adjectives. This can be performed by *part-of-speech tagging*. TTM Testbench uses the *Stanford Log-linear Part-Of-Speech Tagger*[3] to tag the document collection. This tagger uses a Maximum Entropy model, similar to stochastic tagging [19].

After the texts in the document collection are tagged, the operation extracts words tagged with one of a set of user-specified part-of-speech tags. Available tags include nouns, proper nouns and proper noun groups, verbs, adjectives, numbers and adverbs.

4.2 Collocation Extraction

A collocation is an expression consisting of two or more words that corresponds to some conventional way of saying things [14], for example *weapon of mass destruction* or *car bomb*. Collocations are common in natural languages, and a word can not be classified only on the basis of its meaning, sometimes co-occurrence with other words may alter the meaning dramatically.

The task of finding collocations is essentially to determine sequences of words or terms which co-occur more often than would be expected by chance. Hypothesis testing can be used to assess whether this is the case. In our work, the chi-square ($\mathcal{X}^2$) test has been used. When a noun occur together with another noun in the text they are collocation candidates, and the chi-square test is used to determine of they should be considered as a collocation.

[2] http://wordnet.princeton.edu/
[3] http://nlp.stanford.edu/software/tagger.shtml

4.3 Word-Sense Disambiguation

Word sense disambiguation (WSD) is the process of examining word tokens in a text and specify exactly which sense of each word is being used. As an example, consider the word *bank*, and two of its distinct senses: 1) a financial institution and 2) sloping land. When this word occur in a text, it is usually obvious for a human which of the senses of *bank* that is used, but creating robust algorithms for computers to automatically perform this task or more difficult.

In the TTM Testbench we employ the Lesk and adapted Lesk algorithms for WSD [1, 12]. Using these algorithms, the process of WSD consists of two steps: 1) find all possible senses for all the relevant words in a text, and 2) assign each word its correct sense. The first step is straightforward and accomplished by retrieving the possible senses from WordNet. The second step is accomplished by matching the context of the word in the document with the description of the senses in WordNet (glosses). Because the dictionary glosses tend to be fairly short, and may thus provide an insufficient vocabulary for fine-grained distinctions in relatedness, *extended gloss overlaps* is used to overcome the problem of too short glosses [1]. To create the extended gloss in the adapted Lesk algorithm, the algorithm uses the glosses of related words in WordNet (for example hypernyms, hyponyms, meronyms and holonyms for nouns, and hypernyms and troponyms for verbs).

4.4 Concept Extraction

Aiming at improving quality as well as reducing number of items in the mining process, terms are transformed into concept-level document features. This is done by utilizing the hierarchical structure of WordNet. Note that the concept extraction operation is dependent on WSD, since a word may have different senses, and these are linked to different synsets. The operation has three methods for finding concepts in a document. These are described in the following.

First, WordNet contains a relation called *category*. This relation links a synset to a higher-level category, where the category is represented by another synset. An example of this is that *basic training* is linked to the category *military*. By exploring this relation for each disambiguated word, it is possible to extract a set of categories which are descriptive of the contents of a document. Note however that only a limited set of the synsets in WordNet are linked to a category.

The second method of finding concepts in a document is based on finding common parent synsets of the words in the document. This is performed for each combination of disambiguated nouns in the texts. If the distance between the two words is below or equal to a user-specified threshold, the common parent synset is extracted as a concept. As an example of this, consider a part of the WordNet hierarchy, where *yen* and *euro* has *monetary unit* as a common ancestor, but while *euro* is direct child of *monetary unit*, *yen* is child of *Japanese monetary unit* which is child of *monetary unit*. Depending on the distance threshold, this may be extracted as a concept.

Finally, if no concepts was found using the two methods presented above, the user can specify that the parent node(s) of a word is to be extracted in addition to the word. This is found using the hypernym-relation. Recall the figure above, if only *euro* is

present in document, *monetary unit* can be extracted. This method may however result in very high feature dimensionality, and increase the complexity in the rule mining process.

In addition, this concept extraction operation tries to resolve the problem of synonyms in the text. This is done by replacing disambiguated words with the two first words in the synset it belongs to. The reason for using two words instead of only one, is that this may lead to more meaningful terms. For example, if the word *auto* is present in a document, and it belongs to the synset {car, auto, automobile, machine, motorcar}, then *auto* is replaced with the term *car/auto*. All words in the document collection which belong to this synset will therefore be represented by this term.

5 Post-processing

In general, the number of rules from rule mining of text will be very high. In order to reduce this to an amount that can be useful for a user, in the post-processing phase the most interesting rules are selected based on ranking the rules on some interestingness measure(s). Although the traditional support and confidence measures can be employed, these will often have less value in our context. For example, when mining temporal text databases, many interesting rules are rare, i.e., have a low support. We have studied the use of two other techniques that could have potential in our context. The measures are based on 1) semantic distance and 2) clustering.

5.1 Semantic Similarity

Words present in an association rule and that are close together (semantically related) in a knowledge hierarchy like WordNet, are more likely to be known by the user already. Therefore, rules where the words are less semantically related, can be considered more interesting [2].

The semantic similarity can then be used to rank the association rules. The higher the score, the more semantically similar the words in the antecedent and the consequent of the rule are. The rules with the lowest scores can therefore be considered interesting.

In order to calculate semantic distance we use the *JCn Measure* [9]. This measure is based on information content, defined in as the negative log likelihood of encountering an instance of the concept, i.e.:

$$IC(c) = -log(\frac{freq(c)}{N})$$

where $freq(c)$ is the frequency of the concept, and N is the number of concepts in the corpus. The similarity measure of two concepts, c_1 and c_2, is then defined by the following formula, where c is the most specific concept in common between c_1 and c_2 (for example, the most specific concept in common between *desktop computer* and *portable computer* could be *personal computer*):

$$sim(c_1, c_2) = IC(c_1) + IC(c_2) - 2 * IC(c)$$

This measure is calculated after the association rules have been mined. The score of an association rule is calculated as the average semantic similarity between the words

in the antecedent and the consequent of the rule. However, note that it is only possible to calculate semantic similarity with disambiguated nouns or collocations which are present in WordNet. This is because the similarity is calculated between synsets, and the sense is needed to know which synset a word is present in.

5.2 Clustering

Many association rules can be said to display commonsense, for example *hammer⇒nail*. *Hammer⇒shampoo* on the other hand, is more interesting because hammer has little relation with shampoo, they can be said to be dissimilar. With this in mind, dissimilarity between the items can be used to judge the interestingness of a pattern. Based on the approach for structured data presented by Zhao et al. [21] we have experimented with clustering to measure the dissimilarity between items in an association rule.

In the first step of clustering-based rule selection, the document collection is clustered so that documents are grouped together according to their contents. Then, given an association rule A⇒B where A is in cluster C_A and B is in C_B, interestingness is defined as the distance between the two clusters C_A and C_B:

$$Interestingness(A \Rightarrow B) = Dist(C_A, C_B)$$

If the antecedent or consequent consists of more than one item, interestingness is defined as the minimal distance between clusters of antecedent and consequent. Finally, only rules which have terms from different clusters in the antecedent and the consequent are presented to the user.

6 Experiments and Results

This section presents some of the results from applying semantics in the rule mining. The experiments have been performed using the TTM Testbench, extracting collocations and concepts as described above, resulting in association rules that span across texts with different timestamps.

Filtering and weighting (cf. Section 3.2) have not be employed, since the IDF part of TF-IDF dampens the weight of terms which appear in many documents. This may not be always be desired, since association rules containing frequent terms in some cases can be interesting.

A number of document collections based on web newspapers have been used in the experiments. Each document collection have been created by downloading the web page once a day. Due to space constraints we will in this paper only report from the use of a collection based on Financial Times pages. Mining the other collections gave similar results. Due to limitations on the FITI implementation where the memory usage increases with document collection size, we have in the reported results used a relatively small collection consisting of only 107 documents.

The parameters for the operations are as follows:

- Collocation extraction: Only verbs and adjectives are extracted in addition to collocations and single-word nouns.

- Word sense disambiguation: The adapted Lesk algorithm is used with context size of 6 words, and verbs and adjectives are not kept after the disambiguation process.
- Concept extraction: We set the maximum distance in the WordNet hierarchy to 5 (this includes the words themselves), parent nodes of words with no concepts are not added, and original terms are not kept when a concept is found.

As a result, the following terms will be extracted from each document and used in the rule mining process:

- Collocations.
- Proper nouns and proper noun groups.
- Common parents between terms in the same document.
- Categories.
- Disambiguated nouns with no common parent or category.
- Nouns which have not been disambiguated.

The parameter values for the FITI algorithm:

Parameter	Value
Minimum support	0.1
Maximum support	0.5
Minimum confidence	0.5
Maximum confidence	1.0
Maxspan (time/days)	3
Max set size (terms in rule)	3

Unfortunately, experiments showed that determining interestingness based on clustering did not work particularly well. As as result, we used only the semantic similarity measure for rating rules (note that only rules containing at least one disambiguated word on each side of the rule will get a score).

6.1 Evaluation Criteria

Automatically deciding if a rule is interesting or not, is difficult, if not impossible. The main focus in this project will be to see if the association rules and their items are meaningful, and to study whether there is any difference between rules with a high semantic similarity and rules with low semantic similarity; the idea is that rules with low semantic similarity are more interesting than those with high similarity.

6.2 Results From Mining the Financial Times Collection

The result of this experiment was 56 rules (the complete set of rules is given in Table 1). In Tables 2–4, a subset of 15 rules are presented: The 5 first with no semantic similarity (Table 2, keep in mind that it is not possible to calculate the semantic similarity of rules not containing any disambiguated terms, concepts or categories, these will therefore get a semantic similarity of zero and thus appear first in the result set), the 5 with lowest semantic similarity (Table 3), and the 5 with highest semantic similarity (Table 4). The

Table 1. Complete Set Of Rules

Rule#	Rule	Sup	Conf	SemSim
1	{('europe/nnp',0)} → {('market/marketplace#1',1)}	0.16	0.52	0.0000
2	{('china/nnp',0)} → {('military/armed_forces#1',1)}	0.21	0.54	0.0000
3	{('russia/nnp',0)} → {('military/armed_forces#1',1)}	0.12	0.54	0.0000
4	{('iraq/nnp',0)} → {('military/armed_forces#1',1)}	0.10	0.52	0.0000
5	{('uk/nnp',0)} → {('military/armed_forces#1',1)}	0.19	0.53	0.0000
6	{('year#3',0)} → {('china/nnp',1)}	0.10	0.50	0.0000
7	{('europe/nnp',0)} → {('china/nnp',1)}	0.16	0.52	0.0000
8	{('year#3',0)} → {('europe/nnp',1)}	0.11	0.55	0.0000
9	{('commercial_enterprise/business_enterprise#2',0)} → {('eu/nnp',1)}	0.13	0.52	0.0000
10	{('uk/nnp',0)} → {('eu/nnp',1)}	0.18	0.50	0.0000
11	{('depository_financial_institution/bank#1',0)} → {('eu/nnp',2)}	0.13	0.52	0.0000
12	{('russia/nnp',0)} → {('military/armed_forces#1',2)}	0.14	0.62	0.0000
13	{('iraq/nnp',0)} → {('military/armed_forces#1',2)}	0.11	0.57	0.0000
14	{('sarkozy/nnp',0)} → {('military/armed_forces#1',2)}	0.13	0.56	0.0000
15	{('uk/nnp',0)} → {('military/armed_forces#1',2)}	0.21	0.58	0.0000
16	{('europe/nnp',0)} → {('military/armed_forces#1',1)}	0.18	0.58	0.0000
17	{('russia/nnp',0)} → {('head/chief#4',2)}	0.11	0.50	0.0000
18	{('russia/nnp',0)} → {('president_of_the_united_states/united_states_president#1',2)}	0.12	0.54	0.0000
19	{('russia/nnp',0)} → {('head/chief#4',1)}	0.11	0.50	0.0000
20	{('eu/nnp',0)} → {('military/armed_forces#1',2)}	0.21	0.55	0.0000
21	{('china/nnp' 'market/marketplace#1',0)} → {('military/armed_forces#1',1)}	0.11	0.55	0.0593
22	{('market/marketplace#1',0)} → {('military/armed_forces#1',2)}	0.23	0.59	0.0593
23	{('market/marketplace#1',0)} → {('military/armed_forces#1',1)}	0.23	0.59	0.0593
24	{('china/nnp' 'market/marketplace#1',0)} → {('military/armed_forces#1',2)}	0.12	0.59	0.0593
25	{('company#1',0)} → {('market/marketplace#1',2)}	0.15	0.50	0.0600
26	{('investor#1',1) ('uk/nnp',0)} → {('military/armed_forces#1',2)}	0.10	0.85	0.0600
27	{('investor#1' 'uk/nnp',0)} → {('military/armed_forces#1',1)}	0.12	0.72	0.0600
28	{('investor#1',0)} → {('military/armed_forces#1',1)}	0.28	0.69	0.0600
29	{('investor#1',0)} → {('military/armed_forces#1',2)}	0.22	0.55	0.0600
30	{('investor#1' 'uk/nnp',0)} → {('military/armed_forces#1',2)}	0.11	0.67	0.0600
31	{('week/hebdomad#1',0)} → {('military/armed_forces#1',1)}	0.11	0.52	0.0607
32	{('investor#1' 'president_of_the_united_states/united_states_president#1' , 0)} → {('military/armed_forces#1',1)}	0.14	0.83	0.0611
33	{('china/nnp',0) ('president_of_the_united_states/united_states_president#1' , 1)} → {('military/armed_forces#1',2)}	0.11	0.75	0.0622
34	{('china/nnp' 'president_of_the_united_states/united_states_president#1' , 0)} → {('military/armed_forces#1',1)}	0.11	0.71	0.0622
35	{('president_of_the_united_states/united_states_president#1' , 0)} → {('military/armed_forces#1',1)}	0.23	0.63	0.0622
36	{('investor#1' 'head/chief#4',0)} → {('military/armed_forces#1',1)}	0.12	0.72	0.0635
37	{('conflict/struggle#1',0)} → {('military/armed_forces#1',2)}	0.10	0.65	0.0637
38	{('year#3',0)} → {('military/armed_forces#1',1)}	0.11	0.55	0.0638
39	{('head/chief#4',0)} → {('military/armed_forces#1',1)}	0.20	0.64	0.0670
40	{('head/chief#4',0)} → {('military/armed_forces#1',2)}	0.17	0.55	0.0670
41	{('commercial_enterprise/business_enterprise#2',0)} → {('military/armed_forces#1',2)}	0.18	0.70	0.0674
42	{('occupation/business#1',0)} → {('military/armed_forces#1',2)}	0.11	0.60	0.0703
43	{('military/armed_forces#1',1) ('president_of_the_united_states/united_states_president#1' , 0)} → {('investor#1',2)}	0.14	0.62	0.0735
44	{('time#5',0)} → {('military/armed_forces#1',1)}	0.11	0.60	0.0742
45	{('time#5',0)} → {('military/armed_forces#1',2)}	0.10	0.55	0.0742
46	{('military/armed_forces#1',1) ('head/chief#4',0)} → {('investor#1',2)}	0.12	0.62	0.0784
47	{('military/armed_forces#1' 'head/chief#4',0)} → {('investor#1',2)}	0.10	0.61	0.0784
48	{('country/state#1',0)} → {('investor#1',2)}	0.12	0.62	0.0817
49	{('president_of_the_united_states/united_states_president#1' , 0)} → {('investor#1',2)}	0.19	0.53	0.0870
50	{('company#1' 'president_of_the_united_states/united_states_president#1' , 0)} → {('military/armed_forces#1',1)}	0.13	0.74	0.0873
51	{('company#1' 'president_of_the_united_states/united_states_president#1' , 0)} → {('military/armed_forces#1',2)}	0.10	0.58	0.0873
52	{('president_of_the_united_states/united_states_president#1' 'head/chief#4',0)} → {('investor#1',2)}	0.11	0.71	0.0919
53	{('head/chief#4',0)} → {('investor#1',2)}	0.17	0.55	0.0968
54	{('depository_financial_institution/bank#1',0)} → {('military/armed_forces#1',2)}	0.13	0.52	0.0973
55	{('company#1',0)} → {('military/armed_forces#1',2)}	0.16	0.53	0.1124
56	{('company#1',0)} → {('military/armed_forces#1',1)}	0.17	0.56	0.1124

Table 2. The 5 first rules with no semantic similarity

Rule#	Rule
1	{('europe/nnp', 0)} → {('market/marketplace#1', 1)}
2	{('china/nnp', 0)} → {('military/armed_forces#1', 1)}
3	{('russia/nnp', 0)} → {('military/armed_forces#1', 1)}
4	{('iraq/nnp', 0)} → {('military/armed_forces#1', 1)}
5	{('uk/nnp', 0)} → {('military/armed_forces#1', 1)}

Table 3. The 5 rules with the lowest semantic similarity

Rule#	Rule
21	{('china/nnp' 'market/marketplace#1', 0)} → {('military/armed_forces#1', 1)}
22	{('market/marketplace#1', 0)} → {('military/armed_forces#1', 2)}
23	{('market/marketplace#1', 0)} → {('military/armed_forces#1', 1)}
24	{('china/nnp' 'market/marketplace#1', 0)} → {('military/armed_forces#1', 2)}
25	{('company#1', 0)} → {('market/marketplace#1', 2)}

Table 4. The 5 rules with the highest semantic similarity

Rule#	Rule
52	{('president_of_the_united_states/united_states_president#1' 'head/chief#4', 0)} → {('investor#1', 2)}
53	{('head/chief#4', 0)} → {('investor#1', 2)}
54	{('depository_financial_institution/bank#1', 0)} → {('military/armed_forces#1', 2)}
55	{('company#1', 0)} → {('military/armed_forces#1', 2)}
56	{('company#1', 0)} → {('military/armed_forces#1', 1)}

Table 5. Potentially interesting rule

Rule#	Rule
33	{('china/nnp', 0) ('president_of_the_united_states/united_states_president#1', 1)} → {('military/armed_forces#1', 2)}

rule numbers in the rules presented for the individual experiments refer to their number in the full result set.

The terms present in the rules will sometimes include the symbol #, this is used to indicate the sense number of the term in WordNet. This can be used by the user in a lookup in WordNet (for example, the user can determine whether the term *market/marketplace* in rule 22 means a physical location in a city or the world of commercial activity). Another symbol which may appear, is */nnp*. This means that the term is a proper noun. One aspect that becomes clear when inspecting the rules is that it is easier to understand the meaning of the items when they are represented by two synonyms. As an example, see rule 54. Here the item *depository_financial_institution/bank* is present. Because a synonym is present, the rule is more meaningful than if for example only *bank* was present.

As the above show, many of the terms included in the rules are meaningful, and the user can therefore make sense of the discovered rules. Whether semantic similarity is able to distinguish between interesting and uninteresting rules or not, is difficult to decide. The reason for this is that it is not entirely clear what an interesting association rule would look like when mining for association rules in web newspapers.

When looking at the rules from this experiment, it becomes apparent that rule number 33 (Table 5) may also be considered interesting. Consider for example that there is an article discussing an event in China at time 0, then the next day a related article appears where the US President is mentioned. Finally, at time 2 an article containing military news which is related to the two previous articles appear. It is however difficult to know whether these cases are related, or just coincidental. But it gives an indication that it may in fact be possible to detect interesting temporal relationships between news items from different versions of the front page of a web newspaper.

6.3 Summary

The experiments have shown that the main problem of mining textual association rules from web newspapers is that it is difficult, if not impossible, to clearly see which rules are interesting. However, the rules found using the new document feature extraction operations can be said to make sense. Contributing to this is also that synonyms are added to words if available, and thus *head/chief* is easier to understand than only the word *head*.

When it comes to using semantic similarity for rating association rules, it is still an open question whether this can lead to good results. The reason for this is that identifying interesting rules is difficult, and it is therefore not possible to say if rules with low semantic similarity are more interesting than rules with high similarity.

A problem that could affect the usefulness of semantic similarity, is the difficulty of assigning the correct sense to a word. An evaluation of a number of random texts from the document collection showed a precision of only about 35%, which is similar to what has been reported in previous work [1]. In addition, in some cases it was not possible to determine if the correct sense was assigned to a word. The reason for this is that the senses in WordNet are very fine-grained, and it is difficult to spot the difference (also reported by Hovy et al. [7]).

The implication of this problem to the results of this project, is that care must be taken when looking at the association rules since some of the terms may be present due to erroneous word sense disambiguation. However, many words which are disambiguated incorrectly will be filtered out during the rule mining process because their support in the document collection as a whole is too low.

One of the problems with interestingness when mining for association rules in web newspapers, is that what may seem like an interesting rule, really is a coincidence. Consider for example the rule given in the problem description, namely $\{(\text{'Bomb'}, 0)\} \rightarrow \{(\text{'Terror'}, 1)\}$. At first glance, this rule may seem interesting. But after further inspection it may become clear that the news article containing the word 'terror' is in no way related to the article containing 'bomb', instead it may relate to a totally different event and the association rule is totally coincidental.

7 Conclusions and Further Work

In this paper we have extended the previous work on mining association rules in temporal document collections by performing mining based on *semantics* as well as studying the impact of additional techniques for ranking of rules. Based on result from experiments we have illustrated the usefulness of employing semantics in this context, and shown that the impact of using semantic similarity for ranking rules is questionable at best.

Future work will go in two directions: 1) further development of appropriate metrics for rule quality, and 2) improvement of the actual rule mining, so that larger document collections can be mined as well as reducing processing time for smaller collections.

Acknowledgments. The authors would like to thank Trond Øivind Eriksen and Kjell-Inge Skogstad who developed the basic ideas for mining association rules in temporal document collections and implemented the TTM Testbench.

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 (2003)
2. Basu, S., Mooney, R.J., Pasupuleti, K.V., Ghosh, J.: Evaluating the novelty of text mined rules using lexical knowledge. In: Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining (2001)

3. Chakrabarti, S.: Mining the Web - Discovering Knowledge from Hypertext Data. Morgan Kaufmann, San Francisco (2003)
4. Dunham, M.: Data Mining: Introductory and Advanced Topics. Prentice-Hall, Englewood Cliffs (2003)
5. Feldman, R., Sanger, J.: The Text Mining Handbook. Cambridge (2007)
6. Holt, J.D., Chung, S.M.: Efficient mining of association rules in text databases. In: Proceedings of CIKM 1999 (1999)
7. Hovy, E.H., Marcus, M.P., Palmer, M., Ramshaw, L.A., Weischedel, R.M.: OntoNotes: the 90% solution. In: Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (2006)
8. Janetzko, D., Cherfi, H., Kennke, R., Napoli, A., Toussaint, Y.: Knowledge-based selection of association rules for text mining. In: Proceedings of ECAI 2004 (2004)
9. Jiang, J.J., Conrath, D.W.: Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In: Proceedings of International Conference Research on Computational Linguistics (ROCLING X) (1997)
10. Lee, C.-H., Lin, C.-R., Chen, M.-S.: On mining general temporal association rules in a publication database. In: Proceedings of ICDM 2001(2001)
11. Lent, B., Agrawal, R., Srikant, R.: Discovering trends in text databases. In: Proceedings of KDD 1997 (1997)
12. Lesk, M.: Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In: Proceedings of the 5th Annual International Conference on Systems Documentation (SIGDOC 1986) (1986)
13. Lu, H., Feng, L., Han, J.: Beyond intratransaction association analysis: mining multidimensional intertransaction association rules. ACM Trans. Inf. Syst. 18(4), 423–454 (2000)
14. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
15. Mei, Q., Zhai, C.: Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In: Proceedings of KDD 2005 (2005)
16. Nørvåg, K., Skogstad, K.-I., Eriksen, T.: Mining association rules in temporal document collections. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS, vol. 4203. Springer, Heidelberg (2006)
17. Roddick, J.F., Spiliopoulou, M.: Survey of temporal knowledge discovery paradigms and methods. IEEE Transactions on Knowledge and Data Engineering 14(4), 750–767 (2002)
18. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the right interestingness measure for association patterns. In: Proceedings of KDD 2002 (2002)
19. Toutanova, K., Manning, C.: Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In: Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (2000)
20. Tung, A.K.H., Lu, H., Han, J., Feng, L.: Efficient mining of intertransaction association rules. IEEE Transactions on Knowledge and Data Engineering 15(1), 43–56 (2003)
21. Zhao, Y., Zhang, C., Zhang, S.: Discovering interesting association rules by clustering. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS, vol. 3339, pp. 1055–1061. Springer, Heidelberg (2004)

Generating Executable Scenarios from Natural Language

Michal Gordon and David Harel

The Weizmann Institute of Science, Rehovot, 76100, Israel
{michal.gordon,dharel}@weizmann.ac.il

Abstract. Bridging the gap between the specification of software requirements and actual execution of the behavior of the specified system has been the target of much research in recent years. We have created a natural language interface, which, for a useful class of systems, yields the automatic production of executable code from structured requirements. In this paper we describe how our method uses static and dynamic grammar for generating live sequence charts (LSCs), that constitute a powerful executable extension of sequence diagrams for reactive systems. We have implemented an automatic translation from controlled natural language requirements into LSCs, and we demonstrate it on two sample reactive systems.

1 Introduction

Live Sequence Charts are a visual formalism that describes natural "pieces" of behavior and are similar to telling someone what they may and may not do, and under what conditions. The question we want to address here is this: can we capture the requirements for a dynamic system in a far more natural style than is common? We want a style that is intuitive and less formal, and which can also serve as the system's executable behavioral description [1].

To be able to specify behavior in a natural style, one would require a simple way to specify pieces of requirements for complex behavior, without having to explicitly, and manually, integrate the requirements into a coherent design. In [2], the mechanism of *play-in* was suggested as a means for making programming practical for lay-people. In this approach, the user specifies scenarios by playing them in directly from a graphical user interface (GUI) of the system being developed. The developer interacts with the GUI that represents the objects in the system, still a behavior-less system, in order to show, or teach, the scenario-based behavior of the system by example (e.g., by clicking buttons, changing properties or sending messages). As a result, the system generates automatically, and on the fly, live sequence charts (LSCs) [3], a variant of UML sequence diagrams [4] that capture the behavior and interaction between the environment and the system or between the system's parts. In the current work we present an initial natural language interface that generates LSCs from structured English requirements.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 456–467, 2009.

An LSC describes inter-object behavior, behavior between objects, capturing some part of the interaction between the system's objects, or between the system and its environment. LSCs distinguish the possible behavior from the necessary behavior (i.e., liveness, which is where the term "live" comes from), and can also express forbidden behavior — scenarios that are not allowed, and more. Furthermore, LSCs are fully executable using the *play-out* mechanism developed for LSCs in [2], and its more powerful variants [5,6]. To execute LSCs the play-out mechanism monitors at all times what must be done, what may be done and what cannot be done, and proceeds accordingly. Although the execution does not result in an optimal code, nor is the executed artifact deterministic (since LSC are under-specified) it is nevertheless a complete execution of the LSC specification. The execution details are outside the scope of this paper, but are described in detail in [5,2].

By its nature, the LSC language comes close to the way one would specify dynamic requirements in a natural language. We suggest to take advantage of this similarity, and to translate natural language requirements directly into LSCs, and then render them fully executable. One interesting facet of this idea is rooted in the fact that the natural and intuitive way to describe behavioral requirements will generate fragmented multi-modal pieces of behavior which is also the main underling philosophy of LSCs. The play-out mechanisms are able to consider all the fragmented pieces together as an integrated whole, yielding a fully executable artifact. Thus, our translation into LSCs can be viewed as a method for executing natural language requirements for reactive systems.

As to related work (discussed more fully later), we should say here that natural language processing (NLP) has been used in computer-aided software engineering (CASE) tools to assist human analysis of the requirements. One use is in extracting the system classes, objects, methods or connections from the natural language description [7,8]. NLP has been applied to use case description in order to create simple sequence diagrams with messages between objects [9], or to assist in initial design [10]. NLP has also been used to parse requirements and to extract executable code [11] by generating object-oriented models. However, it is important to realize, that the resulting code is intra-object — describes the behavior of each object separately under the various conditions, and it is usually limited to sequential behavior. The resulting OO artifact is focussed on object-by-object specification, and is not naturally inter-object.

The paper is structured as follows: Section 2 contains some brief preliminaries, Section 3 presents an overview of the translation method, and Section 4 demonstrates the details using an example. Section 5 discusses related work and Section 6 concludes.

2 Preliminaries

In its basic form, an LSC specifies a multi-modal piece of behavior as a sequence of message interactions between object instances. It can assert mandatory behavior — what must happen (with a hot temperature) — as well as possible

behavior — what may happen (with a cold temperature). The LSC language [3] has its roots in message sequence charts (MSC) [12] or its UML variant, sequence diagrams [4], where objects are represented by vertical lines, or lifelines, and messages between objects are represented by horizontal arrows between objects. Time advances along the vertical axis and the messages entail an obvious partial ordering. Figure 1 shows a sample LSC. In this LSC the *prechart* events, those that trigger the scenario, appear in the top blue hexagon; in this case, a cold (dashed blue) click event from the user to the `c` button. If the prechart is satisfied, i.e., its events all occur and in the right order, then the main chart (in the black solid rectangle) must be satisfied too. In the example, there is a hot (solid red) event where the light state changes to `on` and a cold condition, in the blue hexagon, with a hot event in the subchart it creates. The meaning is that if the display mode is not `time`, then it must change to `time`. There is no particular order between the events in the main chart in the example, although in general there will be a partial order between them, derived from the temporal constraints along the vertical lifelines.

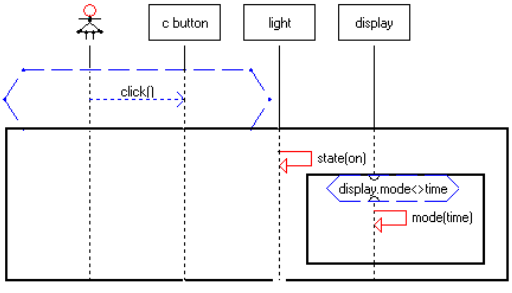


Fig. 1. A simple LSC. The prechart (the blue dashed hexagon) contains the cold event (blue dash arrow) "user clicks the `c` button", while the main chart (the black solid rectangle) shows two hot events (red solid arrow): one shows the light state changing to `on` and the other is a hot event with a cold condition (blue dashed hexagon) that specifies that if the mode is not `time` then it must change to `time`.

The basic LSC language also includes conditions, loops and switch cases. In [2], it has been significantly enriched to include time, scoped forbidden elements, and symbolic instances that allow reference to non-specific instances of a class.

Later, we will be describing a context-free grammar for behavioral requirements that will serve as our controlled English language. To recall, a context-free grammar (CFG) is a tuple $G = (T, N, S, R)$, where T is the finite set of terminals of the language, N is the set of non-terminals, that represent phrases in a sentence, $S \in N$ is the start variable used to represent a full sentence in the language, and R is the set of production rules from N to $(N \cup T)^*$. In the LSC grammar, parts of the grammar are static T_S and other parts are dynamic T_D.

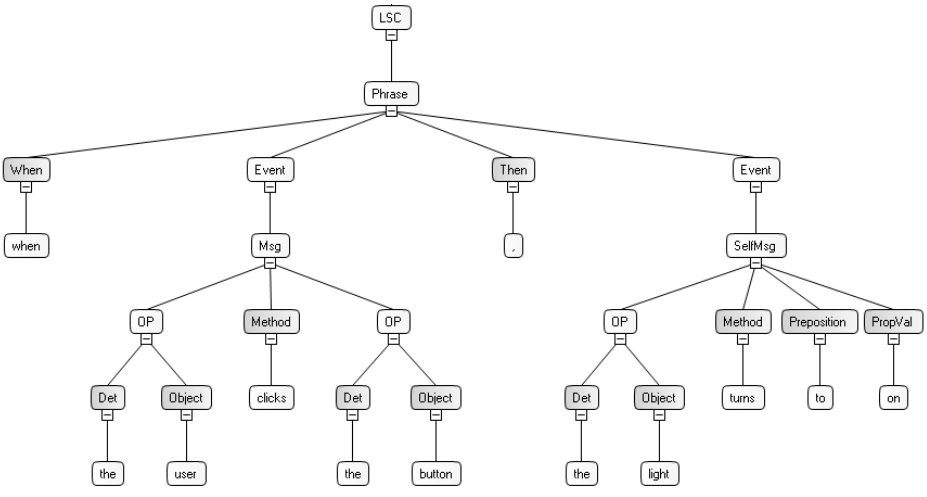


Fig. 2. The parse tree for the sentence "when the user clicks the button, the light turns on". The parts of the LSC grammar detected are shown. There is one message *Msg* which is a message from object phrase (*OP*) `user` to object phrase `button`, and another self message *SelfMsg* of object `light` with method `turn` and argument `on`.

3 Overview of LSC Grammar

Requirements are a way of describing scenarios that must happen, those that can happen, and those that are not allowed to happen. The static terminals describe the flow of the scenario; e.g., "*when* something happens *then* another thing should happen", or "*if* a certain condition holds *then* something *cannot* occur". The dynamic terminals refer to the model, the objects and their behaviors.

The static terminal symbols are *if, then, must, may* etc. They are relevant for inferring the semantics of LSCs. The dynamic terminals are all unrecognized terminals processed by a dictionary and transformed from part of speech to possible parts of the model. They are grouped into `objects`, `properties`, `methods` and `property values` which are not mutually exclusive.

For example in: "The user presses the button", `user` and `button` are both objects. Similarly, `presses` is a verb that is added to the methods terminal list. Other types of terminals are properties and property values. These can be identified as in the following example: "the display color changes to red", where the noun `color`, which is part of the noun phrase, is a property of the display object and the adjective `red` is a possible property value. Property values may also include possible variables for methods.

Figure 2 displays the parse tree for the requirement: "when the user clicks the `b` button, the light turns to on". When analyzing the parse tree, the *when* and *then* hint to where the prechart ends and the main chart begins, the messages added are `click` from the user to the button in the prechart and `turn` with a parameter `on` in the main chart, as seen in Fig. 3(a).

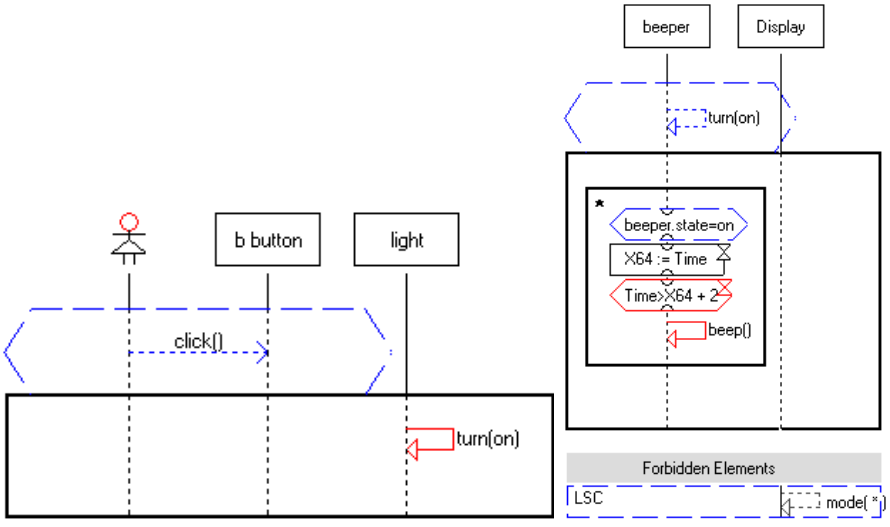


Fig. 3. Sample LSCs. (a) A simple LSC created for the sentence: "when the user clicks the `b` button, the light turns on". (b) A more complex LSC created for the sentence: "when the beeper turns on, as long as the beeper state is on, if two seconds have elapsed, the beeper beeps and the display mode cannot change".

The grammar is inherently ambiguous, due to use of dictionary terminals. The same word could be used for noun, object or property value. We therefore parse each sentence separately and update the grammar as the user resolves ambiguities relevant to the model. Our parser is an active chart parser, bottom-up with top-down prediction [13]. We detect errors and provide hints for resolving them using the longest top-down edge with a meaningful LSC construct. For example a message or a conditional expression that have been partially recognized provide the user with meaningful information.

4 LSC Grammar Constructs

4.1 Example Requirements Translation

We now describe the main parts of our method for automatically translating structured requirements into LSCs. We demonstrate the main language phrases by constructing a simplified version of a digital watch described in [14]. There, the watch behavior was described using statecharts formalism. Here, we describe the same system in natural language and then automatically transform it into LSCs. Generally, the watch displays the time and can switch between different displays that show (and allow changes to) the alarm, date, time and stopwatch. It has an option to turn on a light, and it has an alarm that beeps when the set time arrives.

An example, taken verbatim from [14] is this: "[The watch] has an alarm that can also be enabled or disabled, and it beeps for 2 seconds when the time in the alarm is reached unless any one of the buttons is pressed earlier". This requirement is ambiguous and unclear for our purposes: when a button is pressed should the alarm time be cancelled or should the beeping stop? Basic user knowledge of the system helps us infer that the beeper should stop. Also, the fact that the alarm beeps only when it is enabled is deduced by common knowledge, as it is not explicit in the text. The structured requirements for these will be: "when the time value changes, if the time value equals the alarm value and the alarm state is enabled, the beeper turns on"; "when the beeper turns on, if two minutes have elapsed, the beeper turns to off"; "when the user presses any button, the beeper shall turn off". Although the original requirement is fragmented and separated into several requirements, the combined effect of these requirements will achieve the same goal.

4.2 Translating Constructs

In this section we show how our initial grammar translates controlled natural language to LSCs. The grammar is structured and required rigid and clear requirements, however they are natural to understand and compose. Since we allow multiple generations of similar constructs we hope to enlarge the possible specifications. We shall describe how the basic structures — messages and property changes, and some of the less trivial ideas that include parsing temperature, conditions, loops and symbolic objects. Few advanced ideas such as asserts and synchronization are not supported at the current time, nevertheless, the current grammar allows implementing executable systems and has been tested on the digital watch example and on an ATM machine example.

Messages. The simplest language construct in LSCs is the message between objects, or from an object to itself. Messages can be method calls or property changes. In the case of methods, the verb specifies the method to call. For example "the `c` button is clicked" is mapped into a *self message* from the `c` button to itself. Messages can also be specified between objects as in "the user presses the `c` button". Parameters can also be used as in: "the light turns to on", in which case the `turn` method of the `light` is invoked with a value of `on` as a parameter. When a sentence can be fully parsed into more than one basic structure, the user is notified of the location and selects the terminal to use for the word. For example is the button an argument for press or an object with the method press. The user selection is integrated into the dictionary using weights which effectively cause the button in the rest of the text to be an object, unless specified differently.

Temperature. LSCs allow the user to specify whether something may happen, for which we use a *cold* temperature (depicted in dashed blue lines), or what must happen, which is *hot* (depicted by solid red lines). The grammar allows the user to specify the temperature explicitly by using the English language constructs

may or *must* and some of their synonyms. If the user does not explicitly specify the temperature of the event, it is inferred from the sentence structure. For example, the *when* part is cold and the *then* part is hot. In English it is obvious that the *when* part may or may not happen, but that if it does then the *then* part must happen. See Fig. 4 for an example.

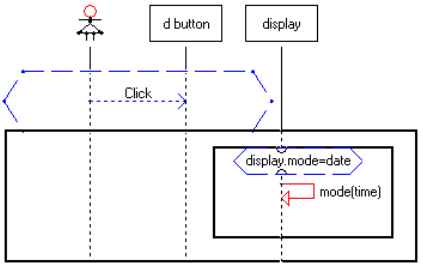


Fig. 4. The LSC created for the sentence "when the user presses the `d` button, if the display mode is date, the display mode changes to time". The message in the *when* part is cold (dashed blue arrow), while the messages in the *then* part are hot (solid red arrows).

Conditions. Conditions, that are frequent in system requirements are readily translated into conditions in the LSC formalism. The grammar accepts expressions that query an object's property values, such as "if the display mode is time". The condition is implemented in the LSC as a cold condition, and all phrases that occur in the *then* part of the phrase appear in the subchart of the condition. The dangling-else ambiguity that appears frequently in programming languages is resolved similar to most parsers by choosing the 'else' that complete the most recent 'if', which is reasonable also in natural text. We allow the user to manipulate the hierarchical structure of the sentence using commas and conjunctions, see, for example, Fig. 5.

Symbolic Objects. In English, definite or indefinite *determiners* are used to specify a specific object or a non-specific object respectively. The determiners are part of the static terminals that differentiate between objects and symbolic objects. Consider the sentence "when the user presses any button, the beeper shall turn to off". The requirement is translated into the LSC of Fig. 6, where the `button` is symbolic (drawn with a dashed borderline) and can be any of the buttons. The LSC semantics also requires that a symbolic object becomes bound using an interaction with another object or a property. Thus, the sentence "when the user presses a button, a display turns on" is not valid, since the `display` is not bound at all and is supposedly symbolic. It is clear that the sentence is ambiguous also to an English reader, and the user is prompt to resolve the problem.

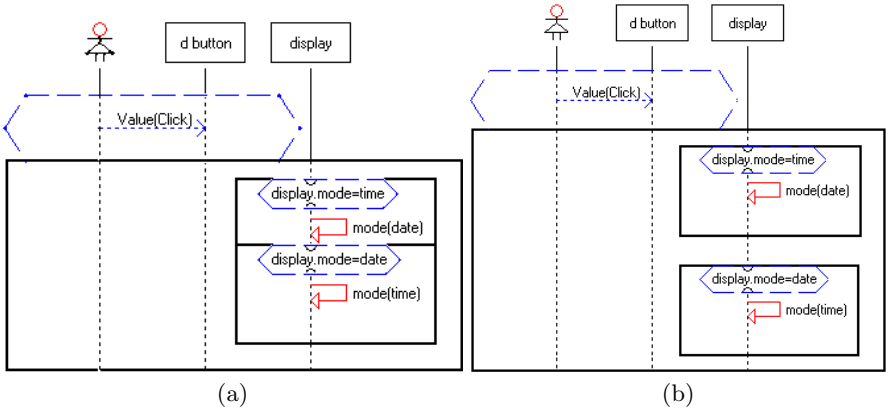


Fig. 5. Conditions in LSCs. (a) The LSC created for the sentence "when the user presses the `d` button, if the display mode is time, the display mode changes to date, otherwise if the display mode is date, the display mode changes to time". (b) Shows what would happen if the *otherwise* would be replaced by an *and*. The second condition is not an alternative to the first, and the behavior would not be as expected. Consider, for example, what would happen if the `display mode` is `time`: the execution would enter both conditions and nothing would happen to the display mode. This behavior could also be avoided by separating the single requirement into two different requirements, resulting in two separate LSCs.

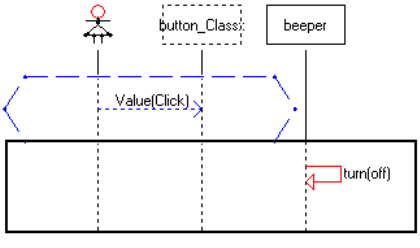


Fig. 6. The LSC created for the sentence "when the user presses any button, the beeper shall turn to off". The `button` object referred to by the user is a non-specific object and is therefore translated as a symbolic object of the button class, shown using a dashed box.

Forbidden Elements. Our grammar also supports forbidden elements when using negation of messages. For example, "the display mode cannot change" would result in a forbidden element. The scope of forbidden elements is important to the semantics of LSCs; i.e., to what parts of the LSC they are relevant. We use the syntax tree and the location of the forbidden statement in it to resolve the scope, conjunction can be used to verify that a forbidden phrase is inside a subchart. See Fig. 3 (b) for an example.

Forbidden Scenarios. In addition to specifying negative events as forbidden elements, one can also specify forbidden scenarios — scenarios that cannot happen. These are specified using language phrases such as "the following can never happen", prefixing the scenario that is to be forbidden. In the LSC, the scenario described is created in the prechart with a hot false condition in the main chart, which entails a violation if the prechart is completed. To separate the 'when' from the 'then' parts of the scenario, we add a synchronization of all the objects referenced in the scenario at the end of the 'when' part as extracted from the syntax tree.

Additional Constructs. The grammar supports translation into additional LSC constructs, such as local variables, time constraints, loops and non-determinism. It is currently of preliminary nature and is being extended to deal with additional ways of specifying new and existing constructs to make it more natural to users. The fact that sentences are parsed separately allows the use of the ambiguous grammar. Resolution of ambiguity is achieved by interaction with the user to obtain information about the model and by propagating model information between different sentences.

4.3 Implementation and Execution

Once the requirements are parsed and the model is known, the objects and their basic methods are implemented separately with the names extracted from the text. We use the dictionary to extract word stems and we also support word phrases for methods or objects by concatenating the words with a hyphen. We implemented the watch's simple interface with the Play-Engine GUIEdit tool described in [2]. In the final implementation, logical objects that have properties or methods, that do not effect the system visually and do not need additional implementation, are created automatically in the Play-Engine.

The GUI was set up to include the objects low level behavior (e.g., the button's `click`, the light's `turn on`, the time's `increase`). In the future we plan to attempt to connect directly to an existing model by extracting the object names and methods by using reflection on the model and matching them to the specification using synonyms [15].

Requirements were written to describe all aspects of the watch's behavior depicted in the statechart of the watch. A demonstration of the implemented watch is available in [16]. We also implemented another system — an ATM — to test the grammar. Since currently the grammar requires explicit repetition of objects and often needs the user to specify the behavior using a particular sentence, we would like to extend the grammar and also integrate some form of reference resolution.

5 Related Work

NLP has been used to aid software engineering in many ways. In [17] controlled natural language use case templates are translated into specifications in CSP

process algebra that may be used for validating the specified use cases. Use cases are specified in a table containing different steps of user action, system state and system response. Our approach allows inputting information of multiple steps in a single sentence more naturally and integrating different requirements. Our LSCs can also be validated or run (see smart play-out [5]) using model checkers.

There are approaches that generate executable object oriented code from natural language. The approach in [8] uses two-level-grammar (TLG) to first extract the objects and methods (a scheme that may be used for our initial phase as well) and it then extracts classes, hierarchies and methods. In [11], TLG is used to output UML class diagrams and Java code. The methods are described in natural language as a sequence of intra-object behaviors. (In contrast, our approach connects inter-object requirements and appears to be more fitting for reactive systems.)

Attempto Controlled English (ACE) [18,19] is a user-friendly language, based on first-order logic with rich English syntax, for translating NL into Prolog. It can by used for basic reasoning and queries but not for reactive systems.

Other works assist UML modeling and the design procedures with support tools that help extract the main objects and message sequences from natural language [20,21], thus making the transition from a NL specification to design less prone to errors. In [21] the scenarios in use cases are parsed to extract a tight representation of the classes and objects for the class diagram.

By and large, we have not encountered a translation that can create a reactive system from fragmented requirements.

6 Conclusions and Future Work

Creating complex reactive systems is not a simple task and neither is understanding natural language requirements. We have presented a method that allows one to translate controlled NL requirements into LSCs, with which a reactive system can be specified. The implementation of the system is thus a set of fragmented yet structured requirements — namely the LSCs, which are both natural and fully executable.

The current situation regarding the execution of LSCs is not without its limitations. For example, LSCs do not always result in a deterministic execution and the execution is also not always optimal. However, there is progress in many directions regarding the execution of LSCs; e.g., using an AI planning algorithm [6] can help the user choose one deterministic and complete path for system execution.

The ability to translate a controlled language into LSCs is a step in the right direction. The translation we suggest is tailored for the LSC language. However, it needs to be extended in order to support more of the rich language that humans normally use.

We would like to extend our scheme so that it becomes reasonably robust to errors, more user-friendly and so that it includes also dialogues that will help users understand how to write controlled requirements. We have yet to test the system on naive subjects.

We would like to add more abilities that will improve the natural language interface with the user. For example allowing specification using language "shortcuts", e.g. using the word *toggles* for changing between a few properties. We would like to add reference resolution, allowing the user to refer to objects previously mentioned as *it*. We would like to integrate NLP tools that resolve aliases for methods and properties, using dictionaries and common sense systems, this would allow the system to understand that different words refer to the same method or property, for example that *click* and *press* are the same method.

Another direction we would like to pursue is to include tools for transforming NL requirements to LSCs and back in a round-trip fashion, to enable easy project modification.

We believe the LSCs and the inter-object approach are naturally close to NL requirements. We hope the work presented here constitutes a small step towards improving the process of engineering reactive systems using natural language tools.

Acknowledgments

The authors would like to thank Shahar Maoz, Itai Segall and Dan Barak for helpful discussions and technical assistance. We would also like to thank the reviewers of an earlier version of this work for their helpful comments.

References

1. Harel, D.: Can Programming be Liberated, Period? Computer 41(1), 28–37 (2008)
2. Harel, D., Marelly, R.: Come, Let's Play: Scenario-Based Programming Using LSC's and the Play-Engine. Springer, Heidelberg (2003)
3. Damm, W., Harel, D.: LSCs: Breathing Life into Message Sequence Charts. Formal Methods in System Design 19(1), 45–80 (2001)
4. UML: Unified Modeling Language Superstructure, v2.1.1. Technical Report formal/2007-02-03, Object Management Group (2007)
5. Harel, D., Kugler, H., Marelly, R., Pnueli, A.: Smart Play-Out of Behavioral Requirements. In: Aagaard, M.D., O'Leary, J.W. (eds.) FMCAD 2002. LNCS, vol. 2517, pp. 378–398. Springer, Heidelberg (2002)
6. Harel, D., Segall, I.: Planned and Traversable Play-Out: A Flexible Method for Executing Scenario-Based Programs. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 485–499. Springer, Heidelberg (2007)
7. Mich, L.: NL-OOPS: From Natural Language to Object Oriented Requirements Using the Natural Language Processing System LOLITA. Natural Language Engineering 2(2), 161–187 (1996)
8. Bryant, B.: Object-Oriented Natural Language Requirements Specification. In: Proc. 23rd Australian Computer Science Conference (ACSC) (2000)
9. Segundo, L.M., Herrera, R.R., Herrera, K.Y.P.: UML Sequence Diagram Generator System from Use Case Description Using Natural Language. In: Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007), pp. 360–363 (2007)

10. Drazan, J., Mencl, V.: Improved Processing of Textual Use Cases: Deriving Behavior Specifications. In: van Leeuwen, J., Italiano, G.F., van der Hoek, W., Meinel, C., Sack, H., Plášil, F. (eds.) SOFSEM 2007. LNCS, vol. 4362, pp. 856–868. Springer, Heidelberg (2007)
11. Bryant, B.R., Lee, B.S.: Two-Level Grammar as an Object-Oriented Requirements Specification Language. In: Proc. 35th Annual Hawaii Int. Conf. on System Sciences (HICSS 2002), p. 280 (2002)
12. ITU: International Telecommunication Union: Recommendation Z.120: Message Sequence Chart (MSC). Technical report (1996)
13. Jurafsky, D., Martin, J.H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice-Hall, Englewood Cliffs (2008)
14. Harel, D.: On Visual Formalisms. Commun. ACM 31(5), 514–530 (1988)
15. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: Introduction to WordNet: An On-line Lexical Database (1993), http://wordnet.princeton.edu/
16. Requirements to LSCs: http://www.wisdom.weizmann.ac.il/~michalk/reqtolscs/
17. Cabral, G., Sampaio, A.: Formal Specification Generation from Requirement Documents. In: Brazilian Symposium on Formal Methods (SBMF) (2006)
18. Fuchs, N.E., Schwitter., R.: Attempto: Controlled natural language for requirements specifications. In: Proc. Seventh Intl. Logic Programming Symp. Workshop Logic Programming Environments (1995)
19. Fuchs, N.E., Schwitter, R.: Attempto Controlled English (ACE). In: Proc. 1st Int. Workshop on Controlled Language Applications, pp. 124–136 (1996)
20. Takahashi, M., Takahashi, S., Fujita, Y.: A proposal of adequate and efficient designing of UML documents for beginners. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part II. LNCS, vol. 4693, pp. 1331–1338. Springer, Heidelberg (2007)
21. Giganto, R.T.: A Three Level Algorithm for Generating Use Case Specifications. In: Proceedings of Software Innovation and Engineering New Zealand Workshop 2007 (SIENZ 2007) (2007)

Determining the Polarity and Source of Opinions Expressed in Political Debates

Alexandra Balahur, Zornitsa Kozareva, and Andrés Montoyo

University of Alicante, Departament of Software and Computing Systems,
Apartado de Correos 99, E-03080 Alicante, Spain
{abalahur,zkozareva,montoyo}@dlsi.ua.es

Abstract. In this paper we investigate different approaches we developed in order to classify opinion and discover opinion sources from text, using affect, opinion and attitude lexicon. We apply these approaches on the discussion topics contained in a corpus of American Congressional speech data. We propose three approaches to classifying opinion at the speech segment level, firstly using similarity measures to the affect, opinion and attitude lexicon, secondly dependency analysis and thirdly SVM machine learning. Further, we study the impact of taking into consideration the source of opinion and the consistency in the opinion expressed, and propose three methods to classify opinion at the speaker intervention level, showing improvements over the classification of individual text segments. Finally, we propose a method to identify the party the opinion belongs to, through the identification of specific affective and non-affective lexicon used in the argumentations. We present the results obtained when evaluating the different methods we developed, together with a discussion on the issues encountered and some possible solutions. We conclude that, even at a more general level, our approach performs better than trained classifiers on specific data.

Keywords: opinion mining, opinion source mining, LSA, political discourse.

1 Introduction

Most people, at the time of taking a decision, base their choice on a series of arguments, which are rational and/or emotional. For example, the factors influencing the purchase of a certain digital camera over another might take into consideration rational arguments, such as price, performance, but also emotional arguments, such as brand reputation and opinion expressed by other buyers of that camera. On the other hand, the decision to vote for a political party or a given candidate, while mostly based on the political beliefs one has, can be strongly influenced by whether or not, from the actions performed by the party or candidate, those political ideas are respected.

Recent years have marked the beginning and expansion of the social web, in which people freely express and respond to opinion on a whole variety of topics. Moreover, at the time of taking a decision, more and more people search for information and

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 468–480, 2009.

opinions expressed on the web on their matter of interest and base their final decision on the information found [1].

While the growing volume of opinion information available on the web allows for better and more informed decisions of the users, the quantity of data to be analyzed imposed the development of specialized Natural Language Processing systems that automatically extract, classify and summarize the opinions available on the web on different topics. Research in this field, of opinion mining (sentiment analysis), has addressed the problem of extracting and classifying opinion from different perspectives and at different levels, depending a series of factors. While determining the overall opinion on a movie is sufficient for taking the decision to watch it or not, when buying a product, people are interested in the individual opinions on the different product characteristics. Moreover, the approaches taken depend on the manner in which a user queries the data – whether it is a general formula such as "opinions on X" or a specific question "Why do people like X?". Determining points of view expressed along a dialogue on a topic can, additionally, involve determining the persons present and whether or not the opinion expressed is on the required topic or on a point previously made by another speaker.

In this paper we propose an approach to determining the opinion expressed on a political topic by different participants in American Congressional floor debates at a document level, using opinion and emotion lexicon. We envisage the development of a generic method to classify text fragments containing opinion and subsequently determine the target of those opinions. We show how this method can be used domain independently and can constitute a relatively high-precision basis for a system classifying opinion. We investigate the degree in which topic identification can help to better classify the opinion expressed in debates, taking into consideration the dependency between the opinion polarity and the target of the given opinion. We then reclassify documents taking into consideration the polarity of the opinion and the topic of the opinion, at a sentence level. We study the possibility to determine the source of an opinion on a topic given the orientation of the opinion and the arguments used for that opinion.

The following parts of the article are structured as follows: in Section 2, we describe the motivation for the present research and the contribution we bring with it. Section 3 presents related work and highlights the differences in approach from other authors. In Section 4, we describe the corpus used in our experiments. Further on, in Section 5, we present the approaches we have taken for solving the different problems we propose and the experiments performed on the data. We then evaluate the methods in Section 6. Finally, we conclude on our approach and depict the lines for future work in Section 7.

2 Motivation and Contribution

The main motivation of the present research is to test a general method for opinion classification that is based on similarity with affective and opinion lexicon. This method was previously used in the TAC 2008 Opinion Pilot task [2] and showed promising results. Whereas most systems concentrate on developing specialized methods for opinion classification within different scenarios, as we could observe

from the task proposed in the Opinion Pilot, a general opinion mining system must deal with many topics, ranging from products to brands, companies, public figures, news topics, etc.). Therefore, when pursuing the goal of classifying opinions, one must first of all have a base system that is able to detect negative and positive opinion. To this aim, we describe and employ a general opinion mining system.

Second of all, taking into consideration the corpus we have at hand, we study the manner in which opinion can be classified along dialogues, depending on the intervening speakers. We evaluate two methods to aggregate opinion scores in order to make a unique classification of opinions from a given speaker. While this type of classification was previously done in [3], their approach was dependent on the previous training on the same kind of data; our approach is data independent.

Last, but not least, we study the possibility to determine the source of the opinion expressed taking into consideration its polarity and the affect words used in expressing the arguments. Since the corpus is already annotated with the party the speaker belongs to, we perform this classification among the two parties represented – democrat and republican. While this type of classification was previously done in [4], the authors took into consideration the general vocabulary used, and not the attitude towards the topic per se and vocabulary related to it.

3 Related Work

Related work includes document-level sentiment analysis and opinion classification in political texts. Research in sentiment analysis at a document level, relevant research include [5], who first selects important sentences based on pre-specified part-of-speech patterns, then computes the semantic orientation of adjectives and subsequently sums up this orientation to determine the document polarity. Other related work include [6], who employs machine learning techniques to determine the overall sentiment in user reviews, [7] who propose classifying opinion on products based on individual opinions on product parts, [8], that studies the problems involved in machine learning approaches and the role of linguistic analysis for sentiment classification in customer feedback data. [9] research on the impact of dependency analysis in sentiment analysis and [10] analyze the role of linguistic knowledge sources in opinion mining.

On the other hand, research in sentiment analysis from political texts included classifying texts as conservative, liberal or libertarian [4] and placing texts on an ideological scale [11, 12]. Other authors proposed methods to represent opposing viewpoints of two parties in conflict [13]. The corpus used in our research was used in [3]. In the research presented, the authors investigate the possibility to determine support and opposition to the proposed legislation from the floor debates. They use subjectivity indicators to train a SVM classifier on part of the data and then employ a minimum-cut graph algorithm to determine the orientation of the opinions. They perform individual evaluations, first classifying individual speech segments and secondly classifying opinion depending on the speakers (assuming that a speaker will maintain the same opinion throughout the debate on a topic).

Our approach differs in many aspects to the one taken in previous work. First, we employ a general algorithm to classify the individual speech segments of the different

persons participating in the debates on each of the topics into positive and negative. We base our classification on similarity measures between the speech segments and the words pertaining to the categories of affect, opinion and attitude. In the first phase, we perform the classification without taking into consideration the target of the opinion expressed in the speech segments and without assuming any opinion consistence with respect to the speakers. The second classification, performed at speaker level, is done independently of the data. While we show the manner in which machine learning can be employed on our method to improve the results of the classifications, we discuss the implications this brings to the generality of the system.

4 Corpus

The corpus we use in our experiments is made up of congressional floor debates and was compiled by [3]. The corpus is available for download and research[1]. It is split into three sets: the development set, the training set and the test set. The first one contains 702 documents (one document corresponds to one speech segment) pertaining to the discussion of 5 distinct debate topics, the training set contains 5660 documents organized on 38 discussion topics and the test set contains 1759 documents belonging to 10 debates. The corpus contains three versions of these three sets, with the difference consisting in the removal of certain clues relating to the topic and the speaker referred to in the speech. The speech-segment file-naming convention, ###_@@@@@@_%%%%$$$_PMV is decoded as follows[2]:

1. ### is an index identifying the bill under discussion in the speech segment (hence, this number also identifies the 'debate' to which the speech segment belongs)
2. @@@@@@ is an index identifying the speaker
3. %%%% is the index for the page of the Congressional record on which the speech segment appears, i.e., a number from 0001 to 3268 corresponding to one of the original HTML pages that we downloaded from govtrack.us .
4. $$$ is an index indicating the position of the speech segment within its page of the Congressional record. Hence, for example, a file named 055_400144_1031004_DON.txt would be the 4th speech on the 1031st HTML page of the record.
5. 'P' is replaced by a party indicator, D or R (or X if no corresponding party could be found). As mentioned in the paper, we purposely *did not* use this information in our experiments.
6. 'M' is replaced by an indicator of whether the bill under discussion is mentioned directly in the speech segment, or whether it is only referenced by another speech segment on the same page. If the bill is directly mentioned in the current speech, the letter M appears in the file name; otherwise, the letter O appears.
7. 'V' is replaced by a vote indicator, Y or N, which serves as the ground-truth label for the speech.

[1] http://www.cs.cornell.edu/home/llee/data/convote.html
[2] Taken from the README file of the corpus.

In the present research, we only make use of the decoding at positions1, 2, 5 and 7. We also need to mention, that out of the three variants in which the data is available, we chose to use the first stage of the data. Therefore, we can use the references that are annotated within the individual speech segments, which is useful for the third approach of the first task.

5 Approaches and Experiments

5.1 Polarity Classification

The first task we performed was classifying the data on the basis of polarity of opinion. The first approach to this task was determining the polarity of the debate segments, taken individually. At this stage, we did not consider the information regarding the speaker. In order to perform this, we used the similarity measure given by Ted Pedersen's Statistics Package[3] with affect, opinion and attitude lexicon. The affect lexicon consisted of three different sources: WordNet Affect [14] - (with 6 categories of emotion – joy, surprise, anger, fear, sadness, disgust), the ISEAR corpus [15] – that contains the 7 categories of emotion – anger, disgust, fear, guilt, joy, sadness and shame, from which stopwords are eliminated) and the emotion triggers database [16]- which contains terms related to human needs and motivations annotated with the 6 emotion categories of WordNet Affect. The opinion lexicon contained words expressing positive and negative values (such as "good", "bad", "great", "impressive" etc.) obtained from the opinion mining corpus in [17] and to which their corresponding nouns, verbs and adverbs were added using Roget's Thesaurus. Finally, the attitude corpus contains the categories of "accept", "approval", "confidence", "importance", "competence", "correctness", "justice", "power", "support", "truth" and "trust", with their corresponding antonymic categories – "criticism", "opposition", "uncertainty", "doubt", "unimportance", "incompetence", "injustice", "objection", "refusal" , "incorrectness".

After obtaining the similarity scores, we summed up the scores pertaining to positive categories of emotion, opinion and attitude and the negative categories, respectively. Therefore, the general positive score was computed as sum of the individual similarity scores for the categories of "joy" and "surprise" from the affect category, the "positive values" of the opinion lexicon and the "accept", "approval", "competence", "confidence", "correctness", "justice", "power", "support", "trust" and "truth". On the other hand, the general negative score was computed as sum of the "anger", "fear", "sadness", "shame" from the affect categories, the "negative values" of the opinion lexicon and the "criticism", "opposition", "uncertainty", "doubt", "unimportance", "incompetence", "injustice", "objection", "refusal" and "incorrectness" categories of the attitude lexicon. The first classification between negative and positive speaker segments was done comparing these two resulting scores and selecting the higher of the two as final value for polarity. We evaluated the approach on the development, training and test sets (Classification 1).

On the other hand, we employed the scores obtained for each of the emotion, opinion and attitude categories, as well as the combined scores used for classifying in

[3] http://www.d.umn.edu/~tpederse/text-similarity.html

the first step for the training of an SVM classifier, using the development and training sets. We then tested the approach on the test set solely. Due to the fact that in the affect category there are more negative emotions (4) than positive ones (only 2), we chose for classification only the two strongest emotions according to the similarity scores found in the first approach. Those two categories were "fear" and ``anger". The results are presented under Classification 2.

Further, we parsed the speaker segments using Minipar[4], in order to determine possible dependency paths between words pertaining to our affect, opinion or attitude lexicon and the topic under discussion or mentioning of another speaker. Our guess was that many of the speech segments that had been classified as negative, although the ground-truth annotation had them assigned a positive value, contained a negative opinion, but not on the topic under discussion, but on the opinion that was expressed by one of the anterior speakers. Therefore, the goal of our approach was to see whether the false negatives were due to the classification method or due to the fact that the object on which the opinion was given was not the one we had in mind when classifying. In order to verify our hypothesis, we extracted from the files in which the opinion words from the files with similarity higher than 0 appeared and sought dependency relations between those words and the mention of a speaker (based on the number assigned) or the words describing the topic discussed – marked in files in which this names appear, the words "bill","legislation", "amendment" and "measure". Affect, opinion or attitude words to which no relation was found to the mentioned topic or a speaker were discarded. In this approach, we did not use anaphora resolution, although, theoretically, it could help improve the results obtained. It would be interesting to study the effect of applying anaphora resolution on this task.

The results of the classification are summed up under Classification 3. Figure 1 presents an example of the dependency analysis for one of the sentences in which an attitude word was identified. It can be see that the word "support" – pertaining to the attitude category, has a dependency path towards the name of the bill under discussion – "h.r. 3283". Figure 2 shows a schematic overview of the first approach, with the resources, tools and methods employed therein.

```
> (
E0   (()    fin C *      )
1    (i     ~ N   2      s      (gov rise))
2    (rise  ~ V   E0     i      (gov fin))
E2   (()    I N   2      subj   (gov rise)  (antecedent 1))
3    (in    ~ Prep       2      mod    (gov rise))
4    (strong      ~ A     5      mod    (gov support))
5    (support     ~ N     3      pcomp-n      (gov in))
6    (of    ~ Prep       5      mod    (gov support))
7    (h.r   ~ N   6      pcomp-n      (gov of))
8    (3283  ~ N   7      num    (gov h.r))
9    (.     ~ U   *      punc)
)
```

Fig. 1. Minipar output for a sentence in topic 421 on act "h.r. 3283"

[4] http://www.cs.ualberta.ca/~lindek/minipar.htm

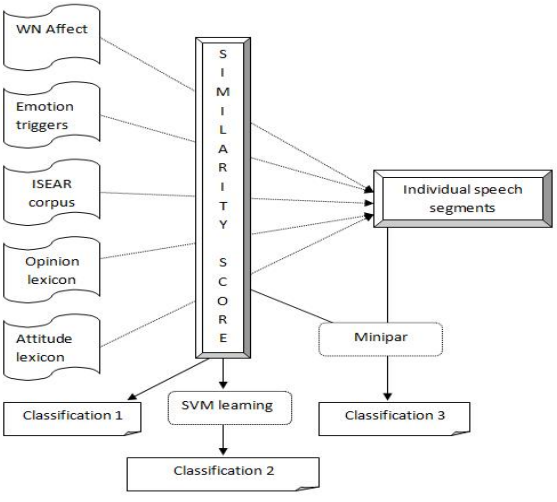


Fig. 2. Resources and tools scheme for the first approach

The second approach on the data was aggregating the individual speaker segments on the same debate topic into single documents we denote as "speaker interventions". We then performed, on the one hand, a classification of these interventions using the sum-up of the scores obtained in the individual speech segments and, on the other hand, based on the highest score in each of the categories. Thirdly, we employed SVM to classify the speaker interventions using the aggregated scores from the individual text segments and the highest scores of the individual speaker segments, respectively. The training was performed on the development and training sets and the classifications (Classification 4 and Classification 5, respectively) were evaluated on the test set.

5.2 Source Classification

The second task we performed was classifying the source of opinions expressed. In the following experiments, we used the fact that the corpus contained the name of the party the speaker belonged to coded in the filenames. The goal was to see whether or not we are able to determine the party a speaker belongs to, by taking into consideration the words used to express opinion on a given subject, the arguments (the words used within the argumentation) and the attitude on the subject in question. Our hypothesis was that, for example, parties in favor of a certain piece of legislation will use both a set of words that are positively speaking on the matter, as well as a set of arguments related to the topic that are highlighting the positive side. In order to perform this task, we used a clustering on the words pertaining to the affect lexicon, opinion lexicon and attitude lexicon, as well as the most frequent words appearing in the individual speaker segments of persons belonging to each of the two parties – Democrat and Republican. As mentioned by [4], there are two problems that arise when intending to classify pertainance to a political party in a topic debate. The first one is the fact that when talking on a certain topic, all or most persons participating in

the debate will use the same vocabulary. The second issue is that a certain attitude on a topic cannot reliably predict the attitude on another topic. Related to the first problem, we verify whether or not attitude towards a topic can be discriminated on the bases of the arguments given in support or against that topic, together with the affect, opinion and attitude lexicon used together with the arguments used. As far as the second issue is concerned, we do not aim to classify depending on topic, but rather predict, on the basis of the arguments and affective words used, the party the speaker belongs to.

6 Evaluation

We evaluated the approaches described in terms of precision and recall. In order to exemplify the manner in which we calculated these scores, we present confusion matrixes for all individual speech segments pertaining to the 5 topics in the development set. The "yes"/"no" category includes the individual speech segments that whose ground truth was "yes"/"no". The "positive"/"negative" category includes the individual speech segments that the system classified as "positive"/"negative". The details related to the quantity of data in each topic and within each of the "yes"/"no", "positive"/"negative" categories are presented only for the development set, due to large number of topics and documents contained within the training and test sets. For the two latter, we present the overall results.

Table 1. Confusion matrices and evaluation results for the topics in the development set

Topic number	Polarity	yes	no	P	R	A
199	positive	30	7	0.81	0.57	0.62
	negative	22	16	0.43	0.70	
553	positive	28	3	0.95	0.76	0.81
	negative	15	29	0.65	0.92	
421	positive	28	3	0.90	0.65	0.76
	negative	15	29	0.66	0.90	
493	positive	44	3	0.93	0.62	0.77
	negative	26	58	0.78	0.95	
052	positive	35	2	0.94	0.59	0.70
	negative	24	26	0.52	0.92	

The following table shows the confusion matrix for the source classification, trained on the development set and tested using a sample of 100 documents from the test set, equally distributed among the Democrat and Republican Party.

Table 2. Results for source classification – confusion matrix test set – 100 documents

	D	R
Classified D	29	21
Classified R	10	40

We computed precision over positive classification as the number of individual text segments that our system classified as positive and that had the ground truth "yes" divided by the number of individual text segments that our system classified as positive and that had the ground truth "yes" summed with the number of individual text segments that our system classified as positive and that had the ground truth "no". We computed precision over negative classification as the number of individual text segments that our system classified as negative and that had the ground truth "no" divided by the number of individual text segments that our system classified as negative and that had the ground truth "no" summed with the number of individual text segments that our system classified as negative and that had the ground truth "yes".

$P_{pos}(199) = 30/37 = 0.81$; $P_{pos}(553) = 0.95$; $P_{pos}(421) = 0.90$; $P_{pos}(493)=0.93$; $P_{pos}(052)=0.94$; $P_{neg}(199) = 16/38 = 0.43$; $P_{neg}(553) = 0.65$; $P_{neg}(421) = 0.66$; $P_{neg}(493) = 0.78$; $P_{neg}(052)=0.52$

We computed recall over positive classification as the number of individual text segments that our system classified as positive and that had the ground truth "yes" divided by the number of individual text segments that our system classified as positive and that had the ground truth "yes" summed with the number of individual text segments that our system classified as negative and that had the ground truth "yes". We computed recall over negative classification as the number of individual text segments that our system classified as negative and that had the ground truth "no" divided by the number of individual text segments that our system classified as negative and that had the ground truth "no" summed with the number of individual text segments that our system classified as positive and that had the ground truth "no".

$R_{pos}(199) = 30/52 = 0.57$; $R_{pos}(553) = 0.76$; $R_{pos}(421) = 0.65$; $R_{pos}(493) = 0.62$; $R_{pos}(052) = 0.59$; $R_{neg}(199) = 16/23 = 0.70$; $R_{neg}(553) = 0.92$; $R_{neg}(421) = 0.90$; $R_{neg}(493) = 0.95$; $R_{neg}(052)= 0.92$

We compute the accuracy score as the sum of the number of correct positive classifications and the number of correct negative classifications, divided by the total number of documents on a topic.

$A(199) = 46/75 = 0.62$; $A(553) = 0.81$; $A(421) = 0.76$; $A(493) = 0.77$; $A(052) = 0.70$

The overall precision over positive classification is computed as average of all the precision scores of all the positive classifications. The precision over negative classifications is computed in the same manner.

The overall recall over positive classification is computed as average of all the recall scores of all the positive classifications. The recall scores over negative classifications is computed in the same manner.

The overall accuracy is computed as average of all the accuracy scores of all the topics.

Table 3. Classification 1 (individual speaker segments) based on sums of similarity scores to affect, opinion and attitude lexicon categories

	P_{pos}	P_{neg}	R_{pos}	R_{neg}	Accuracy
Development set	0.71	0.6	0.63	0.87	0.73
Training set	0.69	0.61	0.63	0.86	0.72
Test set	0.70	0.6	0.62	0.87	0.73

In this approach, we aimed at classifying the individual speaker segments according to their polarity.

Table 4. Classification 2 (individual speaker segments) using dependency parsing

	P_{pos}	P_{neg}	R_{pos}	R_{neg}	Accuracy
Development set	0.72	0.69	0.69	0.85	0.77
Training set	0.71	0.68	0.68	0.85	0.76
Test set	0.70	0.67	0.68	0.84	0.76

In this approach, we aimed at classifying the individual speaker segments according to their polarity, using dependency parsing to determine the target of the opinions expressed.

Table 5. Classification 3 (individual speaker segments) based on SVM

	P_{pos}	P_{neg}	R_{pos}	R_{neg}	Accuracy
Test set	0. 75	0.63	0.66	0.88	0.78

This third classification of the individual text segments according to polarity was done using the scores obtained for each of the emotion/opinion/attitude categories, using the SVM machine learning algorithm.

Table 6. Classification 4 (speaker interventions) based on sum-up of scores

	Accuracy
Development set	0.68
Training set	0.66
Test set	0.67

This polarity classification was done for each of the speakers, by adding the polarity scores obtained for each of the individual speech segments pertaining to the speaker's intervention.

Table 7. Classification 5 (individual speaker segments) based on highest score

	Accuracy
Development set	0.56
Training set	0.55
Test set	0.58

Finally, another classification was done for each of the speaker segments, on the basis of the polarity given by highest score obtained for the speaker's individual speech segments.

As it can be noticed from the scores obtained, the system has a rather balanced behavior on all topics in question. This is a positive fact, because the composition of the documents in the corpus was different, with as much as double number of positive speaker interventions than negative ones. We notice the tendency of the system to overly classify interventions as negative, a fact which is rectified by SVM learning, where better results are obtained. Dependency parsing was also found to help on the classification, improving the system's performance noticeably. SVM performed better than the initial method used for classification, but overall, the performance was lower than that obtained when performing dependency analysis. We can also notice that when uniting speaker segments, the classification is done with better results. We believe this to be due to the fact that in the context of the larger text segments, more of the emotion categories have assigned a value above 0, and therefore more scores are important to the final result. Another factor to take into consideration when we analyze the results is the fact that speaker interventions were not equal as far as text length. From the post evaluation analysis, we found that the system performs better in classifying longer texts, to which more categories of emotion have similarity scores above 0. From our experiments, it was seen that the categories that had the most importance at the time of classifying we those of "fear" and "joy" from the affect list – but given not by the lexicon in WordNet Affect, but from the categories found in the emotion triggers. As far as opinion source is concerned, the results shown in Table 6 demonstrate that a classification with 6.9 accuracy can be easily done using the affect and argument specific lexicon.

7 Conclusions and Future Work

In this article we have proposed different methods to classify opinion from texts using affect, opinion and attitude lexicon. We applied the proposed approaches to Congressional debates. We presented three methods to classify individual speech segments, the first based on a simple sum of similarity scores to the three lexicons used. We showed that applying dependency parsing and discovering the target of the opinion improved the initial classification, not only in the sense of the scores obtained, but also helped to balance between the results obtained for the positive and negative categories, respectively. Further, we showed that using SVM machine learning, we can improve the classification of the opinions, but that SVM performs worse than dependency analysis in the sense that it does not help improve the misclassification of positive opinions as negative ones. We believe this is due to the

fact that the data set is not balanced as far as number of positive versus negative interventions is concerned. We showed that classification is dependent on the text length and that speaker interventions are better classified than individual speech segments alone, based only on the aggregated score obtained for each of the individual segment files. As far as opinion source is concerned, we showed that using affect, opinion and attitude lexicon in relation to the arguments given within the speech segments can help classify the source of opinion with a 69% accuracy. Our experiments also proved that the resources we created for emotion and opinion classification are relevant and important to the task of polarity classification. Future work includes testing alternative resources and tools for sentiment mining and studying the impact of anaphora on the task of opinion classification in different text genres. We believe, on the one hand, that in many cases our approach fails because the target of the opinions expressed are found in the form of anaphoric references. On the other hand, testing alternative resources and methods and comparing the results can give us a measure of the degree in which each of the components we use contributes to the success or failure of the system.

References

1. Pang, B., Lee, L.: Opinion Mining and Sentiment Analysis. Foundations and Trends in Information Retrieval 2(1-2), 1–135 (2008)
2. Balahur, A., Lloret, E., Ferrández, O., Montoyo, A., Palomar, M., Muñoz, R.: The DLSIUAES Team's Participation in the TAC 2008 Tracks. In: Proceedings of the Text Analysis Conference 2008 Workshop, Washington, USA (2008)
3. Thomas, M., Pang, B., Lee, L.: Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In: Proceedings of EMNLP 2006 (2006)
4. Mullen, T., Malouf, R.: A preliminary investigation into sentiment analysis of informal political discourse. In: AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW), pp. 159–162 (2006)
5. Turney, P., Mullen, T., Malouf, R.: A preliminary investigation into sentiment analysis of informal political discourse. In: Proceedings of the AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW 2002) (2002)
6. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 79–86 (2002)
7. Dave, K., Lawrence, S., Pennock, D.: Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In: Proceedings of WWW 2003 (2003)
8. Gamon, M.: Sentiment classification on customer feedback data: Noisy data, large feature vectors, and the role of linguistic analysis. In: Proceedings of the International Conference on Computational Linguistics (COLING) (2004)
9. Matsumoto, S., Takamura, H., Okumura, M.: Sentiment classification using word sub-sequences and dependency sub-trees. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS, vol. 3518, pp. 301–311. Springer, Heidelberg (2005)
10. Ng, V., Dasgupta, S., Arifin, S.M.N.: Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In: Proceedings of the COLING/ACL Main Conference Poster Sessions, July 2006, pp. 611–618. Association for Computational Linguistics, Sydney (2006)

11. Laver, M., Benoit, K., Garry, J.: Extracting policy positions from political texts using wor words as data. American Political Science Review 97, 311–331 (2003)
12. Martin, L.W., Vanberg, G.: A robust transformation procedure for interpreting political text. Political Analysis 16, 93–100 (2008)
13. Lin, W.-H., Wilson, T., Wiebe, J., Hauptmann, A.: Which side are you on? Identifying perspectives at the document and sentence levels. In: Lin, et al. (eds.) Proceedings of the Conference on Natural Language Learning (CoNLL 2006) (2006)
14. Strapparava, C., Valitutti, A.: WordNet-Affect: an affective extension of WordNet. In: Proceedings ofthe 4th International Conference on Language Resources and Evaluation (LREC 2004), Lisbon, pp. 1083–1086 (May 2004)
15. Scherer, K., Wallbott, H.G.: The ISEAR Questionnaire and Codebook (1997)
16. Balahur, A., Montoyo, A.: An Incremental Multilingual Approach to Forming a Culture Dependent Emotion Triggers Database. In: Proceedings of the 8th International Conference on Terminology and Knowledge Engineering (TKE 2008), Copenhagen (2008)
17. Balahur, A., Montoyo, A.: Multilingual Feature–driven Opinion Mining and Summarization from Customer Reviews. In: Kapetanios, E., Sugumaran, V., Spiliopoulou, M. (eds.) NLDB 2008. LNCS, vol. 5039, pp. 345–346. Springer, Heidelberg (2008)

Query Translation and Expansion for Searching Normal and OCR-Degraded Arabic Text

Tarek Elghazaly and Aly Fahmy

Faculty of Computers and Information, Cairo University, Giza, Egypt
{t.elghazaly,a.fahmy}@fci-cu.edu.eg

Abstract. This paper provides a novel model for English/Arabic Query Translation to search Arabic text, and then expands the Arabic query to handle Arabic OCR-Degraded Text. This includes detection and translation of word collocations, translating single words, transliterating names, and disambiguating translation and transliteration through different approaches. It also expands the query with the expected OCR-Errors that are generated from the Arabic OCR-Errors simulation model which proposed inside the paper. The query translation and expansion model has been supported by different libraries proposed in the paper like a Word Collocations Dictionary, Single Words Dictionaries, a Modern Arabic corpus, and other tools. The model gives high accuracy in translating the Queries from English to Arabic solving the translation and transliteration ambiguities and with orthographic query expansion; it gives high degree of accuracy in handling OCR errors.

Keywords: Query Translation, Orthographic Query Expansion, Cross Language Information Retrieval, Arabic OCR-Degraded Text, Arabic Corpus.

1 Introduction

The importance of Cross Language Information Retrieval (CLIR) appears clearly when we consider a case like the Library of Congress [1] which has more than 134 million items and approximately half of the library's book and serial collections are in 460 languages other than English. When people like to retrieve the whole set of documents that represent some interest, they have to repeat search process in each language. Furthermore, as a big number of books and documents are available only in print especially the Arabic ones, they are not 'full text' searchable and they need applying the Arabic OCR process whose accuracy is far from perfect [2]. The goal of this paper is to provide a solid English/Arabic query translation and expansion model to search both normal and OCR-Degraded Arabic Text.

The outline of this paper is as follows: The previous work is reviewed in Section 2. The proposed work is presented in the next sections. Arabic words formalization, normalization and stemming are presented in Section 3. Corpus and Dictionaries are presented in Section 4 and 5. In Section 6 & 7 the work done for CLIR through Query Translation and expansion respectively is detailed, followed by the experimental results and the conclusions in Sections 8 & 9.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 481–497, 2009.

2 Previous Work

2.1 Arabic Morphological Analysis for Information Retrieval (IR)

Several researches have been done to check the effect of light stemming & root based stemming on IR like in [3] [4] and [5]. Al-Jilayl and Frieder concluded in [6] that light stemmer performs than root based stemmer (using enhanced version of Khoja root based stemmer [7]). The effect of either stemming techniques on Information Retrieval was better than no stemming at all. The same result has also been concluded by Larkey et al. in [8].

2.2 Arabic Corpus

As per Hunston in [9], the construction and use of text corpora is continuing to increase [9]. Several research efforts has been done in this field like Kharashi & Evens in [3], Hmeidi et al in [10], Goweder et al in [11] and Darwish et al. in [12].

2.3 CLIR

In CLIR, either documents or queries are translated. There are three main approaches to CLIR: Machine Translation (MT), Comparable or Parallel Corpus, and Machine Readable Dictionaries.

MT systems seek to translate queries from one human language to another by using context. Disambiguation in MT systems is based on syntactic analysis. Usually, user queries are a sequence of words without proper syntactic structure [13]. Therefore, the performance of current MT systems in general language translations make MT less than satisfactory for CLIR [14].

In corpus-based methods, queries are translated on the basis of the terms that are extracted from parallel or comparable document collections. Dunning and Davis used a Spanish-English parallel corpus and evolutionary programming for query translation [15]. Landauer and Littman [16] introduced a method called Cross Language- Latent Semantic Indexing (CL-LSI), and requires a parallel corpus. Unlike parallel collection, comparable collections are aligned based on a similar theme [17].

Dictionary-based methods perform query translation by looking up terms on a bilingual dictionary and building a target language query by adding some or all of the translations. This technique can be considered the most practical method [18]. Ballesteros and Croft [19] developed several methods using MRDs for Spanish-English CLIR and then improved the effectiveness by many ways including resolving the ambiguity [20],[21],[22]. Pirkola [13] studied the effects of the query structure and setups in the dictionary-based method. Mohammed Aljlay and Ophir Frieder investigated for the Arabic-English CLIR [23] (The opposite direction of this paper). They investigated MT and MRD to Arabic-English CLIR using queries from TREC [24]. They concluded that Query Translation for Arabic/English CLIR through Machine-readable dictionaries is cost effective as compared to the other methods such as parallel corpus, Latent Semantic Indexing (LSI), and MT. Ahmad Hasnah and Martha Evens concluded also in [25] that most comprehensive work is to work with the bilingual MRD with solving the problems of terms translation ambiguity.

2.4 CLIR for Arabic OCR-Degraded Text

For handling OCR-Degraded text in CLIR, Darwish K. investigated in [26] the different methods for query term replacement and he found that Word Term Frequency/Document Frequency (WTF/DF) was the best evaluated approach of the evaluated ones. He proved an approach of producing possible replacements for query terms that could have been generated by OCR proved to be a useful technique for improving retrieval of OCR-degraded text.

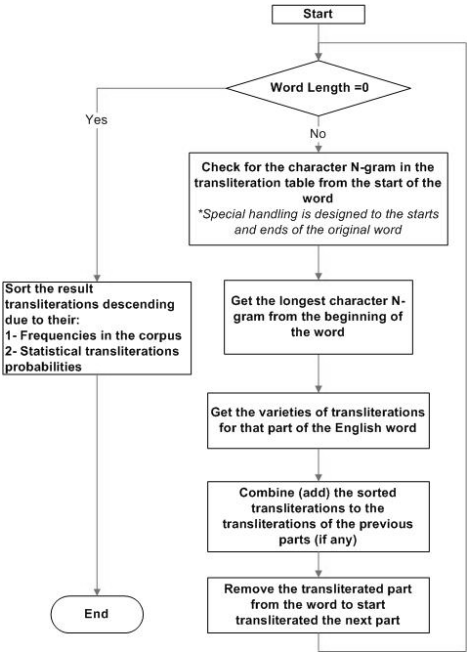


Fig. 1. The Transliteration Model

2.5 Previous Work: Comments and Limitations

As mentioned in [24] and [25], the most cost effective and practical method for CLIR is using MRDs. Darwish K. work in [26] tried in this direction especially in the English/Arabic CLIR supporting also OCR-Degraded Text. But however, it suffered from some limitations. From the Query Translation perspective, it did not provide a solution for the named entities, expressions, and the word collocations in general. For the OCR-Degraded Text handling, it concentrated on the correction of character n-grams (up to 7-gram) but it does not take into consideration neither the higher n-grams nor the position of this character n-gram inside the words. In this paper, we try to find a solid solution for the English/Arabic CLIR for both the normal and the OCR-Degraded Arabic Text overcoming the mentioned limitations.

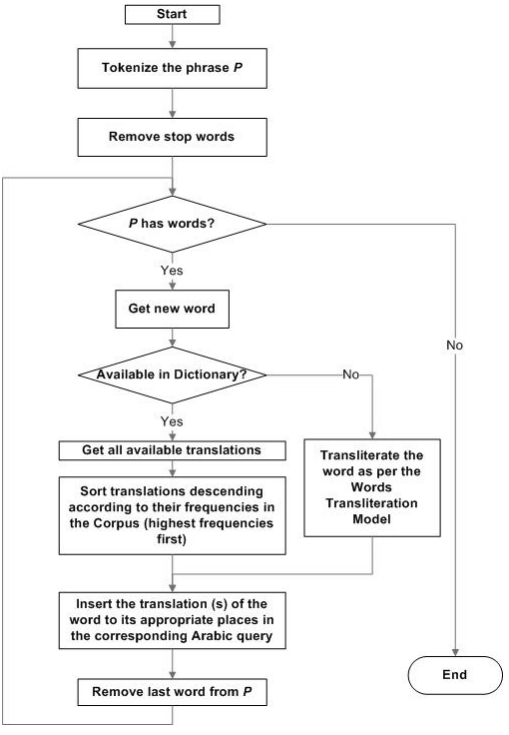


Fig. 2. Single words translation Model

3 The Proposed Light Stemmer for Arabic Information Retrieval

As per the previous work mentioned in section 2, light stemming can be considered as the most effective approach for improving Arabic IR rather than aggressive stemming or the root extraction. In this paper, we propose a light stemmer that normalize then lightly stem Arabic words.

The proposed stemmer works on three steps. First, it normalizes the Arabic word characters that are written differently due to the different writing ways or due to the common writing mistakes. This is to unify 'ي' (Yaa) and 'ى' (Alef Maqsoura) to 'ي', and to unify 'أ' (Alef with Hamza on top), 'إ' ((Alef with Hamza on bottom) ,'ء' (Hamza) ,'آ' (Alef maad) and 'ا' (Alef) to 'ا', and also for 'ـة' (Taa marbouta) ,'ـه' (Ha)) to 'ـه'. The 2nd step is to produce the stems as prefix stripped, suffix stripped, and both prefix and suffix stripped with always considering the longest combinations to form prefixes and same for suffixes. The 3rd step is to discard the produced stems that are not available in the Arabic dictionary. The available prefixes and suffixes are available in [27]. The advantage of the proposed stemmer is that it provides only the stem if it is available in the Arabic dictionary.

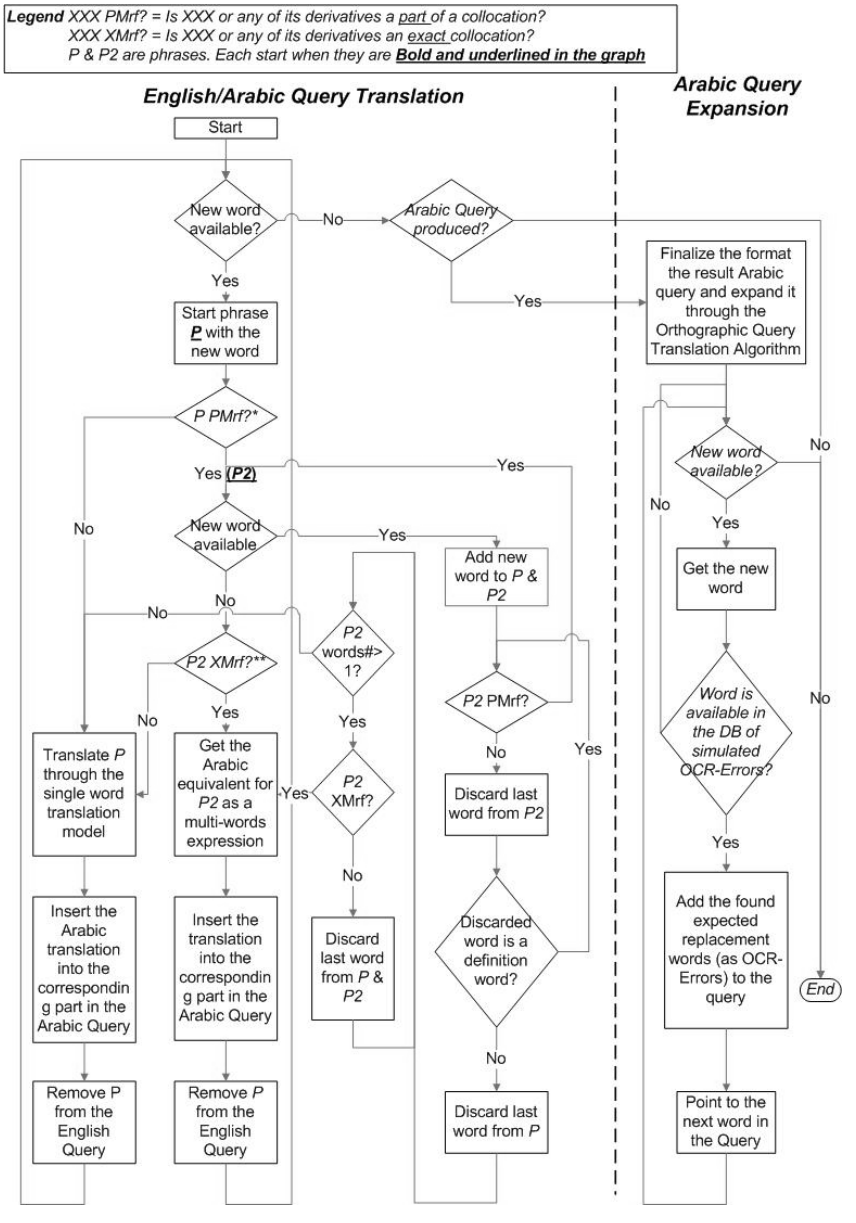


Fig. 3. The Query Translation and Expansion Model

4 The Proposed Modern Arabic Corpus

The proposed Corpus ia a Modern Arabic Corpus that would help in the study of Modern Standard Arabic in general and to use its statistics to solve the Query

Translation replacement ambiguity. Moheet portal [28] has been chosen as the main data source as it gets its news articles from 200 sources from different countries, perspectives, and fields. It takes also Arabic articles through translation into Arabic from non-Arabic sources.

Moheet Portal has been crawled for 336 continuous hours. Articles are parsed to extract the plain Arabic words excluding all Latin characters, punctuations, or diacritics. Then the result Text database was analyzed to get the unique exact words and the unique exact stems (after normalization and stemming). The overall figures of the Corpus are illustrated in Table 1.

Table 1. Statistics of the established Modern Arabic Corpus

Category	*Figures*
Corpus Textual Size	6.8 GB
Number of Textual Documents	98,000
Total No. of extracted Arabic words	46,603,112
No. of unique words (exact)	338,335
No of unique words (stemmed)	155,561

5 Establishing a Word Collocations Dictionary and Evaluating Single Words Dictionaries

5.1 Establishing the Proposed Word Collocation Dictionary

To establish this dictionary, we considered WordNet as the source of English Word Collocations as it is considered as most important resource available to researchers in computational linguistics [29]. Table2 describes some of the WordNet statistics.

Table 2. WordNet Statistics

POS	*Unique Strings*	*Synsets*	*POS*
Noun	117097	81426	145104
Verb	11488	13650	24890
Adjective	22141	18877	31302
Adverb	4601	3644	5720
Total	155327	117597	207016

It has been parsed to extract the collocations (expressions, proper noun, and named entities, with multi-words). The collocations have been translated then reviewed manually [30]. The result Word Collocations Dictionary has the figures in Table 3.

Table 3. Detailed figures for the Collocations Dictionary

	2-gram	*3-gram*	*4-gram*	*5-gram*	*6-gram*	*7-gram*	*8-gram*	*Total*
Noun	51,205	7030	1246	268	73	25	29	59,876
Verb	2,404	297	87	12	5	1	0	2,806
Adv.	437	294	85	12	1	0	0	829
Adj.	426	145	0	2	0	2	0	575
Total	54,472	7,766	1,418	294	79	28	29	64,086

5.2 Single Words Dictionaries

Dictionary 1. The main goal of producing this dictionary is to provide a modern dictionary based on a data that are originally from an English/Arabic source and is slimmed to cover the practical Arabic meanings to the English words. The raw data for this dictionary is an English/Arabic dictionary data as one of the outputs of the Arabeyes project [31], [32]. The output Dictionary DB has 87,423 English words and every English word has from one to two Arabic translations.

Dictionary 2. The main goal of producing this dictionary is to provide a dictionary based on a data that are originally extracted from an Arabic/English source. The raw data for this dictionary is an Arabic/English Dictionary from Computing Research Laboratory (CRL), New Mexico State University [33]. The output DB has unique 30,389 English words and every English word has from 1 to 248 Arabic translations. The average number of Arabic translations for every English word is 5.

Dictionary 3. The main goal of producing this dictionary is to provide a big one based on a data that are originally extracted from an English/Arabic source and is as big as it covers almost the whole set of available English words. The main English words are available from CRL dictionary and WordNet [34]. Every word has been translated through a free industry-known online dictionary [35] and all translations have been collected. The output DB has around 52,000 English words and Every English word has from one to 205 English translations. Each Arabic word has about 8 corresponding English translations in average.

6 The Proposed Model for CLIR through Query Translation

In this section we will introduce our proposed model for CLIR through Query Translation. This includes the models for Names Transliteration, Single Words Translation, collocations Translation, solving the ambiguities, and Query Translation.

6.1 Transliteration Model

The model's main idea is to check the longest n-gram character section in the start of the word to be translated directly from the n-gram transliteration table, doing the same

for the end, of the words, then the medium sections of the word. As there are often many transliteration probabilities for the same section, all these probabilities are taken into consideration due to the frequency of the corpus and the probability of that section with respect to the transliteration table. Table 4 illustrates the transliteration table used [36]. The model is illustrated in Fig 1.

Table 4. Transliteration Table

N-gram	T	P	T	P	T	P	T	P
A	ا	0.6	0	0.2	ي	0.1	ع	0.1
a (End)	ه	0.6	ا	0.4				
ai	ي	0.5	أي	0.5				
alk	وك	0.9	الك	0.1				
au	ا	0.4	او	0.4	و	0.2		
B	ب	1						
bb	ب	1						
a (Start)	ا	0.9	ع	0.1				
au (Start)	ا	0.8	او	0.2				
e (Start)	ا	0.4	0	0.1	ي	0.3	يا	0.3
i (Start)	ا	0.7	أي	0.2	ع	0.1		
mc (Start)	ماك	0.9	مك	0.1				
o (Start)	ا	0.3	او	0.7				
u (Start)	ا	0.8	او	0.2				
wr (Start)	ر	1						
C	ك	0.9	شن	0.1				
cc	ك	1						
ce	س	0.8	سي	0.2				
ch	شن	0.8	ش	0.2				
ci	س	0.2	سي	0.8				
cy	س	0.2	سي	0.8				
ck	ك	1						
D	د	1	0	0				
dd	د	1						
E	0	0.6	ا	0.1	ي	0.3		
ea	ي	0.9	يا	0.1				
e (End)	0	0.9	ه	0.1				
ee	ي	1						
ey	أي	0.8	ي	0.2				
F	ف	1						
ff	ف	1						
ph	ف	1						
G	غ	0.5	ج	0.4	ق	0.1		
ge	ج	0.8	غ	0.2				

N-gram	T	P	T	P	T	P
ie	ي	0.7	أي	0.3		
j	ج	0.9	ي	0.1		
k	ك	1				
kk	ك	1				
kh	خ	1				
l	ل	1				
ll	ل	0.8	لل	0.1	ي	0.1
m	م	1				
mm	م	1				
n	ن	1				
nn	ن	1				
o	و	0.7	ا	0.1	0	0.2
ois	وا	0.8	وين	0.1	بوين	0.1
oo	و	1				
ou	و	0.6	او	0.4		
ough	او	0.4	وف	0.2	و	0.4
ough (End	ه	0.8	وف	0.2		
p	ب	1				
pp	ب	1				
q	ك	0.5	ق	0.5		
qu	كو	0.6	ك	0.3	ق	0.1
r	ر	1				
rr	ر	1				
s	س	0.6	ز	0.2	ص	0.2
sch	ش	0.8	سشن	0.2		
s (End)	س	0.6	ز	0.4		
sh	ش	1				
ss	س	0.8	ص	0.2		
t	ت	0.7	ط	0.3		
th	ث	0.3	ث	0.4	ذ	0.3
tio	ش	0.8	شيو	0.2		
tt	ت	0.9	ط	0.1		
u	و	0.8	0	0.2		
ue (End)	0	0.8	و	0.2		

6.2 Single Words Translation Model

The main idea of the proposed model is to get the input as phrase which is not a collocation or a multi-words expression, tokenize that phrase, remove stop words, and get the Arabic equivalent for each word. If the English word does not have an Arabic equivalent word, then the word will be transliterated through the transliteration mode. The model is illustrated in Fig 2.

6.3 The Model for Checking Collocations Parts

The main idea of this model is to check if the current part of the search sentence is a part of collocation. Continuous checking for that purpose will lead to get the longest

collocation in the search sentence. For example both “United Nations” and “United Nations Children’s Fund” are collocations. This continuous checking will succeed in finding the correct collocation which is the second one (the longest). The main benefit of considering the longest collocation is getting the most accurate translation as described in the next section in details.

The model checks the entered query words in the Word Collocations Dictionary either exact or stemmed (through the WordNet rues) taking into consideration that only base forms of words even those comprising collocations, are stored in WordNet [37].

6.4 Solving Translation / Transliteration Ambiguity

Every single English word –that are available in dictionaries- has one or more Arabic Equivalents up to 248 ones (as in Dictionary 2). Also, a word that is not available in dictionaries and has to be transliterated has many probabilities as every character has one or many probability. A word Like Lincoln will have 18 Arabic transliterations. As the query may have many words, the ambiguity will be very high. In this section we propose several methodologies for solving the ambiguity of translation and transliteration trough collocations dictionary, using corpus, and using transliteration probabilities.

Word Collocations Dictionary. If the query has the phrase "United Nations Children's Fund", the direct translation will be for every words respectively (20, 6, 14, 19). This means that only this English phrase would have 20*6*14*19=31,920 Arabic translations which is totally unpractical especially that the mentioned English phrase has only one Arabic translation which is "صندوق الأمم المتحدة لرعاية الطفولة ". Using the proposed collocation dictionary solves this problem and gives the correct translation accurately and directly. If the word is not a part of a word collocation, the next two methods (transliteration probabilities and Corpus) are used.

N-gram Transliteration probabilities. This method used in case that the word is not a part of a collocation and is not available in the dictionary. IT proposes Arabic word which is the result of concatenating the Arabic character(s) which have the highest transliteration probability to each English character(s), with respect to the transliteration table made by Nasreen AbdulJaleel and Leah S. Larkey after their statistical study [36].

Corpus. This handles both cases for translation or even transliteration. It is working by always sorting the transliterations/translation of every word in the query descending according to their frequencies in the corpus. The resulting Arabic query will have the most used Arabic translation/transliteration for every English word.

6.5 The Segmentation and Query Translation Model

The proposed English to Arabic query translation model works with all the proposed models to produce an accurate Query Translation. The "English/Arabic Query Translation" part of Fig 3 describes the model. Fig 2 and Fig 1 describe the blocks of Fig 3 that describes the single words translation and the transliteration models respectively.

7 Improving Arabic OCR-Degraded Text Retrieval

In this section introduces the final step of the proposed model which is handling the Arabic OCR errors. It starts with defining the OCR accuracy, presenting the model for simulating Arabic OCR errors, and establishing the real training and test sets.

7.1 Defining the OCR accuracy

Scientists and OCR commercial providers usually consider the OCR accuracy from character point of view as the below definition considering the error as character insertion, substation, or deletion:.

$$\text{Character Accuracy} = \frac{\text{n - Number of Errors}}{\text{n}} \tag{1}$$

Where n is the total number of characters in the correct text ("groundtruth") [38].

However, this definition is considered sometimes misleading from many of the OCR commercial consumers who prefer to count the OCR accuracy from word point of view. Considering a sample page of 200 words that contain 1000 character in total, assuming the OCR output of this page having only 10 character errors each in a separate word, this means character accuracy of 98% where it means word accuracy of 90%. This difference is one of the reasons of considering complete words in modeling the OCR Errors in this paper.

7.2 Modeling the OCR Errors

Generally the current models for simulating OCR-Errors are mainly depending on a 1-gram and sometimes n-gram character replacement algorithm. However, in Arabic, as character shape defers up to its position in the words (begin, middle, end, Isolated). So, it is too difficult to include all these variables (7-gram character for example) plus the character position in one model. The proposed model is a word based noisy channel model. It will be trained and tested on the complete words from both the training and test sets.

7.3 Improving IR for Arabic OCR-Degraded Text through Orthographic Query Expansion

The proposed Orthographic Query Expansion model attempts to find different mis-recognized versions of a query word in the text being searched. It starts by checking every word in the Arabic Query against the word DB resulted from training the model of simulating the Arabic OCR errors on the established Training Set. Then the query is expanded by every word found as a probable mistaken word provided by the OCR. Fig 3 in the "Arabic Query Expansion" part illustrate the model and Table 5 illustrates an illustrative example.

Orthographic Query expansion depends directly on the simulation of the OCR-Errors and so we can consider its accuracy as the accuracy of the OCR-Errors simulation model. The main idea is to start with a text based file (the original file), this file is formatted to have only one word per line. Then we convert this file to

Table 5. Example of Query Translation and Orthographic Expansion

Word	Individual words Translations/ Transliteratio ns Probabilities	After Solving the ambiguity using Collocations part in the model	Default Translation After applying the proposed model completely	Expanded Arabic Query to include the expected OCR Errors
Lincoln	18	18		2
United	20			1
States	12	1	1	2
Civil	18			2
War	8			1
Stories	6	6		2
Total	3,732,480	108	1	16

image based one (pdf), then apply the OCR process to have the same file as OCR-Degraded one. Now, we have the same version of the file as original and OCR-Degraded. As I have one word per line, it becomes easy to compare each original word to the equivalent OCR-Degraded word (original/degraded pair). This pair will be used for both training and testing.

7.4 Establishing the Training and Test Sets

For establishing the Training and Test Sets, a pool of 200 long documents from the corpus have been reformatted to have one word per line, converted to image based document (PDF) using "Adobe Acrobat Professional" [39], then the Arabic OCR process (through Sakhr OCR [40]) has been applied on them to have as a result 150 long documents (30 pages each) with their original text and the corresponding OCR-Degraded Text.

7.5 Training Set Statistics

To be able to examine the accuracy of modeling the OCR errors against different sizes of Training Set, different sets of documents have been selected from the Training and Test Set Pool. These sets have the range from 5 to 100 relatively long documents (550-900). Statistics about the Training Set are illustrated in Table 6 and Fig 4.

7.6 Defining the Model Accuracy

The following definition has been considered to define the accuracy for modeling the OCR errors:

$$Accuracy_{TSSn} = \frac{AccurateRplacements_{TSSn}}{TotalOCR\_DegradedWords} \quad (2)$$

i.e. $Accuracy_{TSSn}$ (for certain Training Set Size) equals the no of accurate replacements (with respect to the training set size) divided by the total number of OCR-Degraded words.

Table 6. Training Set Statistics

Training. Set size (no. of documents)	No. of all tested cases (total number of words)	No. of unique cases (same original. / Degraded pair of words)	No. of unique OCR-degraded words	Number of words read correctly by the OCR	No. of words read wrongly by the OCR (real training pairs)	Max no. of n-gram chars i.e. longest word (Degraded / Original)
5	4429	1367	1321	2720	1709	15/14
10	8003	1992	1927	4851	3152	15/14
15	11266	2557	2468	6907	4359	15/14
20	14043	2943	2836	8588	5455	15/14
25	16946	3296	3174	10376	6570	15/14
30	19485	3712	3577	11906	7579	15/14
35	22106	3960	3818	13514	8592	15/14
40	25344	4335	4181	15473	9871	15/14
45	27832	4546	4373	16970	10862	15/14
50	30114	4799	4617	18295	11819	15/14
55	32660	5169	4980	19795	12865	15/14
60	34932	5381	5186	21116	13816	15/14
65	37374	5592	5382	22538	14836	15/14
70	39975	5952	5728	24109	15866	15/14
75	42275	6158	5929	25511	16764	15/14
80	44621	6366	6131	26912	17709	16/14
85	47002	6572	6333	28378	18624	16/14
90	49298	6730	6481	29724	19574	16/14
95	51659	6901	6646	31151	20508	16/14
100	53910	7184	6921	32527	21383	16/14

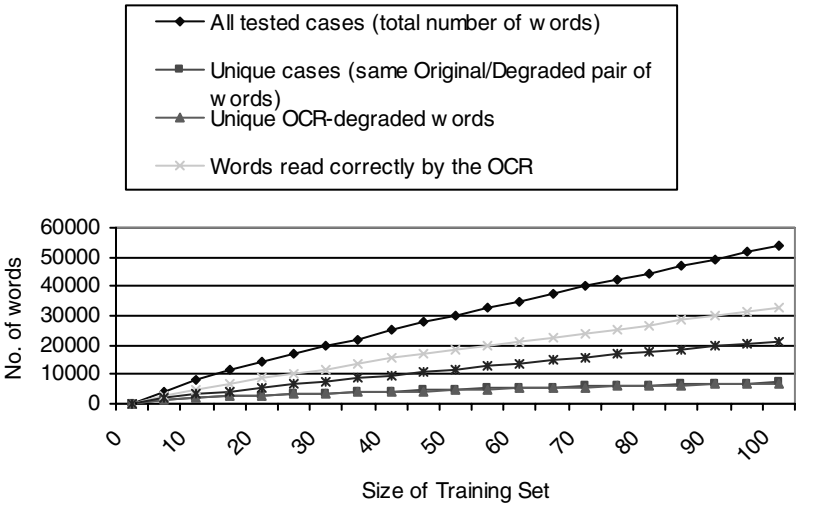


Fig. 4. Training Set Statistics

In other words, if the mistaken OCR-degraded word is available in the training set with the correct original word, then this will be considered as accurate replacement. Otherwise, it will be considered as not accurate. The accuracy for a specified training

set size is the number of accurate replacements that are available in this Training-Set, divided by the total number of the OCR-Degraded words.

8 Experiments

Experiments have been performed to test every option in the model (drawn in Fig 3). This is including the Query Translation accuracy with the different parameters and options for translation, transliteration, and solving ambiguities. Then, the Orthographic Query Expansion part has been tested against different training set sizes.

100 queries have been fed to the Query Translation Model with. Every query has from one to 9 English words including proper names and queries about different fields (political, history, shopping, events, tourism, and miscellaneous). The experiments analyzed the effect of the proposed models on solving the translation ambiguity using the collocations dictionary, the different single words dictionaries, and the proposed corpus. The experiments also examined the transliteration model efficiency and solving the transliteration ambiguity through both the corpus statistics and the characters transliteration probabilities. Table 7, Table 8, and Table 9 summarize the results.

Table 7. Testing results of the Query Translation Model- Collocations Coverage and Accuracy

	Accuracy
Coverage for the fed collocations	95%
Accuracy for the collocations translation	95%

Table 8. Testing results of the Query Translation Model– Transliteration Coverage and Solving Ambiguity

	Corpus	Transliteration Probabilities
The real accurate translation is one of the produced transliteration	90%	
1st hit is the best one of the produced translation	70%	63%
1st hit is the real best translation / translation	60%	54%

Table 9. Summary Testing results of the Query Translation Model Single Words Dictionaries Coverage and Solving Ambiguity

	Dict. 1	Dict. 2	Dict. 3
The real accurate translation is one of the produced translation	94%	82%	94%
1st hit is the best one of the produced translation	98%	95%	80%
1st hit is the real best translation	92%	78%	75

For the Orthographic Query expansion part, which depends directly on the OCR-Errors Simulation model, two test sets has been selected from the Training and test pool. The 1st one includes 50 long documents (26,579 words) and the second one contains double this number (100 long documents containing 51,765 words). There is no intersection between the Training and Test sets. Table 10 illustrates the statistics of the Test Sets. Fig 5 illustrates the model accuracy for the two test sets across different training set sizes.

Table 10. Test Sets Statistics for the OCR-Errors Simulation Model

Category	*Statistical Number*	*Statistical number*
No of documents (long documents)	50	100
No of unique pairs (Original word-degraded word)	4,208	6,528
Total no. of words	26,579	51,765
No of words read correctly	15, 823	30,995
No of words read wrong	10,756	20,761
No of words read wrong and (and not read as NULL)	10,175	19,598

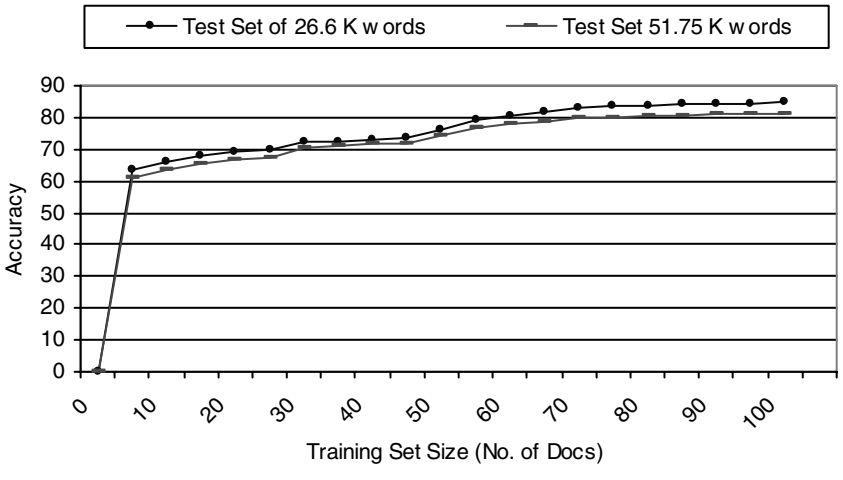


Fig. 5. Accuracy of the OCR-Errors Simulation model

9 Conclusions

The most important contribution was proposing the Query Translation and Expansion model which covers the collocation, transliteration and the normal single English words inside the Query and expands the Arabic query to handle the expected Arabic OCR Errors.

The collocation detection and translation model, supported by the well-introduced collocation dictionary, gives high accuracy in detecting and translating collocations even when they are written in non exact way (derivations). The only non-detected

collocations are those which are local one like 'African Cup' i.e. as the collocation dictionary size increases, the collocation detection, translation, and so the overall query translation accuracy will also be enhanced.

Solving the transliteration ambiguity is effective through either the corpus or the n-gram character transliteration probabilities. However, the corpus option gave better results.

The three single words dictionaries gave different results, yet compared with the other two dictionaries, Dictionary 1, which is based on “ArabeYes” project data, gave a significant accuracy although it is much smaller. This highlights the importance for the dictionary for Query Translation to be modern and practical. The 2 other dictionaries gave many possible Arabic equivalents, even if they are rarely used or are likely to mislead in query translation. The corpus may give those non-relevant translations a weight, not because it is the correct translation in this case, but may be because the term is frequently used in general, but indicating another meaning rather than what is meant in the query.

Orthographic Query expansion based on the proposed model for simulating the OCR errors starts with giving intermediate accuracy with very limited training set then high accuracy after increasing the size of the training set even with increasing the Test Set size.

References

1. The official web site of the Library of Congress (Retrieved December 4, 2006), http://www.loc.gov/about/facts.html
2. Kanungo, T., Marton, G.A., Bulbul, O.: OmniPage vs. Sakhr: Paired model evaluation of two Arabic OCR products. In: Proc. of SPIE Conf. on Document Recognition and Retrieval (1999)
3. Al-Kharashi, I.A., Evans, M.W.: Comparing words, stems, and roots as index terms in an Arabic information retrieval system. Journal of the American Society for Information Science (JASIS) 5(8), 548–560 (1994)
4. Abu-Salem, H., Al-Omari, M., Evens, M.: Stemming Methodologies over Individual Query Words for an Arabic Information Retrieval System. JASIS 50(6), 524–529 (1999)
5. Beesley, K.: Arabic Morphological Analysis on the Internet. In: Proceedings of the 6th International Conference and Exhibition on Multi-lingual Computing, Cambridge (1998)
6. Aljlayl, M., Frieder, O.: On Arabic Search: Improving the Retrieval Effectiveness Via Light Stemming Approach. In: Proceeding the 11th ACM International Conference on Information and Knowledge Management, Illions Institute of Technology, pp. 340–347. ACM Press, New York
7. Khoja, S., Garside, R.: Stemming Arabic Text. Computing Department, Lancaster University, UK (Retrieved, April 2007), http://zeus.cs.pacificu.edu/shereen/research.htm
8. Larkey, L.S., Connell, M.E.: Arabic Information Retrieval at Umass in TREC-10. In: Text REtrieval Conference (2001)
9. Hunston, S.: Corpora in applied linguistics. Cambridge University Press, Cambridge (2002)
10. Hmeidi, I., Kanaan, G., Evens, M.: Design and Implementation of Automatic Indexing for Information Retrieval with Arabic Documents. Journal of the American Society for Information Science 48(10), 867–881 (1997)

11. Goweder, A., De Roeck, A.: Assessment of a significant Arabic corpus. In: The Arabic NLP Workshop at ACL/EACL 2001, Toulouse, France (2001)
12. Darwish, K., Doermann, D., Jones, R., Oard, D., Rautiainen, M.: TREC-10 experiments at University of Maryland CLIR and video. In: Text RE-trieval Conference TREC10 Proceedings, Gaithersburg, MD, pp. 549–562 (2001)
13. Pirkola, A.: The Effects of Query Structure and Dictionary Setups in a Dictionary-based Cross-Language Information Retrieval. In: SIGIR 1998, Melbourne, Australia (1998)
14. Oard, D.: A Comparative Study of Query and Document Translation for Cross-language Information Retrieval. In: Proceedings of the 3rd Conference of the Association for Machine Translation in the Americas, pp. 472–483 (1998)
15. Davis, M.W., Dunning, T.E.: Query Translation Using Evolutionary Programming for Multi-lingual Information Retrieval. In: Proceedings of the Fifth Annual Conference on Evolutionary Programming (1995)
16. Landauer, T.K., Dumais, S.T., Littman, M.L.: Full Automatic Cross-Language Document Retrieval using Latent Semantic Indexing. In: 1996, update of the original paper on the 6th Conf. of UW center for New OED and Text Research, pp. 31–38 (1990)
17. Sheridan, P., Ballerini, J.P.: Experiments in Multilingual Information Retrieval using the SPIDER System. In: The 19th Annual International ACM SIGIR 1996, pp. 58–65 (1996)
18. Adriani, M., Croft, W.: The Effectiveness of a Dictionary-Based Technique for Indonesian-English Cross-Language Text Retrieval, CLIR Technical Report IR-170, University of Massachusetts, Amherst (1997)
19. Ballesteros, L., Croft, B.: Dictionary Methods for Cross-Lingual Information Retrieval. In: 7th DEXA Conf. on Database and Expert Systems Applications, pp. 791–801 (1996)
20. Ballesteros, L., Croft, B.: Phrasal Translation and Query Expansion Techniques for Cross-language Information Retrieval. In: SIGIR 1997, pp. 84–91 (1997)
21. Xu, J., Croft, W.B.: Query Expansion using Local and Global Document Analysis. In: The 19th Annual International ACM SIGIR 1996, Zurich, Switzerland, pp. 4–11 (1996)
22. Ballesteros, L., Croft, B.: Resolving Ambiguity for Cross-Language Retrieval. In: SIGIR 1998, pp. 64–71 (1998)
23. Aljlayl, M., Frieder, O.: Effective Arabic-English Cross-Language Information Retrieval Via Machine-Readable Dictionaries and Machine Translation, Information Retrieval Laboratory, Illinois Institute of Technology (2002)
24. The Text REtrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense, http://trec.nist.gov/
25. Hasnah, A., Evens, M.: Arabic/English Cross Language Information Retrieval Using a Bilingual Dictionary, Department of Computer Science University- Qatar, and Illinois Institute of Technology
26. Darwish, K.: Probabilistic Methods for Searching OCR-Degraded Arabic Text, A PhD Dissertation, University of Maryland, College Park (2003)
27. Elaraby Ahmed, M.A.M.: A Large-Scale Computational Processor of the Arabic Morphology, and Applications, M.Sc. Thesis, Cairo University, Faculty of Engineering, pp. 37–39 (2000)
28. (Retrieved December 4, 2006), http://www.moheet.com
29. Fellbaum, C.: WordNet, An Electronic Lexical Database. MIT Press, Cambridge (1998)
30. Adly, A.: Senior Translation Consultant
31. Retrieved December 4, 2008, http://www.arabeyes.org
32. Last time visited April 15, 2007, http://sourceforge.net/project/showfiles.php?group_id=34866&package_id=93898
33. Last time visited April 15, 2007, http://crl.nmsu.edu/Resources/lang_res/arabic.html
34. Last time visited April 15, 2007, http://wordnet.princeton.edu/

35. Last time visited April 15, 2007, http://dictionary.Sakhr.com/
36. AbdulJaleel, N., Larkey, L.S.: Statistical Transliteration for English-Arabic Cross Language Information Retrieval. In: Proceedings of the twelfth international conference on Information and knowledge management table of contents, New Orleans, LA, USA (2003)
37. WordNet documentations, MORHY (7N), Princeton University, Cognitive Science Laboratory (January 2005), http://wordnet.princeton.edu/
38. Rice, S.V., Kanai, J., Nartker, T.A.: The 3rd Annual Test of OCR Accuracy, TR 94-03, ISRI, University of Nevada, Las Vegas (April 1994)
39. Adobe Company, http://www.adobe.com
40. Sakhr Software, http://www.Sakhr.com

NLP for Shallow Question Answering of Legal Documents Using Graphs*

Alfredo Monroy[1], Hiram Calvo[1,2], and Alexander Gelbukh[1]

[1] Center for Computing Research, National Polytechnic Institute
Mexico City, 07738, Mexico
alopezm301@ipn.mx, hcalvo@cic.ipn.mx, gelbukh@gelbukh.com
[2] Nara Institute of Science and Technology, Takayama, Ikoma, Nara 630-0192, Japan
calvo@is.naist.jp

Abstract. Previous work has shown that modeling relationships between articles of a regulation as vertices of a graph network works twice as better than traditional information retrieval systems for returning articles relevant to the question. In this work we experiment by using natural language techniques such as lemmatizing and using manual and automatic thesauri for improving question based document retrieval. For the construction of the graph, we follow the approach of representing the set of all the articles as a graph; the question is split in two parts, and each of them is added as part of the graph. Then several paths are constructed from part A of the question to part B, so that the shortest path contains the relevant articles to the question. We evaluate our method comparing the answers given by a traditional information retrieval system—vector space model adjusted for article retrieval, instead of document retrieval—and the answers to 21 questions given manually by the general lawyer of the National Polytechnic Institute, based on 25 different regulations (academy regulation, scholarships regulation, postgraduate studies regulation, etc.); with the answer of our system based on the same set of regulations. We found that lemmatizing increases performance in around 10%, while the use of thesaurus has a low impact.

1 Introduction

Previous work [20] has shown that modelling relationships between articles of a regulation as vertices of a graph network works twice as better than traditional information retrieval systems for returning articles relevant to the question. Despite being that approach language independent, in this work we experiment by using natural language techniques such as lemmatizing and using manual and automatic thesauri for improving question based document retrieval. We focus in Spanish language. For automatic thesaurus we used a distributional thesaurus [19], and for the manual thesaurus we used a human oriented dictionary (Anaya) [21]. The advantage of using a distributional thesaurus is that the approach remains language independent—not being the

* We thank the support of Mexican Government (SNI, SIP-IPN, COFAA-IPN, and PIFI-IPN) and Japanese Government. Second author is a JSPS fellow.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 498–508, 2009.

case with the human oriented dictionary. On the other hand, using a lemmatizer would make this approach language dependent, as for a particular a lemmatizer is needed. In particular, we want to measure the advantage of using these resources and we want to measure the possible benefit from adding this kind of information. Our system gives answers which consist of set of articles related to the question and also the relevant articles related with them to complement the answer. This is called shallow QA, because its operation lies in the middle of snippet retrieval and giving the exact answer.

We test our system with regard to a traditional vector space model information retrieval system to answer questions particularly for the Spanish language given a set of 25 regulation documents from the National Polytechnic Institute. For details of the rest of the System, see sections 2 and 3. The addition of NLP techniques is explained with further detail in Section 4. Evaluation and experiments are shown in Sections 5 and 6.

2 Related Work

There are not many works particularly devoted to the legal domain, despite of its wide use and application. Particularly for the legal domain, the workshop *Question Answering for interrogating legal documents* took place in 2003, in the framework of the JURIX Forum (The foundation for Legal Knowledge Based Systems). Several works showed that a common problem is that traditional Information Retrieval Methods are not adequate to find the relevant fragments which answer legal questions because they do not consider the logic relationships between articles. In addition, many questions require an answer which cannot be found explicitly in a single article, or fragments of them, but intrinsically in the relationship between articles [9,10]. Some works use logic inference mechanisms such as COGEX System [14] and the system by Quaresma *et al.* [7]. However, these systems need expensive resources such as ontologies, axioms, and are language dependent. To avoid such requirements, we use a graph for capturing the relationships between articles in regulations as proposed in [20].

3 System Design

The architecture of this SQAS is based on the work shown in [20]. It was designed considering common characteristics posed by regulation documents, as well as the kind of questions and answers expected by the user. It is important to mention that regulation texts have a defined structure, they are composed of chapters, and these, in turn, are subdivided in articles. This makes possible to use different techniques which with other kind of texts would not be possible. Articles from a single regulation text are related between them, and also there are links between different regulations.

We focus in questions where the answer can be given as a set of articles from a regulation. For example, for the question: *Is it possible to award a honourable mention to a bachelor if he chose to graduate using the qualification option?* the answer can be given as a set of articles: *See Chapter II of "On Graduating Options" and article 13, Chapter VII, "On the Professional exam", article 43.* When one looks to such articles, they say:

The option of graduating by qualification proceeds if the student's average is higher than 9.0 and all of his subjects were approved in an ordinary way.

The candidate can only aspire to the award of honorific mention, if, in addition to covering other requisites disposed in this regulation, he presents professional exam

From those articles, it can be concluded that Graduating by qualification does not require presenting the professional exam, so that it cannot be included within the article 43 fraction II of the mentioned Regulation; when this option is chosen, the candidate cannot obtain honorific mention.

This system does not return a completely logically evaluated answer such as 'yes' or 'no', which would need a more complex machinery; because of this, it is considered as part of the Shallow Question Answering Systems.

The fundamental parts of the system are shown in the graph in Figure 1. Question pre-processing consists on constructing the query based on the question, and Answer Extraction consists on adding the generated query as two new nodes (A and B). Then the shortest path between A and B is sought. We will show that this path contains articles highly related to the question, and they share certain degree of similarity between them. The following picture depicts this. A1, A2, ... A5 are articles, while A and B are parts of the question.

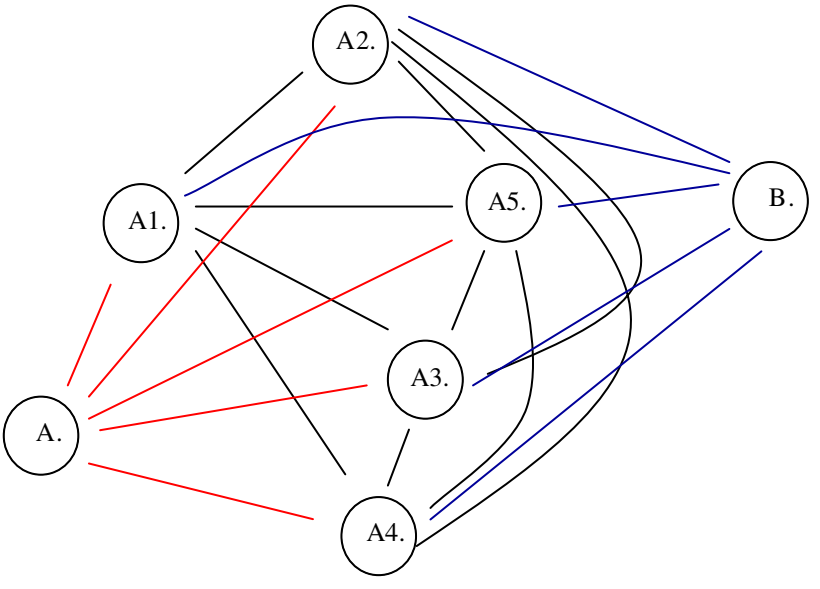


Fig. 1. Graph representing articles (A1, A2, ... A5) and question (A, B) for the Question Answering System

3.1 Graph Description

The documents (articles of regulations) were represented as vectors, following the Vector Space Model [12, 13]. Each term was weighted by the TF·IDF measure (Term Frequency Inverse Document Frequency). See equations (**1**), (**2**) and (**3**).

$$tfidf = tf_{t,j} \cdot idf_i \tag{1}$$

$$tf_{t,j} = \frac{n_{i,j}}{\sum n_{k,j}} \tag{2}$$

Where $n_{i,j}$ corresponds to the number of occurrences for each term from the article a_j and the denominator represents the occurrence of all terms in the article a_j

$$idf_i = \log \frac{|A|}{\left|\{a_j : t_i \in a_j\}\right|} \tag{3}$$

Where |A| is the total number of articles in the document collection and $\left|\{a_j : t_i \in a_j\}\right|$ is the number of articles where the term t_i appears.

Finally, a graph was constructed for each document collection, see Figure 2. Each node represents an article of a regulation text, and the associated values to the vertices $V_{i,j}$ between each pair of nodes represents the inverse value of the standard similarity cosine measure.

3.2 Question Processing

The question is pre-processed and then integrated into the graph in the same way that the regulation collection. Each question is converted to lowercase, punctuation symbols are eliminated (parenthesis, hyphens, numbers, etc.). Stopwords are kept (QK) or removed (QR) and finally, words that do not exist in the document collection are removed—this is equivalent to finding a similarity measure of 0 for them. The weighting values are calculated with regard to the document collection (See eq. (2)).

From the query of the previous module, two new *articles* are added to the graphs. The answer extraction consists on finding the paths of minimal weight using the Dijkstra algorithm: Once the first minimal path is found, the nodes which constitute it are eliminated, and the Dijkstra algorithm is run again until the graph becomes disconnected for the pair of nodes which constitute the query. The *answer paths* are ordered from less to more weight, and are returned to the user with the text of each article corresponding to the node in the regulation collection.

4 NLP Techniques for Shallow QA

Generally speaking, a thesaurus can be considered as a list of words with other words related to them. For example[1] *car* has the following **synonyms**: auto, automobile, machine, motor, motorcar, motor vehicle; and its **related words** are: bus, coach, minibus; beach buggy, brougham, compact, convertible, coupe, dune buggy, fastback, gas-guzzler, hardtop, hatchback, hot rod, jeep, limousine, roadster, sedan, sports car, station wagon, stock car, subcompact, van; flivver, jalopy.

[1] Example from the Merriam Webster on-line thesaurus (http://www.merriam-webster.com)/

Related words can be found generally in two ways:

- Manually built—Made by humans who collect the set of related words which they consider that can be associated to a specific word.
- Automatically built (distributional thesaurus)—Built automatically from a selected corpus. They obtain related words usually by comparing contexts on which the word is used [19]. For example, for the phrases *drink juice, drink lemonade, make juice, make lemonade, delicious juice, delicious lemonade*, then it might have enough evidence to conclude that *juice* and *lemonade* are related words.

On the other hand, thesaurus can be classified by their domain:

- Specific domain—Thesauri (manual or distributional) built based on a particular subject, *v. gr.* medicine.
- General domain—They have words from different areas, covering a general vocabulary.

Thesauri have been extensively used for query expansion [15, 16], so we expected that our system would benefit from their usage. We used mainly two tools: (1) A distributional thesaurus built from the Encarta Encyclopedia [17,18]; and (2) A manual thesaurus based on the Anaya dictionary [21].

We expand the query (with already omitted words): *go procedure representative election alumni council scholar* to, for example: *go transport take guide drive use procedure representative alumni student council meeting scholar educational*. Some words have a greater number of related words than others; some words are not present in the thesaurus, so they are not expanded.

We used two thesauri for these experiments:

1. A distributional thesaurus created from the Encyclopedia Encarta [17, 18]
2. A thesaurus based on the Anaya dictionary [21] using the synonyms from that dictionary.

Another NLP technique used in this work was lemmatizing. Although this is a language-dependent resource, many languages have a lemmatizer; moreover, it is possible to lemmatize unsupervisedly.

5 Evaluation

The evaluation of the QAS is based on the following criteria:

i Relevance: The output of the QAS should answer to the questions of the kind “yes or no”; this is measured by determining if the articles that the general lawyer considers to produce an answers were returned by the system.

ii Noise degree in the answer: Alien and irrelevant information that the system returns is quantified.

To implement the measure of these both criteria, we use the following procedure: Initially, the answer is limited to 100 articles. It is unlikely that the answer is found after such number of articles. The returned articles were divided in groups of 5 (the

maximum number of articles that can be found in a single answer of the lawyer). Each group was given the value V_i, following equation (4)

$$V_i = 1 - \frac{(n-1)}{N} \tag{4}$$

Where n is the group number (the first 5 articles constitute the group 1, the next 5, group 2, etc). N is the maximum number of packages that can be found (N=20, as 5·20=100 articles).

Finally, each article returned for a determined answer is graded using the following expression:

$$Grade \quad CR_i = \frac{\sum_{i=1}^{n} nV_i}{n_{AR}} \tag{5}$$

Where CR_i is the grade assigned to the answer of the i-th question, n is the package number where the answer-article was found, and n_{AR} is the number of answer-articles found.

The final mark of the system is the average of the grades CR_i, i.e.:

$$Grade \quad CS = \frac{\sum_{i=1}^{np} CR_i}{np} \tag{6}$$

Where np is the total number of questions for evaluation.

6 Experiments and Results

For this report, our system was tested with 21 questions. We tested against a basic Information Retrieval System (IRS) based on the vector space model. This IRS uses the same set of vectors used for the construction of the graph, but this time they were directly compared with the cosine measure with the query. The vectors which are more similar to the query are returned, so that the output of this system is the set of articles relevant to the query. The results were then compared with the results of our QAS.

As we mentioned in Section 3.2, we tested with keeping and removing stopwords. This yields two derived regulation document collections: DCK (Document collection keeping stopwords) and DCR (Document collection removing stopwords), and two kinds of queries, which correspond to keeping or removing stopwords in the query.

Additionally, we performed four experiments with regard to the automatic division of the query in part A and part B. After the procedure described in sections 3.2, the query is divided in two new *articles* (nodes) A and B with the following contents, according to one of the four followings types of division:

Half Division (H): Node A will contain the left half of the question, and Node B will contain the rest. If the number of words in the query is odd, the Node A will contain one word more than Node B.

Mixed Division (M): In this type of division, terms are mixed: odd words are in Node A and even words are in Node B.

Reversed Half Division (H'): As in Half Division (H) but the contents of node A and B are exchanged.
Reversed Mixed Division (M'): As in Mixed Division (M) but contents of the Node A and Node B are exchanged. Even words are in Node A and odd words are in Node B.

Table 1. Results with no lemmatization

Stopwordst				Division	Precision	Answers found		Not found
Article List		Query						
Keep	Remove	Keep	Remove			Fully	Partially	
Our system								
LBA		✓		H	0.5342	12	5	4
				M	0.4988	13	3	5
				H´	**0.5351**	**12**	**5**	**4**
				M´	0.5023	13	3	5
			✓	H	0.4851	10	5	6
				M	0.4865	10	6	5
				H´	0.4858	9	6	6
				M´	0.4889	10	6	5
	LBA_R		✓	H	0.4603	9	6	6
				M	0.4723	10	5	6
				H´	0.4683	10	5	6
				M´	0.4716	10	5	6
IR system based on Vector Model Space								
LBA		✓		-	**0.2253**	**3**	**5**	**13**
	LBA_R		✓	-	0.1892	4	6	11

Table 2. Results using lemmatization

Stopwordst				Division	Precision	Answers found		Not found
Article List		Query						
Keep	Remove	Keep	Remove			Fully	Partially	
Our system								
LBA		✓		H	0.5170	11	8	2
				M	0.5536	12	8	1
				H´	0.5175	12	7	2
				M´	0.5548	12	8	1
			✓	H	0.5187	12	7	2
				M	0.6151	12	8	1
				H´	0.5175	12	7	2
				M´	0.5548	12	8	1
	LBA_R		✓	H	0.5026	13	6	2
				M	0.6103	13	7	1
				H´	0.5115	13	6	2
				M´	**0.6230**	**13**	**7**	**1**
IR system based on Vector Model Space								
LBA		✓		-	0.1976	2	5	14
	LBA_R		✓	-	**0.3055**	**5**	**7**	**9**

Table 3. Results using lemmatization and distributional Thesaurus

Stopwordst				Division	Precision	Answers found		Not found
Article List		Query						
Keep	Remove	Keep	Remove			Fully	Partially	
Our system								
LBA		✓		H	0.5228	12	6	3
				M	0.4933	11	6	4
				H´	0.5253	12	6	3
				M´	0.4905	11	6	4
			✓	H	0.5250	12	6	3
				M	0.5138	12	5	4
				H´	**0.5273**	**12**	**6**	**3**
				M´	0.5126	12	5	4
	LBA_R		✓	H	0.5228	12	6	3
				M	0.4933	11	6	4
				H´	0.5253	12	6	3
				M´	0.4905	11	6	4
IR system based on Vector Model Space								
LBA		✓		-	0.1869	3	3	15
	LBA_R		✓	-	**0.2107**	**4**	**4**	**13**

Table 4. Results using lemmatization and the Anaya Thesaurus

Stopwordst				Division	Precision	Answers found		Not found
Article List		Query						
Keep	Remove	Keep	Remove			Fully	Partially	
Our system								
LBA		✓		H	0.4689	12	6	3
				M	**0.5461**	**13**	**6**	**2**
				H´	0.4719	11	7	3
				M´	0.5456	13	6	2
			✓	H	0.4883	13	6	2
				M	0.5257	11	6	4
				H´	0.4856	12	7	2
				M´	0.5276	11	6	4
	LBA_R		✓	H	0.4736	12	7	2
				M	0.5325	12	6	3
				H´	0.4783	12	7	2
				M´	0.5336	12	6	3
IR system based on Vector Model Space								
LBA		✓		-	0.2428	4	4	13
	LBA_R		✓	-	**0.2815**	**4**	**7**	**10**

We tested every combination of these parameters. Table 1 shows the description and name of each experiment. We compared the performance of our system with two experiments based in the document collection DCK (Document Collection Keeping

Table 5. Summary of best results

Stopwordst				Division	Precision	Answers found		Not found
Article List		Query						
Keep	Remove	Keep	Remove			Fully	Partially	
With no lemmatization								
LBA		✓		H	0.5351	12	5	4
lemmatizing								
	LBA_R		✓	**DMT´**	**0.6230**	**13**	**7**	**1**
lemmatizing+ Anaya dictionary								
LBA		✓		DMT	0.5461	13	6	2
lemmatizing + distibutional thesaurus								
LBA			✓	DM´	0.5273	12	6	3

stopwords) and DCR (Document Collection Removing stopwords). These experiments are shown in the last two rows of Table 1.

7 Conclusions and Future Work

According to the reported values, the best result was found for the experiment where the list of articles and query keep non-content words, use division by the half, and walking from node B to node A. This is not significantly changed when walking from node A to node B.

Table 2 shows the results for the experiments where the article list was lemmatized. The best result was obtained for the experiment where the list of articles and query had stopwords removed. Precision is 0.6230 for this case. The precision when modifying the division type are very close, so that it has almost the same impact to traverse the nodes from A to B than doing so the reverse way. We compare with the traditional information retrieval system which obtains a precision of 0.3055. Lemmatizing the list of articles produced a rise in precision.

In Table 3, we show the results of adding a distributional thesaurus, in addition to lemmatizing the list of articles. The best result was found for the experiment where the article list and query keep stopwords. This value is 0.5461. The distributional thesaurus seems to have added noise, since the performance is inferior to previous experiments.

Finally, we experimented with using a manual thesaurus, which is based in the human oriented dictionary Anaya for Spanish (See Table 4). The best result was obtained when lemmatizing articles, keeping content words, but removing them from the query. The precision for this case was 0.5273.

From Table 1 to Table 4, we observed that the results of the proposed system are better than the traditional IR system based on a Vector Space Model. The best results are summarized in Table 5. Using a lemmatizer on the list of articles improves the performance by approximately 10% from the previous work.

The thesauri, in addition to the lemmatizer, were used with the purpose of improving the search results; however, we did not obtain significant improvement. Furthermore, combining both strategies led to a decrease of performance with regard to only

lemmatizing. The thesaurus expands the terms of the query. Usually it augments it with 9 or more terms per content word. It is possible that 9 terms are so many, so that we should experiment by using only 2 or 3 of them. In addition, the thesauri we are using (both manual and distributional) are general, as they are based in Anaya Dictionary and Encarta Encyclopaedia, respectively. So is then, that the query is expanded with terms not necessarily related with the context, creating confusion.

In general, dividing the query in different ways, as well as traversing the nodes in one way or another does not affect very much the performance of the system. This is a good effect for the model, since it shows that finding answers within the information contained there is not highly sensible to a particular way of using the graph, *i.e.*, the information is contained mainly in the way the structure is created, and not in the way it is traversed.

There is still room for improvement, as a future work we propose using a distributional thesaurus based on the same corpus of legal documents, as well as using different degrees of query expansion and using different similarity measures other than TF·IDF.

References

1. Hirschman, L., Gaizauskas, R.: Natural Language Question Answering: The View From Here. Natural Language Engineering 7(4), 275–300 (2001)
2. Hoojung, C., Song, Y.-I., Han, K.-S., Yoon, D.-S., Lee, J.-Y., Rim, H.-C.: A Practical QA System in Restricted Domains. In: Workshop on Question Answering in Restricted Domains. 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, pp. 39–45 (2004)
3. Erik, T., Sang, K., Bouma, G., de Rijke, M.: Developing Offline Strategies for Answering Medical Questions. In: Workshop on Question Answering in Restricted Domains. 20th National Conference on Artificial Intelligence (AAAI 2005), Pittsburgh, PA, pp. 41–45 (2005)
4. Fabio, R., Dowdall, J., Schneider, G.: Answering questions in the genomics domain. In: Proceedings of the ACL 2004 Workshop on Question Answering in Restricted Domains, Barcelona, Spain, pp. 46–53 (2004)
5. Niu, Y., Graeme, H.: Analysis of Semantic Classes in Medical Text for Question Answering. In: Workshop on Question Answering in Restricted Domains. 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, pp. 54–61 (2004)
6. Zhuo, Z., Da Sylva, L., Davidson, C., Lizarralde, G., Nie, J.-Y.: Domain-Specific QA for the Construction Sector. In: Workshop of IR4QA: Information Retrieval for Question Answering, 27th ACM-SIGIR, Sheffield (July 2004)
7. Paulo, Q., Rodrigues, I.P.: A question-answering system for Portuguese juridical documents. In: Proceedings of the 10th international conference on Artificial intelligence and law. International Conference on Artificial Intelligence and Law, Bologna, Italy, pp. 256–257 (2005)
8. Paulo, Q., Rodrigues, I.P.: A collaborative legal information retrieval system using dynamic logic programming. In: Proceedings of the 7th International Conference on Artificial Intelligence and Law, Oslo, Norway, pp. 190–191 (1999)
9. Doan-Nguyen, H., Kosseim, L.: The problem of precision in restricted-domain question-answering. Some proposed methods of improvement. In: Workshop on Question Answering in Restricted Domains. 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, pp. 8–15 (2004)

10. Diekema Anne, R., Yilmazel, O., Liddy, E.D.: Evaluation of restricted domain question-answering systems. In: Workshop on Question Answering in Restricted Domains. 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain, pp. 2–7 (2004)
11. Rada, M.: Random Walks on Text Structures. In: Gelbukh, A. (ed.) CICLing 2006. LNCS, vol. 3878. Springer, Heidelberg (2006)
12. Manning Christopher, D., Schutze, H.: Foundations of Statistical Natural Language processing. MIT Press, Cambridge (1999)
13. Salton, G., Wong, A., Yang, C.S.: A vector Space Model for Automatic Indexing. Information Retrieval and Language Processing (1975)
14. Dan, M., Clark, C., Harabagiu, S., Maiorano, S.: COGEX: a logic prover for question answering. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Canada, vol. 1, pp. 87–93 (2003)
15. Rila, M., Tokunaga, T., Tanaka, H.: Query expansion using heterogeneus thesauri. Information Processing and Management 36, 361–378 (2000)
16. Pizzato, L.A.S., de Lima, V.L.S.: Evaluation of a thesaurus-based query expansion technique. In: Mamede, N.J., Baptista, J., Trancoso, I., Nunes, M.d.G.V. (eds.) PROPOR 2003. LNCS, vol. 2721, pp. 251–258. Springer, Heidelberg (2003)
17. Calvo, H., Gelbukh, A., Kilgarriff, A.: Distributional thesaurus versus wordNet: A comparison of backoff techniques for unsupervised PP attachment. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 177–188. Springer, Heidelberg (2005)
18. Biblioteca de Consulta Microsoft Encarta 2004, Microsoft Corporation (1994–2004)
19. Lin, D.: An information-theoretic measure of similarity. In: Proceedings of ICML 1998, pp. 296–304 (1998)
20. Alfredo, M., Calvo, H., Gelbukh, A.: Using Graphs for Shallow Question Answering on Legal Documents. In: Gelbukh, A., Morales, E.F. (eds.) MICAI 2008. LNCS, vol. 5317, pp. 165–173. Springer, Heidelberg (2008)
21. Lázaro Carreter, F. (ed.): Diccionario Anaya de la Lengua, Vox (1991)

Semantic Clustering for a Functional Text Classification Task

Thomas Lippincott and Rebecca Passonneau

Columbia University
Department of Computer Science
Center for Computational Learning Systems
New York, NY USA
{tom,becky}@cs.columbia.edu

Abstract. We describe a semantic clustering method designed to address shortcomings in the common bag-of-words document representation for functional semantic classification tasks. The method uses WordNet-based distance metrics to construct a similarity matrix, and expectation maximization to find and represent clusters of semantically-related terms. Using these clusters as features for machine learning helps maintain performance across distinct, domain-specific vocabularies while reducing the size of the document representation. We present promising results along these lines, and evaluate several algorithms and parameters that influence machine learning performance. We discuss limitations of the study and future work for optimizing and evaluating the method.

1 Introduction

The bag-of-words document representation achieves excellent performance on many machine learning tasks. However, this performance can be sensitive to the changes in vocabulary that occur when the training data cannot be reasonably expected to be representative of all the potential testing data. In this situation, it may be possible to exploit higher-level relationships between the vocabularies by consolidating and generalizing the specific bag-of-words features into a smaller number of semantic clusters using an external semantic resource. This would have the dual benefits of retaining performance across domains and reducing the dimensionality of the document representation.

The approach presented here was developed in response to characteristics of the machine learning phase of the ANTArT [1], a component of the CLiMB research project [2]. CLiMB developed a toolkit for image catalogers that facilitates harvesting descriptive meta-data from scholarly text for annotating digital image collections. ANTArT, collaborating with experts in art history and image cataloging, developed a set of functional semantic labels to characterize how art-historical texts function with respect to their associated images (see Table 4 for the complete list), drawing on texts from four art-historical periods.

Three data sets were prepared from art history texts covering two time periods: Near Eastern art (two data sets), and Medieval art (one data set). Each

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 509–522, 2009.

data set consisted of 25 images of works of art and the paragraphs from the text describing them. The annotators were asked to tag each sentence with the relevant functional semantic labels using a special purpose browser interface that included a training session. These annotations were then used to train and evaluate binary classifiers for each label. In this study we focus on the three most frequently-used labels, *Image Content*, *Historical Context* and *Implementation*.

The rest of the paper is organized as follows: we describe the baseline results and the reasoning that led us to investigate the clustering method, and summarize previous work with similar aims. After outlining the steps of semantic clustering, we describe the algorithms and parameters that we evaluated. We present our results along several dimensions and discuss their significance, and conclude with a discussion of future work.

2 Motivation

Table 1 shows classifier performance by train and test data for the three functional semantic labels using bag-of-words features. We swap in each of the three data sets as train and test data. The performance is highest when both data sets focus on the same time period (Near East). When the Medieval data set is used for training or testing, performance decreases dramatically. This happens to a greater degree with the *Historical Context* and *Implementation* labels than with *Image Content*, which we attribute to greater sensitivity of those labels to period-specific terminology. This shows that the bag-of-words model does not maintain its performance across historical periods for important functional semantic labels. This will be a recurring problem as data sets from different historical periods are added.

Early in the ANTArT study, we noted [3] that certain semantic word classes are correlated with the functional semantic labels. The intuition was that, while the data sets may exhibit distinct vocabularies, the distribution of words with the same hypernyms remains discriminative. Table 2 shows pairs of terms from texts on the two historical periods that are highly-correlated with the *Historical Context* label in their respective data set and belong to the same semantic class.

Table 1. ROC Area of support vector machine classifiers for the three functional semantic labels using bag-of-words feature sets. This shows the performance hit taken when train and test data come from different domains.

Data Sets		Label		
Train	Test	Historical Context	Implementation	Image Content
Near East 1	Near East 2	0.630	0.607	0.602
Near East 1	Medieval	0.576	0.518	0.576
Near East 2	Near East 1	0.620	0.575	0.617
Near East 2	Medieval	0.514	0.521	0.578
Medieval	Near East 2	0.573	0.564	0.597
Medieval	Near East 1	0.541	0.538	0.603

Table 2. Examples of discriminative bag-of-words features for the *Historical Context* label

Near East	Medieval	Semantic Class
Sumer	England	*Geographic region*
priest	monk	*Religious person*
Egyptian	Irish	*Nationality*
ancient	medieval	*Time period*
ziggurat	cathedral	*Place of worship*

Several manually enumerated semantic classes were assembled, such as *body-parts* and *time-values*. While this method addresses feature set reduction from a semantic perspective, it could not scale or generalize, as it required manual compilation of the semantic classes, solely from the training data. The clustering method presented here is an attempt to automate and generalize this intuition.

3 Previous Work

Most approaches to word clustering are statistical, using distribution, collocation and mutual information to collapse features containing redundant information into a smaller number of equivalence classes. Pereira et al. [4] describe a method that uses distribution within different syntactic contexts as the basis for clustering. Tishby et al. [5] introduced the *information bottleneck* approach to compressing feature sets while maximally preserving information about the target variable. Bekkerman et al. [6] achieved excellent results on a common multi-label classification task (20 Newsgroup) by combining this approach with support vector machines, and discuss how relevant vocabulary size can affect the potential benefits of word clustering, in the context of different labeling methodologies. Slonim et al. [7] applied the approach to a document clustering task, and report performance by the number of word clusters generated.

Few studies have made use of semantic resources in word clustering. Termier et al. [8] attempted to combine the statistical approach (latent semantic indexing) with semantic properties derived from WordNet. The results showed the semantic information actually decreasing performance of simple LSI, and while several of the theoretical benefits are discussed, they are not achieved. However, the classification task that the hybrid method was evaluated on (Reuters, by topic) does not present the issues that would benefit from using an external semantic resource.

Several algorithms have been proposed for computing "semantic similarity" between two WordNet concepts based on hypo/hypernym relationships. In addition to the three purely WordNet-based metrics we used in this study, Resnik [9], Jiang [10] and Lin [11] have proposed measures that also consider information theoretic properties of the concepts in an auxiliary corpus. Budanitsky and Hirst [12] give a high-level overview of the potential for evaluating and comparing such metrics, and note the difficulty of designing simple measurements that would be generally useful for the nebulous concept "semantic similarity".

4 Data Sets

Table 3 compares the size of the data sets in terms of word count and vocabulary size for the two parts of speech that we consider. It also shows the inter-annotator agreement as a weighted Kappa score [13]. Since annotators could assign more than one label to a sentence (in practice, about 25% of the time) we use a set-distance metric [14] to count partial agreement.

Table 3. Vocabularies of the three data sets based on WordNet's morphology function, and Kappa agreement score between the coders

Data set	Tokens	Nouns	Verbs	Kappa score
Near East 1	3888	734	466	0.55
Near East 2	5524	976	614	0.50
Medieval	6856	1093	652	0.56

Two annotators working independently considered each sentence in the texts, and applied one or more functional semantic labels to characterize its function with respect to the image it describes. These labels, and their usage counts by the two coders, are shown in Table 4. Because of data sparseness for the other labels, our results only consider *Image Content*, *Historical Context* and *Implementation*.

Table 4. Functional semantic labels with each annotator's usage in all three texts

Label	Coder A	Coder B
Image Content	220	215
Historical Context	123	156
Implementation	75	138
Significance	59	51
Interpretation	67	26
Comparison	26	32
Biographic	10	6

5 Methodology

Figure 1 shows an overview of the experimental procedure. It begins with separate data sets for training and testing, a functional semantic label L to consider, a part-of-speech P to use for clustering, the number of clusters N to generate, and a similarity metric M. The similarity metric maps two WordNet senses to a real number between 0 and 1.

The training sentences are tokenized and lemmatised by WordNet's *morphy* function, and the set of unique lemmas of the specified part-of-speech is extracted. A matrix of the pairwise similarities is constructed using the specified

1. Input
   - Labeled data sets $TRAIN$ and $TEST$
   - Functional semantic label L
   - Part of speech $P = noun|verb$
   - Cluster count N
   - Similarity metric $M : sense, sense- > real$
2. Build clustering model on training data
   - Find all U unique lemmas of type P in data using WordNet's *morphy* function
   - Construct UxU matrix M where M_{xy} is the similarity between U_x and U_y
   - Use matrix to train expectation maximization clusterer with a target cluster count N
3. Build document representation of training and testing data
   - For each lemma l in the document, apply model the vector $[M(U_1, l), M(U_2, l) \ldots M(U_{|U|}, l)]$
   - Features are the frequency of each cluster $1 \ldots N$
4. Train SVM binary classifier for label L on training data
5. Evaluate classification performance on testing data

Fig. 1. Overview of semantic clustering

metric. An expectation maximization clusterer is then trained on the matrix, with the specified target cluster count. The output is a clustering model that can assign an unseen word to a cluster based on its similarities to each of the training lemmas.

Document representations are then built in the same manner for the training and testing data. Each lemma in the document is clustered according to its similarities to the training lemmas. Where the bag-of-words representation records lemma frequencies, the clustering representation records cluster frequencies. These are computed by applying the clustering model from the previous step to each lemma in the document. Training and testing of an SVM classifier for the label L is then performed using the two document representations.

Semantic similarity metrics operate on unambiguous senses, and so we needed to address polysemy in the texts. We tested two simple policies when calculating the similarity between two tokens. The *first sense* policy simply takes the first WordNet sense for each token, which is its most frequent sense in WordNet's sample texts [15]. The *closest sense* policy chooses the pair of senses that maximized the similarity calculation for the pair of tokens. Both methods have drawbacks: our data comes from a specific academic field with its own vocabulary, and WordNet's sample texts may be too general (e.g. the most common general sense of "profile" is not its art-historical meaning). The *closest sense* policy is more complicated, but an interesting experimental question in itself: it may use different senses for the same token depending on the token it is being compared to. Depending on the vocabulary, it might be that a preponderance of a semantic class will overcome this noise. This is discussed by Scott et al. [16], where the *closest sense* approach is also used.

The three similarity metrics we tested all base their values on positions within WordNet's hyper/hyponym ontology of senses, taking two senses (S_1 **and** S_2) as

input and returning a real number. The simplest metric, *path similarity*, uses the shortest *distance* between the two senses. *Leacock Chodorow similarity* [17] also takes into account the maximum *depth* of the ontology being used. *Wu-Palmer similarity* [18] is the most sophisticated metric we evaluated. It considers the most specific ancestor (least common subsumer or *LCS*) of the two senses, its depth in the ontology, and its shortest *distance* to each of the two senses.

WordNet has separate ontologies for verbs and nouns, and we tested the clustering method independently for both. The results indicate distinctive properties of the two ontologies. Miller et al. [15] discuss fundamental differences in the idea of "hyponymy" as applied to nouns versus verbs.

We varied the target number of clusters from 5 to 100 at intervals of 5. In principle, the maximum number of clusters that could be aimed for is the vocabulary size itself, in which each lemma would become its own cluster. Our results indicate that our 100-cluster upper bound may have been too conservative for at least one of the experiments (see figure 2, top right).

Expectation maximization [19] is an iterative algorithm often used for data clustering. It associates each data instance with a probability distribution describing potential cluster membership. Iteration ends when improvement falls below some small threshold (we use 1e-6) or the number of iterations passes some maximum value (we use 100). Our results here simply map each lemma (data instance) to its likeliest cluster. In future work we may use the full probability distribution. The computation is performed using the Weka [20] implementation of expectation maximization clustering.

The support vector machine is a machine learning algorithm that has achieved widespread success on text classification tasks. It divides the training data by an N-1-dimensional plane, where N is the dimensionality of the feature representations. This division is optimized to achieve the maximum separation between the training instances. We use an extended version that handles the typical situation where the training data is not linearly separable, by penalizing misclassified instances and optimizing the separation achieved. Its explicit design for achieving maximum generality makes it particularly attractive for our study. We use the Weka implementation of sequential minimal optimization [21], which is a method for efficiently solving the maximization problem, and a linear kernel.

Our classification results are presented as the area under the ROC (receiver operating characteristics) curve. This is particularly well-suited for evaluating classification tasks with sparse data and large skew in the label distribution [22] such as ours.

6 Results

6.1 Clustering Parameters

Figure 2 shows the average performance by number of clusters used, broken down by the part of speech used for the clusters and the functional semantic label targeted by the classifier. The most striking feature is the superior performance of the verb clusters.

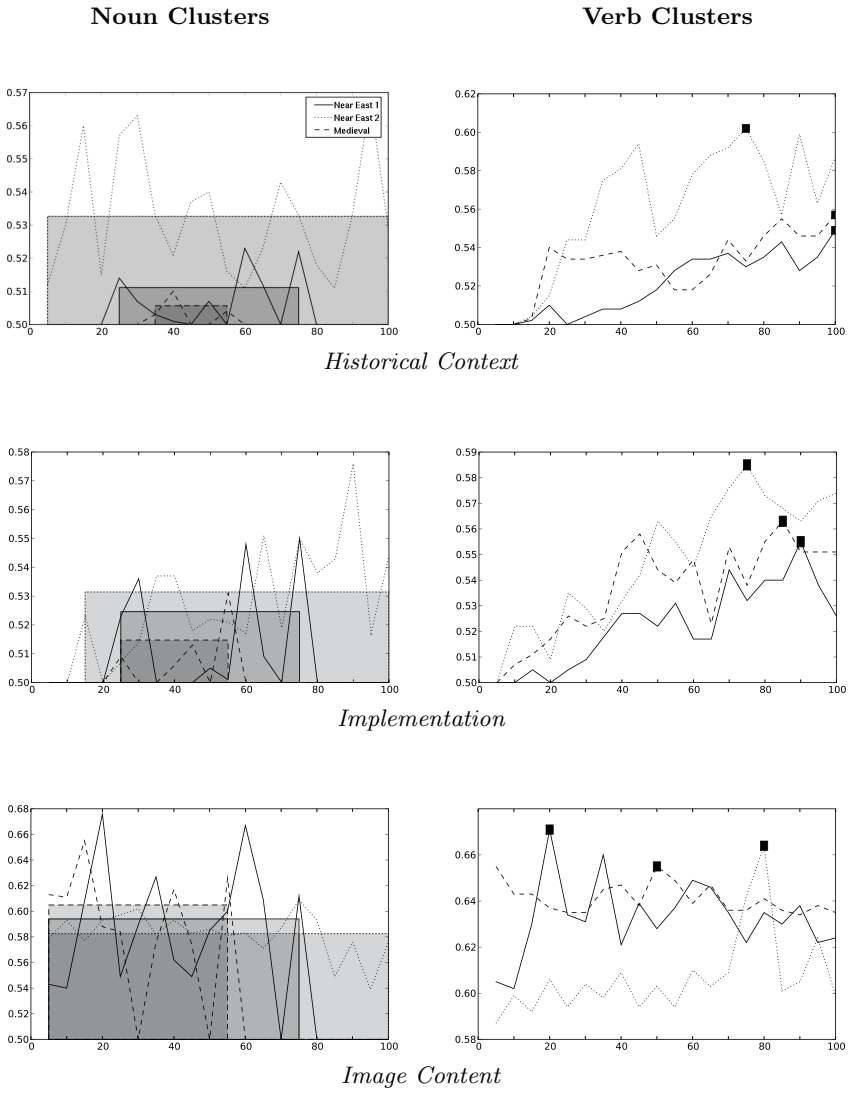


Fig. 2. Classifier performance (ROC average on Y-axis) by cluster count (X-axis), by part-of-speech for the three most common labels. The shaded regions in the left-hand figures cover the interval of cluster counts that had positive performance for each data set, and their height is the average performance over that interval. The black boxes in the right-hand figures indicate the best performance on the data set.

While the *Image Content* label shows the highest performance, it also shows the least regularity with respect to the cluster count parameter. Its performance is likely due to it being the easiest of the labels to learn, which has been noted in earlier work [1]. Its irregularity may also support the intuition that physical descriptors such as colors and dimensions are less tied to historical period.

The shaded areas in the noun cluster graphs (left side) each correspond to one of the three data sets. On the X-axis they highlight the interval from the smallest effective (performed above-random) cluster count to the largest. Their height represents the average performance across that interval. The labels all show the counter-intuitive property that the effective ranges for data sets with a larger vocabulary are always a subset of those with a smaller vocabulary. In other words, in choosing how many clusters to induce from a data set, there appears to be a narrower range of good choices for a larger vocabulary.

The black rectangles in the verb cluster graphs (right side) mark the highest performance for each data set. For the two labels that show the clearest trends in the verb clusters, *Historical Context* and *Implementation*, the data set with the smallest vocabulary peaks before the two larger data sets, supporting the intuition that a smaller vocabulary will perform best with fewer clusters. The regularity of the verb clusters, with steadily increasing performance compared to the erratic behavior of the noun clusters, lends more credibility to this observation. This distinction between verb and noun behavior must be confirmed on larger data sets before looking for an explanation in linguistic theory or resource-specific biases.

Several of the performance curves (particularly for the Near East 1 and Medieval data sets, *Historical Context* label, verb clusters) appear to be increasing at the upper limit cluster count (100). This indicates that the optimal cluster count is higher than we expected, and the testing range should be increased.

The three functional semantic labels show the same relationship when varying the sense choice iteration methods and similarity metrics (Figure 3). Better performance is achieved by simply using the first WordNet sense, rather than maximizing the similarity on a pairwise basis. The Wu-Palmer similarity metric

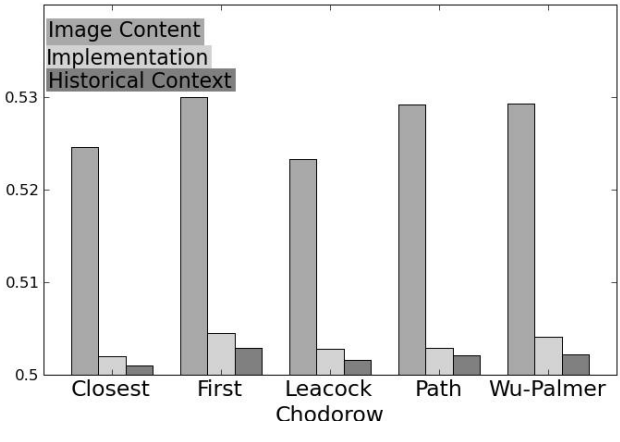


Fig. 3. Comparison of average performance above random (for ROC area, 0.5) by sense choice iteration method ("closest sense" and "first sense") and similarity metric. Each triplet shows the average performance for all runs with that parameter, broken down by the three labels.

outperforms the Path and Leacock Chodorow metrics. The differences are least pronounced on the *Image Content* label, which is the most domain-independent label and similar to a traditional "topical" classification. The standard deviations are massive compared to these differences, on the order of several times the magnitude: this may be due to the broad range of variables we tested, and requires further investigation.

6.2 Cluster Quality

The original motivation for the study was the observation that there were obvious word-groups recurring above the lexical level. It is natural, therefore, to want to examine a cluster with respect to some simple characterization of its members. It is not guaranteed that a cluster will have a simple concept that describes its members, except in the cases where the clustering process mirrors this human intuition.

The example in Figure 4 demonstrates an effective cluster (roughly identified as "human body parts") while also illustrating some shortcomings. All words in the figure are placed in the same cluster by the model, which was trained on the first Ancient Near East data set. Examining the training data manually, every recognizable body-part term is clustered appropriately, for perfect recall.

The benefit comes from the words that occur solely in the training or testing data: in a bag-of-words or statistical approach, these would have been useless features. But there are problems with the cluster's precision: words like "vessel", "ass" are examples of polysemy creating false positives (as in "blood vessel" and "posterior", when they are actually used as "drinking vessel" and "donkey"). "quick" is an example of clustering on an extremely rare usage of the word ("an area of the body highly sensitive to pain"), although as a reviewer pointed out this particular example would be addressed by initial part-of-speech tagging (but consider "orb"). There is also no strict guarantee of a simple characterization of

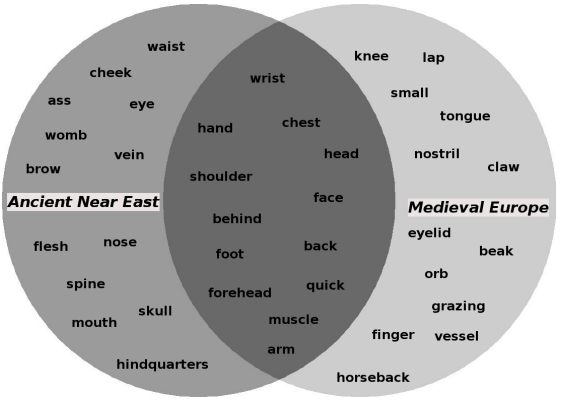


Fig. 4. Cluster example that roughly captures "human physiology"

the resulting clusters. "Human physiology", while very accurate in this case, is not precise: most people lack literal claws and beaks. This is because the clustering does not try to reconcile its results explicitly with the WordNet hierarchy. The cluster is in fact generally looking for "body parts", and the focus on human physiology is due to the texts in question. But this is exactly the point: classifiers trained on veterinary texts could use the same cluster on texts in human medicine.

The same experimental run also automatically induced high-quality clusters that roughly correspond to "quantities", "occupations", "building materials", "geographic regions" and so forth.

6.3 Performance

Table 5 compares the ROC areas of the bag-of-words-based classifiers with that of the *best*-performing cluster-based classifier. The clustering method outperforms bag-of-words 66% of the time when the train and test data come from different domains. This comparison assumes that we are able to choose the optimal parameters for building our clusters. Further investigation and optimization of the parameters discussed above with respect to the vocabulary of the training data is the key to realizing this.

Table 5. Comparison of ROC Area for the three functional semantic labels, when the train and test data come from different domains. Rows with a white background use bag-of-words features, rows with a grey background use the clustering features.

Data Sets		Label		
Train	Test	Historical Context	Implementation	Image Content
Near East 1	Medieval	0.576	0.518	0.576
		0.549	0.530	0.676
Near East 2	Medieval	0.514	0.521	0.578
		0.568	0.609	0.576
Medieval	Near East 2	0.573	0.564	0.597
		0.555	0.563	0.655
Medieval	Near East 1	0.541	0.538	0.603
		0.557	0.558	0.656

6.4 Improved Generality and Dimensionality

The semantic clusters show less sensitivity to training and testing domains than the bag-of-words features. Table 6 compares the standard deviation of classifier performance for the three labels for the basic bag-of-words model to the cluster model. The results also show the relationship between the domain-sensitivity of the functional semantic label and the improvements from the clustering method.

Comparing the original bag-of-words vocabularies of the data sets with the range of cluster counts we generated shows a reduction in document representation size between 10 and 100 times. The computational benefits of this reduction

Table 6. Standard deviation across different data set combinations

Label	Bag-of-Words	Clusters
Image Content	0.03279	0.02359
Historical Context	0.06657	0.02116
Implementation	0.03636	0.01903

are purchased at the cost of the off-line clustering process, which is resource-intensive but highly parallelizable. If it results in greater generality, it need only be done for a single domain in the given task.

7 Future Work

While space limitations prevented discussing it here, when considering any single configuration of the clustering parameters, the performance-cluster count graphs exhibit an unexpected periodic behavior. This is masked by the averages and maximums presented above, and it will require further investigation to determine what clustering parameters or characteristics of the data are responsible for this behavior.

There is an enormous amount of semantic and lexical information in WordNet that could be incorporated into the clustering process. At present, senses are associated via metrics that depend solely on directly mapping lemmas into the noun and verb hierarchies, which limits the resulting clusters in several ways (e.g. biased towards reproducing this structure, dropping information provided by adjectives, etc.). Evaluating and refining potential additions to the process (e.g. noun-adjective lexical relationships, metonymy, etc.) is a major future task.

The clustering model generated by the expectation maximization algorithm may be used more fully: rather than the simple approach of membership-frequency, the probability distribution could be used in generating the document representations. For example, in the current method, the sense "orange (the fruit)" might be counted part of a "fruit" cluster or an "edible object" cluster, but not both, even if it has near-equal probabilities of membership in either cluster. Crediting all clusters according to the lemma's probability of membership would capture this. At the other extreme, a simpler, discrete clustering algorithm like K-means, might be more appropriate (and less computationally intensive) for our current approach. Finally, it may be possible to use fully non-parametric clustering (i.e. without a specified number of clusters) to determine the optimal cluster size, but this is complicated by the fact that optimal cluster size in this case is not simply determined by the inter/intra-cluster coherence. It also depends on the utility of the resulting clusters for generalizing in the particular domains. For example, separate clusters for "grain", "legumes" and so forth might maximize intra-cluster coherence on some training data, but if the ideal semantic class is "agricultural products" these smaller clusters will be sub-optimal.

To address the affects of word sense ambiguity we have begun manual disambiguation on the three data sets, which will give us an upper limit on the improvements to expect from automatic approaches. Another possibility is using disambiguated corpora such as those used for SENSEVAL, which would also test the method on larger and more familiar data. This would require creating a functional semantic task, similar to the ANTArT project, using the new data set.

The reviewers suggested several methods and labeling tasks to compare our method with. A baseline usage of WordNet might be to use hypernym-frequency as the feature set. Sentiment analysis could be a useful source of well-explored labeling tasks that can be partitioned into distinct domains (e.g. by "product type") for training and testing. The size of the ANTArT data set was a limitation, and large product reviews databases (Amazon, IMDB, etc.) could help us understand the significance of the irregular behavior of noun clusters.

Finally, while our method presented here makes no use of distributional statistics or correlations of the words and labels, combining the approaches could raise clustering performance dramatically while maintaining generality. This could be particularly useful for deciding how many clusters to generate, e.g. if and how to subdivide broad clusters. For this study we performed exhaustive tests of cluster count to find trends related to the vocabulary of the data sets. It would be useful, when choosing to partition a set of words into one large cluster or two smaller clusters, to determine how the two potential choices would relate to label distribution in the training data.

8 Conclusion

We have presented a text classification task that responds poorly to the typical bag-of-words feature space, due to the nature of the data sets and labels involved. We described a method that builds a more general and compact document representation using measures of semantic similarity, and presented results testing several options for its component algorithms and parameters. We argued that the results show several of the desirable properties of the approach, and outlined future work in constructing an implementation that optimizes performance while maintaining these properties.

This approach has potential applications for any task that uses bag-of-words for document representation. Information retrieval could use clustering to expand open class terms or handle queries of a functional semantic nature. New multilingual WordNet implementations with pointers between senses could be used to automatically extend these benefits across languages. This document representation could also prove useful for investigating more abstract cognitive processes, such as analogy and inference, and for drawing comparisons between lexical resources and the cognitive structures they try to represent.

References

[1] Passonneau, R., Yano, T., Lippincott, T., Klavans, J.: Functional Semantic Categories for Art History Text: Human Labeling and Preliminary Machine Learning. In: International Conference on Computer Vision Theory and Applications, Workshop 3: Metadata Mining for Image Understanding (2008)

[2] Klavans, J., Sheffield, C., Abels, E., Bedouin, J., Jenemann, L., Lippincott, T., Lin, J., Passonneau, R., Sidhu, T., Soergel, D., Yano, T.: Computational Linguistics for Metadata Building: Aggregating Text Processing Technologies for Enhanced Image Access. In: OntoImage 2008: 2nd International Language Resources for Content-Based Image Retrieval Workshop (2008)

[3] Yano, T.: Experiments on Non-Topical Paragraph Classification of the Art History Textbook (unpublished) (2007)

[4] Pereira, F., Tishby, N., Lee, L.: Distributional clustering of English words. In: Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics, pp. 183–190 (1993)

[5] Tishby, N., Pereira, F.C., Bialek, W.: The information bottleneck method. In: Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing, pp. 368–377 (1999)

[6] Bekkerman, R., El-Yaniv, R., Tishby, N., Winter, Y.: On feature distributional clustering for text categorization. In: SIGIR 2001: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 146–153. ACM, New York (2001)

[7] Slonim, N., Tishby, N., Winter, Y.: Document clustering using word clusters via the information bottleneck method. In: ACM SIGIR 2000, pp. 208–215. ACM Press, New York (2000)

[8] Termier, R., Rousset, M.-c., Sebag, M.: Combining statistics and semantics for word and document clustering. In: Ontology Learning Workshop, IJCAI 2001, pp. 49–54 (2001)

[9] Resnik, P.: Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 448–453 (1995)

[10] Jiang, J., Conrath, D.: Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In: The 10th International Conference on Research in Computational Linguistics (1997)

[11] Lin, D.: An Information-Theoretic Definition of Similarity. In: Proceedings of the 15th International Conference on Machine Learning, pp. 296–304. Morgan Kaufmann, San Francisco (1998)

[12] Budanitsky, A., Hirst, G.: Evaluating WordNet-Based measures of semantic distance. Computational Linguistics 32(1), 13–47 (2006)

[13] Artstein, R., Poesio, M.: Inter-Coder Agreement for Computational Linguistics. Computational Linguistics 34(4), 555–596 (2008)

[14] Passonneau, R.J.: Measuring agreement on set-valued items (MASI) for semantic and pragmatic annotation. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC), Genoa, Italy (May 2006)

[15] Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: Introduction to WordNet: An On-line Lexical Database (1993)

[16] Scott, S., Matwin, S.: Text Classification using WordNet Hypernyms. In: Use of WordNet in Natural Language Processing Systems: Proceedings of the Conference, pages 38–44, pp. 45–52. Association for Computational Linguistics (1998)

[17] Leacock, Chodorow: Filling in a sparse training space for word sense identification (1994)
[18] Wu, Z., Palmer, M.: Verb semantics and lexical selection. In: 32nd Annual Meeting of the Association for Computational Linguistics (1994)
[19] Dempster, A.P., Laird, N.M., Diftler, M., Lovchik, C., Magruder, D., Rehnmark, F.: Maximum likelihood from incomplete data via the EM algorithm (1977)
[20] Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
[21] Platt, J.C.: Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, Advances in Kernel Methods - Support Vector Learning (1998)
[22] Fawcett, T.: ROC graphs: Notes and practical considerations for data mining researchers. Technical Report Tech report HPL-2003-4, HP Laboratories, Palo Alto, CA, USA (2003)

Reducing the Plagiarism Detection Search Space on the Basis of the Kullback-Leibler Distance

Alberto Barrón-Cedeño, Paolo Rosso, and José-Miguel Benedí

Department of Information Systems and Computation,
Universidad Politécnica de Valencia,
Valencia 46022, Camino de Vera s/n, Spain
{lbarron,prosso,jbenedi}@dsic.upv.es
http://www.dsic.upv.es/grupos/nle/

Abstract. Automatic plagiarism detection considering a reference corpus compares a suspicious text to a set of original documents in order to relate the plagiarised fragments to their potential source. Publications on this task often assume that the search space (the set of reference documents) is a narrow set where any search strategy will produce a good output in a short time. However, this is not always true. Reference corpora are often composed of a big set of original documents where a simple exhaustive search strategy becomes practically impossible.

Before carrying out an exhaustive search, it is necessary to reduce the search space, represented by the documents in the reference corpus, as much as possible. Our experiments with the METER corpus show that a previous search space reduction stage, based on the Kullback-Leibler symmetric distance, reduces the search process time dramatically. Additionally, it improves the Precision and Recall obtained by a search strategy based on the exhaustive comparison of word n-grams.

1 Introduction

The easy access to a wide range of information via electronic resources such as the Web, has favoured the increase of plagiarism cases. When talking about text, plagiarism means to use text written by other people (even adapting it by rewording, insertion or deletion), without any credit or citation. In fact, the reuse of self-written text is often considered as self-plagiarism.

Plagiarism detection with reference tries to find the source of the potentially plagiarised fragments from a suspicious document in a set of reference documents. Some techniques based on the exhaustive comparison of suspicious and original documents have already been developed. These techniques apply comparison of sentences [7], structure of documents [15] and entire documents [10]. Examples of the used comparison strategies are dot plot [15] and n-grams [10].

One of the main difficulties in this task is the great size of the search space, i.e., the reference documents. To our knowledge, this problem has not been studied deeply enough, neither there are published papers on this issue. Given a suspicious document, our current research is precisely oriented to the reduction

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 523–534, 2009.

of the search space. The proposed approach is based on the Kullback-Leibler distance, which has been previously applied to many applications, ranging from image retrieval [5] to document clustering [13]. The reduction of search space for plagiarism detection is a more specific case of clustering: instead of grouping a set of related documents, the task is to define a reduced set of reference documents containing texts with a high probability of being the source of the potentially plagiarised fragments in a suspicious document. The final objective is to relate potentially plagiarised sentences to their source. Our experiments show that a reduction of the search space based on the Kullback-Leibler distance improves processing time as well as the quality of the final results.

The rest of the paper is structured as follows. Section 2 gives an overview of plagiarism detection including some state-of-the-art approaches. Section 3 defines the proposed method for reducing the search space as well as it describes the exhaustive search strategy we have opted for. Section 4 gives a description of the corpus we have used for our experiments. Section 5 describes the experiments and the obtained results. Finally, Section 6 gives some conclusions and draws future work.

2 Plagiarism Detection Overview

In automatic plagiarism detection, a correct selection of text features in order to discriminate plagiarised from non-plagiarised documents is a key aspect. Clough [3] has delimited a set of features which can be used in order to find plagiarism cases such as changes in the vocabulary, amount of similarity among texts or frequency of words. This kind of features has produced different approaches to this task.

Intrinsic plagiarism analysis [11] is a different task from plagiarism detection with reference. It captures the style across a suspicious document in order to find fragments that are plagiarism candidates. This approach saves the cost of the search process, but it does not give any hint about the possible source of the potentially plagiarised text fragments.

In those cases where a reference corpus is considered, the search process has been based on different features. *Ferret* [10] considers text comparison based on word n-grams. The reference, as well as the suspicious text, is split into trigrams, composing two sets which are compared. The amount of common trigrams is considered in order to detect potential plagiarism cases. *PPChecker* [7] considers the sentence as the comparison unit in order to compare local similarity. It differentiates among exact copy of sentences, word insertion, word removal and rewording on the basis of a Wordnet-based word expansion process.

A major difficulty in this task is the dimension of the reference corpus D. Even assuming that D contains the source document of the plagiarised fragments in a suspicious text s, the search strategy must be efficient enough to accurately find it in a reasonable time. An exhaustive comparison of sentences, paragraphs or any other text chunk s_i in order to answer the question: *is there a chunk* $s_i \in s$ *included in a document of* D*?* could be impossible if D is very large.

The complexity of the comparison process is $O(n \cdot m)$, where n and m are the lengths of s and D in fragments. Some efforts have already spend to improve the search speed, such as fingerprinting [16]. In this case a numerical value (fingerprint), which becomes the comparison unit, is assigned to each text chunk of the reference as well as the suspicious text. However, in this case each suspicious document is still compared to the entire reference corpus. In [15] a structural comparison of documents in order to reduce the search space is performed. Unfortunately, this method requires reference and suspicious documents written in LaTeX.

3 Method Definition

Given a reference corpus of original documents D and a plagiarism suspicious document s, our efforts are oriented to efficiently localise the subset of documents D' such that $|D'| \ll |D|$. The subset D' is supposed to contain those documents d with the highest probabilities of including the source of the plagiarised text fragments in s. After obtaining this subset, an exhaustive search of the suspicious sentences in s over D' can be performed. Our search space reduction method, the main contribution of this work, is based on the Kullback-Leibler symmetric distance.

3.1 The Kullback-Leibler Distance

The proposed search space reduction process is based on the Kullback-Leibler distance, which has shown good results in text clustering [2,13]. In 1951 Kullback and Leibler proposed the after known as *Kullback-Leibler divergence* (KL_d) [8], also known as cross-entropy. Given an event space, KL_d is defined as in Eq. 1. Over a feature vector $\mathcal{X}$, KL_d measures the difference (or equality) of two probability distributions P and Q.

$$KL_d(P \mid\mid Q) = \sum_{x \in \mathcal{X}} P(x) log \frac{P(x)}{Q(x)} \ . \tag{1}$$

KL_d is not a symmetric measure, i.e., $KL_d(P \mid\mid Q) \neq KL_d(Q \mid\mid P)$. Due to this fact, Kullback and Leibler (and also some other authors) have proposed symmetric versions of KL_d, known as Kullback-Leibler symmetric distance (KL_δ). Among the different versions of this measure, we can include:

$$KL_\delta(P \mid\mid Q) = KL_d(P \mid\mid Q) + KL_d(Q \mid\mid P) \ , \tag{2}$$

$$KL_\delta(P \mid\mid Q) = \sum_{x \in \mathcal{X}} (P(x) - Q(x)) log \frac{P(x)}{Q(x)} \ , \tag{3}$$

$$KL_\delta(P \mid\mid Q) = \frac{1}{2} \left[KL_d \left(P \mid\mid \frac{P+Q}{2} \right) + KL_d \left(Q \mid\mid \frac{P+Q}{2} \right) \right] \ , \tag{4}$$

$$KL_\delta(P \mid\mid Q) = \max(KL_d(P \mid\mid Q), KL_d(Q \mid\mid P)) \ . \tag{5}$$

The equations correspond respectively to the versions of Kullback and Leibler [8], Bigi [2], Jensen [6] and Bennet [1]. A comparison of these versions showed that there is no significant difference in the obtained results [13]. We use Eq. 3 due to the fact that it only implies an adaptation of Eq. 1 with an additional subtraction. The other three options perform a double calculation of KL_d, which is computationally more expensive.

Given a reference corpus D and a suspicious document s we calculate the KL_δ of the probability distribution P_d with respect to Q_s (one distance for each document $d \in D$), in order to define a reduced set of reference documents D'. These probability distributions are composed of a set of features characterising d and s (Subsections 3.2 and 3.3). An exhaustive search process (Subsection 3.4) can then be applied on the reduced set D' instead of the entire corpus D.

3.2 Features Selection

A feature selection must be made in order to define the probability distributions P_d. We have considered the following alternative techniques:

1. *tf* (term frequency). The relevance of the i-th term t_i in the j-th document d_j is proportional to the frequency of t_i in d_j. It is defined as:

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad , \tag{6}$$

   where $n_{i,j}$ is the frequency of the term t_i in d_j and is normalised by the frequency of all the terms t_k in d_j.
2. *tfidf* (term frequency-inverse document frequency). The weight *tf* of a term t_i is limited by its frequency in the entire corpus. It is calculated as:

$$tfidf_{i,j} = tf_{i,j} \cdot idf_i = tf_{i,j} \cdot log \frac{|D|}{|\{d_j \mid t_i \in d_j\}|} \quad , \tag{7}$$

   where $|D|$ is the number of documents in the reference corpus and $|\{d_j \mid t_i \in d_j\}|$ is the number of documents in D containing t_i.
3. *tp* (transition point). The transition point tp^* is obtained by the next equation:

$$tp^* = \frac{\sqrt{8 \cdot I_1 + 1} - 1}{2} \quad . \tag{8}$$

   I_1 is the number of terms t_k appearing once in d_j [12]. In order to give more relevance to those terms around tp^*, the final term weights are calculated as:

$$tp_{i,j} = (\langle tp^* - f(t_i, d_j) \rangle + 1)^{-1} \quad , \tag{9}$$

   where, in order to guarantee positive values, $\langle \cdot \rangle$ is the absolute value function.

The aim of the feature selection process is to create a ranked list of terms. Each probability distribution P_d is composed of the top terms in the obtained list, which are supposed to better characterise the document d. We have experimented with $[10, \cdots, 90]\%$ of the terms with the highest $\{tf, tfidf, tp\}$ value in d (Section 5).

3.3 Term Weighting

The probability (weight) of each term included in P_d is simply calculated by Eq. 6, i.e., $P(t_i, d) = tf_{i,d}$. These probability distributions are independent of any other reference or suspicious document and must be calculated only once.

Given a suspicious document s, a preliminary probability distribution Q'_s is obtained by the same weighting schema, i.e., $Q'(t_i, s) = tf_{i,s}$. However, when comparing s to each $d \in D$, in order to determine if d is a source candidate of the potentially plagiarised sections in s, Q'_s must be adapted.

The reason is that the vocabulary in both documents will be different in most cases. Calculating the KL_δ of this kind of distributions could result in an infinite distance ($KL_\delta(P_d \mid\mid Q'_s) = \infty$), when a t_i exists such that $t_i \in d$ and $t_i \notin s$. The probability distribution Q_s does depend on each P_d. In order to avoid infinite distances, Q_s and P_d must be composed of the same terms. If $t_i \in P_d \cap Q'_s$, $Q(t_i, s)$ is smoothed from $Q'(t_i, s)$; if $t_i \in P_d \setminus Q'_s$, $Q(t_i, s) = \epsilon$. This is simply a back-off smoothing of Q. In agreement with [2], the probability $Q(t_i, s)$ will be:

$$Q(t_i, s) = \begin{cases} \gamma \cdot Q'(t_i \mid s) & \text{if } t_i \text{ occurs in } P_d \cap Q'_s \\ \epsilon & \text{if } t_i \text{ occurs in } P_d \setminus Q'_s \end{cases} . \tag{10}$$

Note that terms occurring in s but not in d are not relevant. γ is a normalisation coefficient estimated by:

$$\gamma = 1 - \sum_{t_i \in d, t_i \notin s} \epsilon \ , \tag{11}$$

respecting the condition:

$$\sum_{t_i \in s} \gamma \cdot Q'(t_i, s) + \sum_{t_i \in d, t_i \notin s} \epsilon = 1 \ . \tag{12}$$

ϵ is smaller than the minimum probability of a term in a document d. After calculating $KL_\delta(P_d \mid\mid Q_s)$ for all $d \in D$, it is possible to define a subset of source documents D' of the potentially plagiarised fragments in s. We define D' as the ten reference documents d with the lowest KL_δ with respect to s.

3.4 Exhaustive Search Process

Once the reference subcorpus D' has been obtained, the aim is to answer the question "*Is a sentence $s_i \in s$ plagiarised from a document $d \in D'$?*". Plagiarised text fragments use to appear mixed and modified. Moreover, a plagiarised sentence could be a combination of various source sentences. Due to these facts, comparing entire documents (and even entire sentences) could not give satisfactory results.

In order to have a flexible search strategy, we codify suspicious sentences and reference documents as word n-grams (reference documents are not split into sentences). It has been shown previously that two independent texts have a low level of matching n-grams (considering $n > 1$). Additionally, codifying texts in this

way does not decrease the representation level of the documents. In the particular case of [10], this has been shown for trigrams. In order to determine if the i-th sentence s_i is plagiarised from d, we compare the corresponding sets of n-grams. Due to the difference in the size of these sets, we carry out an asymmetric comparison on the basis of the *containment* measure [9], a value in the range of $[0, 1]$:

$$C(s_i \mid d) = \frac{|N(s_i) \cap N(d)|}{|N(s_i)|} \quad , \tag{13}$$

where $N(\cdot)$ is the set of n-grams in $\cdot$.

Once every document d has been considered, s_i becomes a candidate of being plagiarised from d if the maximum $C(s_i \mid d)$ is greater than a given threshold.

4 The Corpus

In our experiments, we have used the *XML* version of the *METER corpus* [4]. This corpus was created as part of the METER (MEasuring TExt Reuse) Project.[1] The METER corpus is composed of a set of news reported by the Press Association (PA). These news were distributed to nine British newspapers (The Times, The Guardian, Independent and The Telegraph, among others), which can use them as a source for their own publications.

For experimental purposes, we have considered 771 PA notes as the original documents, which is the entire set of PA notes in this corpus version. The corpus of suspicious documents is composed of 444 newspaper notes. We selected this subset due to the fact that the fragments in their sentences are identified as *verbatim*, *rewrite* or *new*. These labels mean that the fragment is copied, rewritten or completely independent from the PA note, respectively.

Verbatim and rewritten fragments are triggers of a plagiarised sentence s_i. s_i is considered plagiarised if it fulfils the inequality $|s_{i_v} \cup s_{i_r}| > 0.4|s_i|$, where $|\cdot|$ is the length of $\cdot$ in words whereas s_{i_v} and s_{i_r} are the words in verbatim and rewritten fragments, respectively. This condition avoids erroneously to consider sentences with named entities and other common chunks as plagiarised. Some statistics about the reference and suspicious corpora are included in Table 1. The

Table 1. Statistics of the corpus used in our experiments

Feature	**Value**
Reference corpus size (kb)	1,311
Number of PA notes	771
Tokens / Types	226k / 25k
Suspicious corpus size (kb)	828
Number of newspapers notes	444
Tokens / Types	139k / 19k
Entire corpus tokens	366k
Entire corpus types	33k

[1] http://www.dcs.shef.ac.uk/nlp/meter/

pre-processing for both reference and suspicious documents consists of word-punctuation splitting ($w, \rightarrow [w][,]$) and a Porter stemming process [14].[2]

5 Experiments

As we have pointed out, the aim of the proposed method is to reduce the search space before carrying out an exhaustive search of suspicious sentences across the reference documents.

Once P_d is obtained for every document $d \in D$, the entire search process is the one described in Fig. 1.

Algorithm 1: Given the reference corpus D and a suspicious document s:

// Distance calculations
Calculate $Q'_s(t_k) = tf_{k,s}$ for all $t_k \in s$
For each document d in the reference corpus D
    Define the probability distribution Q_s given P_d
    Calculate $KL_\delta(P_d \mid\mid Q_s)$
// Definition of the reduced reference corpus
$D' = \{d\}$ such that $KL_\delta(P_d \mid\mid Q_s)$ is one of the 10 lowest distance measures
$n_{s_i} = [n\text{-grams in } s_i]$ for all $s_i \in s$
// Exhaustive search
For each document d in the reduced reference corpus D'
    $n_d = [n\text{-grams in } d]$
    For each sentence s_i in s
        Calculate $C(n_{s_i} \mid n_d)$
    If $\max_{d \in D'}(C(n_{s_i} \mid n_d)) \geq Threshold$
        s_i is a candidate of being plagiarised from $\arg\max_{d \in D'}(C(n_{s_i} \mid n_d))$

Fig. 1. Plagiarism detection search process

We have carried out three experiments in order to compare the speed (in terms of seconds), and quality of the results (in terms of Precision, Recall and F-measure), of the plagiarism detection process with and without search space reduction. The experiments explore four main parameters:

1. Length of the terms composing the probability distributions: $l = \{1, 2, 3\}$
2. Feature selection technique: tf, $tfidf$ and tp
3. Percentage of terms in d considered in order to define P_d: $[10, \cdots, 90]\%$
4. Length of the n-grams for the exhaustive search process: $n = \{1, 2, \cdots, 5\}$

In the first and second experiments, we carried out a 5-fold cross validation. The objective of our first experiment was to define the best values for the first

[2] We have used the Vivake Gupta implementation of the Porter stemmer, which is available at http://tartarus.org/~martin/PorterStemmer/

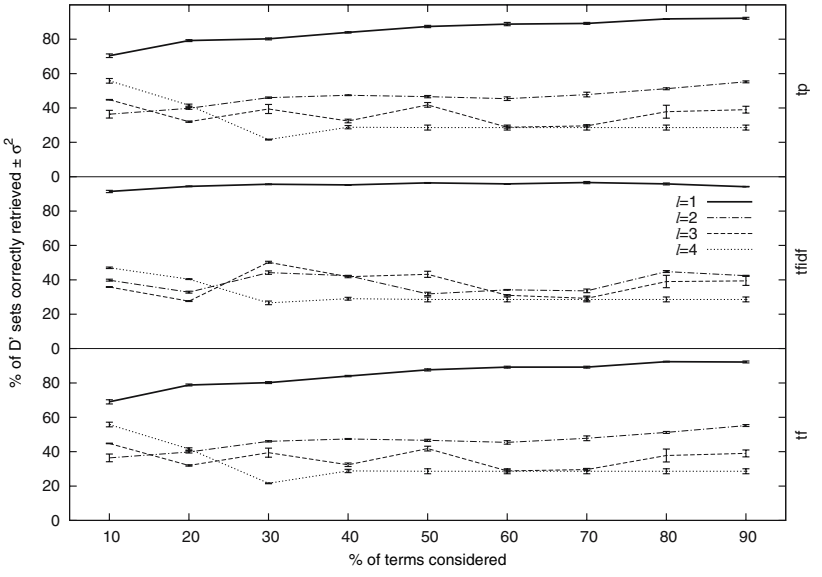


Fig. 2. Evaluation of the search space reduction process. Percentage of sets correctly retrieved ($\{tf, tfidf, tp\}$ = feature extraction techniques, l = term length).

three parameters of the search space reduction process. Given a suspicious document (newspaper) s we consider that D' has been correctly retrieved if it includes the source document (PA) of s. Figure 2 contains the percentage of sets correctly retrieved in the experiments carried out over the different development sets. In the five cases the results were practically the same.

The best results for the three feature selection techniques are obtained when unigrams are used. Higher n-gram levels produce probability distributions where a good part of the terms has a weight near to 1. These distributions (where almost all the terms have the same probability) do not allow KL_δ to determine how close are two documents. Regarding the feature selection techniques, considering tf does not give good results. In this case a good number of functional words (prepositions and articles, for example), which are unable to characterise a document, are considered in the corresponding probability distributions. The results obtained by considering tp are close to those of tf. Considering mid-terms (which tries to discard functional words), seems not to characterise either this kind of documents because they are too noisy. The results with this technique could be better with longer documents. The best results in this case are obtained with $tfidf$. Functional and other kinds of words that do not characterise the document are eliminated from the considered terms and the probability distributions characterise correctly the reference (and after the suspicious) document.

Regarding the length of the probability distributions, the quality of the retrieval is practically constant when considering $tfidf$ with unigrams. The only real improvement is achieved when considering 20% of the document vocabulary;

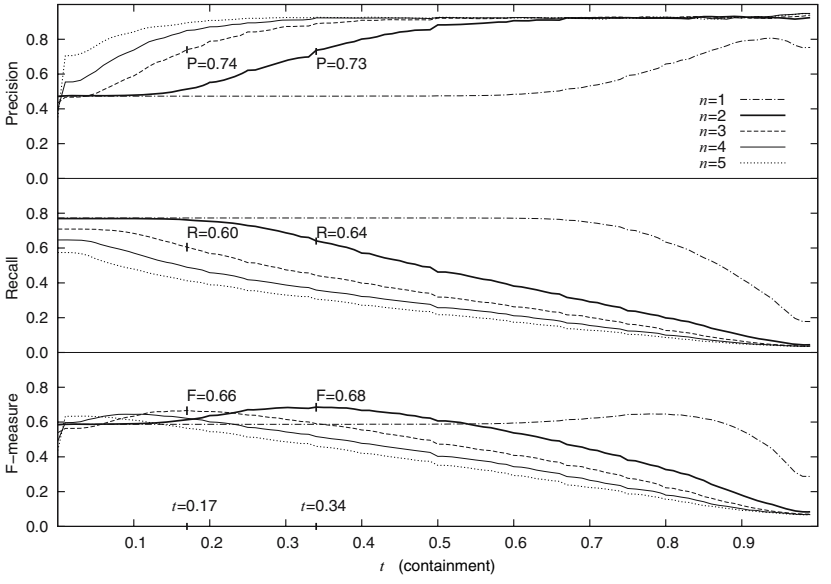


Fig. 3. Exhaustive search process evaluation. (n = n-gram level, t =threshold).

the percentage of correctly retrieved documents increases from 91% to 94% when 10% and 20% of the vocabulary is considered. The best option is to consider the 20% of the vocabulary in order to compose the probability distributions of the reference documents. In this way we obtain a good percentage of correctly retrieved reference documents with a sufficiently low dimension for the probability distributions. When applying the best parameters over the corresponding test sets, the obtained results did not show significant variations.

The second experiment aims to explore the fourth parameter (on the exhaustive search process). The containment threshold was varied in order to decide whether a suspicious sentence was plagiarised or not. Precision, Recall and F-measure were estimated by considering the five development sets of suspicious documents. Figure 3 shows the results obtained with n in the range $[1, 5]$ over the entire reference corpus D.

The text codification, based on a simple bag of words ($n = 1$), does not consider context information and style. This results in a good Recall (practically constant until $threshold = 0.7$). However, the probability of a reference document of containing the entire vocabulary of a suspicious sentence s_i is too high. Due to this reason, Precision is the lowest among all the experiments. On the other side, considering n-grams of level 4 (and higher) produces a rigid search strategy. Minor changes in the plagiarised sentences avoid their detection, resulting in the lowest Recall values.

The best results are obtained when considering bigrams and trigrams (best F-measures are 0.68 and 0.66, respectively). In both cases, the word n-grams are short enough to handle modifications in the plagiarised fragments as well as long

Table 2. Results comparison: exhaustive search versus space reduction + exhaustive search. (P =Precision, R =Recall, F = F-measure, t = avg. processing time (secs.))

Experiment	threshold	P	R	F	t
Without space reduction	0.34	0.73	0.63	0.68	2.32
With space reduction	**0.25**	**0.77**	**0.74**	**0.75**	**0.19**

enough to compose strings with a low probability of appearing in any (but the plagiarism source) text. Trigram-based search is more rigid, resulting in better Precision. Bigram-based search is more flexible, allowing a better Recall. The difference is reflected in the threshold in which the best F-measure values are obtained for both cases: 0.34 for bigrams versus 0.17 for trigrams. The threshold with the best F-measure t^* was after applied to the corresponding test set. The obtained results were exactly the same ones obtained during the estimation, confirming that $t^* = \{0.34, 0.17\}$ is a good threshold value for bigrams and trigrams, respectively.

The third experiment shows the improvement obtained by carrying out the reduction process in terms of speed and quality of the output. Table 2 shows the results obtained by bigrams when s is searched over D as well as D', i.e., the original and the reduced reference corpora. In the first case, we calculate the containment of $s_i \in s$ over the documents of the entire reference corpus D. Although this technique by itself obtains good results, considering too many reference documents that are unrelated to the suspicious one, produces noise in the output, affecting Precision and Recall. An important improvement is obtained when $s_i \in s$ is searched over D', after the search space reduction process.

With respect to the processing time, the average time needed by the method to analyse a suspicious document s over the entire reference corpus D is about 2.32 seconds, whereas the entire process of search space reduction and the analysis of the document s over the reduced subset D' needs only 0.19 seconds[3]. This big time difference is due to three main factors: (1) P_d is pre-calculated for every reference document, (2) $Q'(s)$ is calculated once and simply adapted to define each Q_s given P_d, and (3) instead of searching the sentences of s in D, they are searched in D', which only contains 10 documents.

With respect to the output quality, Precision and Recall become higher when the search space reduction is carried out. Moreover, this result is obtained considering a lower containment threshold. The reason for this behaviour is simple: when we compare s to the entire corpus, each s_i is compared to many documents that are not even related to the topic of s, but contain common n-grams. Note that deleting those n-grams composed of "common words" is not a solution due to the fact that they contain relevant information about the writing style. The reduction of the threshold level is due to the same reason: less noisy comparisons are made and plagiarised sentences that were not considered before are now taken into account.

[3] Our implementation in Python has been executed on a Linux PC with 3.8GB of RAM and a 1600 MHz processor.

6 Conclusions

In this paper we have investigated the impact of the application of a search space reduction process as the first stage of plagiarism detection with reference. We have additionally explored different n-grams levels for the exhaustive search subprocess, which is based on the search of suspicious sentences codified as n-grams over entire documents of the reference corpus. The obtained results have shown that bigrams as well as trigrams are the best comparison units. Bigrams are good to enhance Recall whereas trigrams are better to enhance Precision, obtaining an F-measure of 0.68 and 0.66 over the entire reference corpus, respectively.

The search space reduction method is the main contribution of this work. It is based on the Kullback-Leibler symmetric distance, which measures how closed two probability distributions are. The probability distributions contain a set of terms from the reference and suspicious documents. In order to compose them, term frequency, term frequency-inverse document frequency and transition point (tf, $tfidf$ and tp, respectively) have been used as feature selection techniques. The best results were obtained when the probability distributions were composed of word unigrams selected by $tfidf$.

In the experiments a comparison of the obtained results was made (also in terms of time performance) by carrying out the exhaustive search of n-grams over the entire as well as the reduced reference corpora. When the search space reduction was applied, the entire reference corpus (700 documents approximately) was reduced to only 10 reference documents. In this optimised condition, the plagiarism detection process needs on average only 0.19 seconds instead of 2.32. Moreover, the F-measure was improved (from 0.68 to 0.75 when using bigrams).

As future work we would like to consider a different measure from the Kullback-Leibler distance for the search space reduction process. Moreover, it would be interesting to carry out an exhaustive search process based on the fingerprinting technique (after the reduction process). Additionally, we would like to validate the obtained results in a bigger corpus composed of larger documents. Unfortunately we do not have knowledge about the existence of a corpus matching the required characteristics and creating one is by itself a hard task.

Acknowledgements. We would like to thank Paul Clough for providing us the METER corpus. This work was partially funded by the MCyT TIN2006-15265-C06-04 research project and the CONACyT-MEXICO 192021/302009 grant.

References

1. Bennett, C.H., Gács, P., Li, M., Vitányi, P.M., Zurek, W.H.: Information Distance. IEEE Transactions on Information Theory 44(4), 1407–1423 (1998)
2. Bigi, B.: Using Kullback-Leibler distance for text categorization. In: Sebastiani, F. (ed.) ECIR 2003. LNCS, vol. 2633, pp. 305–319. Springer, Heidelberg (2003)
3. Clough, P.: Plagiarism in Natural and Programming Languages: an Overview of Current Tools and Technologies. Research Memoranda: CS-00-05, Department of Computer Science. University of Sheffield, UK (2000)

4. Clough, P., Gaizauskas, R., Piao, S.: Building and Annotating a Corpus for the Study of Journalistic Text Reuse. In: 3rd International Conference on Language Resources and Evaluation (LREC 2002), Las Palmas, Spain, vol. V, pp. 1678–1691 (2002)
5. Do, M.N., Vetterli, M.: Texture Similarity Measurement Using Kullback-Leibler Distance on Wavelet Subbands. In: International Conference on Image Processing, vol. 3, pp. 730–733 (2000)
6. Fuglede, B., Topse, F.: Jensen-Shannon Divergence and Hilbert Space Embedding. In: IEEE International Symposium on Information Theory (2004)
7. Kang, N., Gelbukh, A., Han, S.-Y.: PPChecker: Plagiarism pattern checker in document copy detection. In: Sojka, P., Kopeček, I., Pala, K. (eds.) TSD 2006. LNCS (LNAI), vol. 4188, pp. 661–667. Springer, Heidelberg (2006)
8. Kullback, S., Leibler, R.A.: On Information and Sufficiency. Annals of Mathematical Statistics 22(1), 79–86 (1951)
9. Lyon, C., Malcolm, J., Dickerson, B.: Detecting Short Passages of Similar Text in Large Document Collections. In: Conference on Empirical Methods in Natural Language Processing, Pennsylvania, pp. 118–125 (2001)
10. Lyon, C., Barrett, R., Malcolm, J.: A Theoretical Basis to the Automated Detection of Copying Between Texts, and its Practical Implementation in the Ferret Plagiarism and Collusion Detector. In: Plagiarism: Prevention, Practice and Policies Conference, Newcastle, UK (2004)
11. Meyer zu Eissen, S., Stein, B.: Intrinsic plagiarism detection. In: Lalmas, M., MacFarlane, A., Rüger, S.M., Tombros, A., Tsikrika, T., Yavlinsky, A. (eds.) ECIR 2006. LNCS, vol. 3936, pp. 565–569. Springer, Heidelberg (2006)
12. 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)
13. 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)
14. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
15. Si, A., Leong, H.V., Lau, R.W.H.: CHECK: A Document Plagiarism Detection System. In: ACM Symposium on Applied Computing, CA, pp. 70–77 (1997)
16. Stein, B.: Principles of Hash-Based Text Retrieval. In: Clarke, Fuhr, Kando, Kraaij, de Vries (eds.) 30th Annual International ACM SIGIR Conference, pp. 527–534 (2007)

Empirical Paraphrasing of Modern Greek Text in Two Phases: An Application to Steganography

Katia Lida Kermanidis and Emmanouil Magkos

Ionian University, Department of Informatics
7 Pl. Tsirigoti, 49100, Corfu, Greece
{kerman,emagos}@ionio.gr

Abstract. This paper describes the application of paraphrasing to steganography, using Modern Greek text as the cover medium. Paraphrases are learned in two phases: a set of shallow empirical rules are applied to every input sentence, leading to an initial pool of paraphrases. The pool is then filtered through supervised learning techniques. The syntactic transformations are shallow and require minimal linguistic resources, allowing the methodology to be easily portable to other inflectional languages. A secret key shared between two communicating parties helps them agree on one chosen paraphrase, the presence of which (or not) represents a binary bit of hidden information. The ability to simultaneously apply more than one rules, and each rule more than one times, to an input sentence increases the paraphrase pool size, ensuring thereby steganographic security.

Keywords: paraphrasing, shallow parsing, supervised learning, steganography.

1 Introduction

Given an original sentence, that conveys a specific meaning, paraphrasing means expressing the same meaning using a different set of words or a different syntactic structure. Significant research effort has been put into the identification as well as the generation of paraphrases. Paraphrasing has been used extensively for educational purposes in language learning, as well as in several NLP tasks like text summarization [4], question answering [6] and natural language generation. Recently it has found yet another use in steganography.

Regarding paraphrase identification, previous approaches have utilized supervised [8] or unsupervised ([2][3]) machine learning techniques. The authors in [13] use named entity recognition in newswire articles from different newspapers to detect pairs of sentences that discuss the same topic and find sentence parts that are paraphrases. Regarding paraphrase generation, the use of finite state automata [10], i.e. paraphrase lattices, has been proposed, as well as the application of empirical rules [9] and statistical machine translation techniques [12].

In the present work, paraphrases of Modern Greek free text are learned in two phases. Henceforth, the term 'paraphrasing' will stand for shallow syntactic

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 535–546, 2009.

transformations, i.e. swaps of consecutive phrasal chunks. Modern Greek is suitable for shallow paraphrasing, due to the permissible freedom in the ordering of the phrases in a sentence. The rich morphology allows for more freedom in chunk ordering compared to other languages, more strict in syntax, like English or German.

A set of empirical rules is first applied to the input sentences in order to change their phrase ordering. The resulting paraphrases are then encoded into feature-value vectors and fed into supervised learning schemata so as to further filter out paraphrases that are syntactically incorrect and/or stylistically unnatural.

The paraphrase learning process is based on resource economy, i.e. the utilization of as minimal linguistic resources as possible, enabling thereby the methodology to be easily applicable to other morphologically rich languages. Regarding pre-processing tools, a phrase chunker is used for splitting the sentence into chunks. No use of other external thesauri, treebanks, lexica or other sophisticated tools of any kind is made.

The paraphrased text may then be used for hiding secret information, i.e. steganography. Steganographic security will depend on the correctness and the naturalness of the paraphrases. The supervised filtering phase ensures higher paraphrasing accuracy, which leads to higher security in protecting the hidden message from malicious eavesdroppers.

The rest of this paper is organized as follows. Section 2 describes the corpus that was used, as well as the chunking tool. Sections 3 and 4 are dedicated to the two phases of the paraphrase learning process. Section 5 introduces steganography through text media and describes the use of the produced paraphrases for hiding secret information. Section 6 presents the outcome of the paraphrase evaluation process and provides some interesting topics for discussion.

2 Corpus and Pre-processing

The text corpus used in the experiments is the ILSP/ELEFTHEROTYPIA corpus [7]. It consists of 5244 sentences; it is balanced in domain and genre, and manually annotated with complete morphological information. Further (phrase structure) information is obtained automatically by a multi-pass chunker [14].

During chunking, noun (NP), verb (VP), prepositional (PP), adverbial phrases (ADP) and conjunctions (CON) are detected via multi-pass parsing. The chunker exploits minimal linguistic resources: a keyword lexicon containing 450 keywords (i.e. closed-class words such as articles, prepositions etc.) and a suffix lexicon of 300 of the most common word suffixes in Modern Greek. The chunked phrases are non-overlapping. Embedded phrases are flatly split into distinct phrases. Nominal modifiers in the genitive case are included in the same phrase with the noun they modify; base nouns joined by a coordinating conjunction are grouped into one phrase. The chunker identifies basic phrase constructions during the first passes (e.g. adjective-nouns, article-nouns), and combines smaller phrases into longer ones in later passes (e.g. coordination, inclusion of genitive modifiers, compound phrases).

3 Phase 1: The Paraphrasing Rules

A set of nine paraphrasing rules for Modern Greek has been created. The rules have been developed empirically, they are bidirectional and they are grouped into two categories. The first consists of rules that swap the positions of two consecutive chunks; the second consists of rules that swap the positions of the two border chunks in a sequence of three consecutive chunks. The complete rule set is shown in Table 1.

The first rule is for subject-verb swapping. The second rule swaps a purpose or result clause with the preceding verb. Rules 3 and 4 swap phrases that participate in a coordinating structure, i.e. phrases that surround the conjunction *και* (and). Morphological (case) agreement between NPs and prepositional agreement between the PPs is obligatory in order for the swap to be admissible. The fifth rule takes advantage of the freedom in adverb positioning and swaps the adverb with the preceding phrase. Rules 6 to 8 swap certain types of PPs with their preceding verbs. Rule 9 swaps a copular verb with its predicate.

Unlike the syntactic tools presented in previous approaches [9], that may be applied only once to a given sentence, every rule described here may be applied multiple times (i.e. in multiple positions) to a sentence. Furthermore, more than one rules may be applied to a sentence simultaneously.

Each sentence is checked for rule applicability. More specifically, the rule set that may be applied to it is determined, as well as the number of times each of these rules is applicable. All possible combinations of rule applications are produced and the resulting sentences are henceforth called the *initial pool* of paraphrases. In the given corpus the size of the initial pool may vary from zero (the sentence does not allow for any paraphrasing) to as many as 80 paraphrases. The following example shows the initial pool of paraphrases for a corpus sentence.

```
NP[Στενοί της συνεργάτες] VP[είναι] ADP[επίσης] NP[η Π.
Καραβίτη] CON[και] NP[ο Π. Αντωνόπουλος].

([Close colleagues of hers] [are] [also] [P. Karaviti] [and] [P.
Antonopoulos])

Paraphrase 1 (Rule 1):
VP[Είναι] NP[στενοί της συνεργάτες] ADP[επίσης] NP[η Π.
Καραβίτη] CON[και] NP[ο Π. Αντωνόπουλος].

Paraphrase 2 (Rule 3):
NP[Στενοί της συνεργάτες] VP[είναι] ADP[επίσης] NP[ο Π.
Αντωνόπουλος] CON[και] NP[η Π. Καραβίτη].

Paraphrase 3 (Rule 5):
NP[Στενοί της συνεργάτες] ADP[επίσης] VP[είναι] NP[η Π.
Καραβίτη] CON[και] NP[ο Π. Αντωνόπουλος].

Paraphrase 4 (Rules 1 & 3):
VP[Είναι] NP[στενοί της συνεργάτες] ADP[επίσης] NP[ο Π.
Αντωνόπουλος] CON[και] NP[η Π. Καραβίτη].
```

```
Paraphrase 5 (Rules 3 & 5):
NP[Στενοί της συνεργάτες] ADP[επίσης] VP[είναι] NP[ο Π.
Αντωνόπουλος] CON[και] NP[η Π. Καραβίτη].
```

Table 1. The set of paraphrasing rules.

Rule	Example
1. NP(nom) VP → VP NP(nom)	[ο Γιάννης] [ήρθε] → [ήρθε] [ο Γιάννης] [John] [came] → [came] [John]
2. VP_1 VP_2(να) → VP_2(να) VP_1	[θέλει] [να παίξει] → [να παίξει] [θέλει] [he wants] [to play] → [to play] [he wants]
3. NP_1 και NP_2 → NP_2 και NP_1 (the 2 NPs are in the same case)	[η γιαγιά] και [ο παππούς] → [ο παππούς] και [η γιαγιά] [grandma] and [grandpa] → [grandpa] and [grandma]
4. PP_1 και PP_2 →PP_2 και PP_1 the 2 PPs start with the same preposition	[στην Γερμανία] και [στην Αγγλία] →[στην Αγγλία] και [στην Γερμανία] [in Germany] and [in England] → [in England] and [in Germany]
5. XP ADVP → ADVP XP XP: NP, PP or VP	[αποφασίστηκε] [εύκολα] → [εύκολα] [αποφασίστηκε] [it was decided] [easily] → [easily] [it was decided]
6. VP PP(σε) → PP(σε) VP	[κατέβηκε] [στο πάρκο] → [στο πάρκο] [κατέβηκε] [he went down] [to the park] → [to the park] [he went down]
7. VP PP(με) → PP(με) VP	[η νίκη] [επιτυγχάνεται] [με θυσίες] → [η νίκη] [με θυσίες] [επιτυγχάνεται] [victory] [is achieved] [with sacrifices] → [victory][with sacrifices][is achieved]
8. VP PP(για) → PP(για) VP	[αποφασίζει] [για τους άλλους] → [για τους άλλους] [αποφασίζει] [he decides] [for the others] → [for the others] [he decides]
9. VP(cp) NP(nom) → NP(nom) VP(cp)	[είναι] [έξυπνοι] → [έξυπνοι] [είναι] [they are] [clever] → [clever] [they are]

Rules 1 and 5 cannot be applied to the sentence simultaneously due to overlapping of the related phrases (simultaneous application of more than one rules requires that the phrases linked to the rules do not overlap), so the initial pool consists of five paraphrases.

The nine paraphrasing rules were applied to the 5244 corpus sentences. Figure 1 shows the distribution of the number of sentences depending on the sentence length (i.e. the number of chunks forming the sentence). Figure 2 shows the distribution of the number of sentences depending on the initial paraphrase pool size. Almost 80% of the sentences have at least one paraphrase, an impressive number, given that more than 24% of the input sentences consist of five or less chunks.

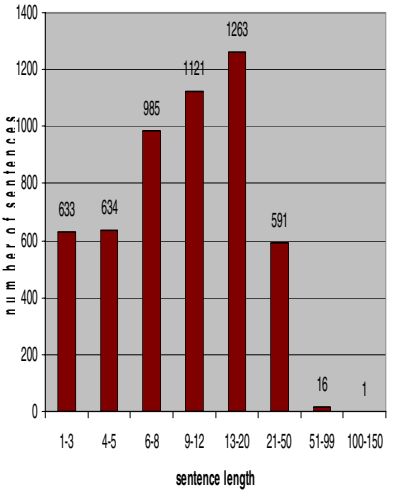


Fig. 1. Distribution of sentence length

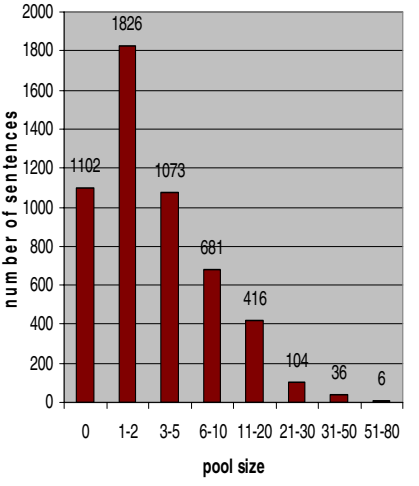


Fig. 2. Distribution of the pool size

4 Phase 2: Paraphrase Learning

Due to the use of the paraphrased sentences in steganography, correctness in syntax as well as naturalness are of great significance. Steganographic security depends largely on paraphrasing accuracy. Therefore the produced paraphrases are further filtered using supervised learning.

The positions of possible phrase swaps in the input sentences are identified, according to the paraphrasing rules. In the example in section 3, the swap positions are: 1 (between the first and the second phrase), 2 and 4.

A learning vector is created for every input sentence and each swap position. The features forming the vector encode syntactic information for the phrase right before the swap position, as well as two phrases to the left and two phrases to the right. Thereby, context information is taken into account. Every phrase is represented through a set of six features, shown in Table 2.

Table 2. The features of the learning vector

	1	2	3	4	5	6
NP	NP	case	morph	pron	gen	-
VP	VP	-	-	word	cop	-
PP	PP	-	-	prep	-	-
CON	CON	-	-	lemma	-	num
ADP	ADP	-	-	lemma	-	num

The first feature for every phrase is the phrase type. The case is one of three characters denoting the grammatical case of the headword in an NP (nominative, accusative or genitive). The headword is the noun in the nominative or accusative case. If there is no noun, it is the adjective in the nominative or the accusative. Else it

is the numeral or pronoun. If no element is present in either of these two cases, the headword is the first element of the phrase. Case is very important, as case agreement is decisive for the application of Rule 3, and the case value is significant for Rules 1 and 9. The *morph* feature is a three-letter code denoting whether an NP contains a definite or indefinite article and its grammatical case. The (in)definitiveness of an NP also affects the applicability of rule 3. The fourth feature varies according to the phrase type. For NPs it is the type of pronoun appearing in them, in case one does. It is the first word introducing a VP, the preposition introducing a PP, and the conjunction or the adverb in a CON and an ADP respectively. The presence of a genitive element is often decisive for the applicability of rules 1, 3 and 9. The fifth feature is binary and encodes whether there is an element in the genitive case in an NP, or a copular verb in a VP. Finally, the *num* feature is the number of tokens (words) within a CON or an ADP. This feature is very important, as a one-word coordinating CON phrase almost always allows a swap between the phrases it links, while a multi-word CON phrase very often does not (decisive for rules 3 and 4). In the following example the multi-word conjunction changes the meaning of the sentence in case of a swap.

NP[Την απόφαση] VP[θα πάρει] NP[η ΔΕΗ] CON[και όχι] NP[η κυβέρνηση].

(NP[The decision] VP[will be made by] NP[the National Power Company] CON[and not by] NP[the government].

The number of tokens constituting an ADP, as well as the adverb lemma, also affect phrase swapping. Rule 5 is safely applicable when the adverb expresses manner or mode, but cannot be applied if, for example, the adverb is relative like *όπως* (how), *όπου* (where), etc., or the phrase is formed by certain adverbial expressions.

To sum up, a total of 5 (phrases) x 6 (features/phrase) features constitute the feature vector, plus the binary target class: valid (yes) / not valid (no) paraphrase. Native speakers have manually annotated 519 instances (vectors), corresponding to 193 original sentences, with the correct class label. 26.4% of them were classified as incorrect paraphrases.

Several learning schemata have been experimented with for classification. The following table shows the prediction results for various stand-alone classification algorithms: decision trees (unpruned C4.5 tree), *k*-NN instance-based learning (k=5), support vector machines (first degree polynomial kernel function, sequential minimal optimization algorithm for training the classifier). Accuracy is the number of correctly classified instances divided by the total number of instances. Experiments were performed using 10-fold cross validation.

The majority of the incorrectly classified instances are negative (not valid), probably due to their rare occurrence in the data, compared to the positive instances.

Table 3. Results: stand-alone classifiers

	C4.5	k-NN	SVM
Accuracy	75.9%	77.9%	79.2%

Table 4. Results: ensemble learning

	Bagging	Boosting
Accuracy	80.3%	80.1%

Support vector machines cope better with predicting negative labels, and reach an f-score of 64.1% for the rare class.

To improve classification accuracy, ensemble learning schemata, like bagging and boosting, have also been experimented with. The C4.5 unpruned classifier was used as a base learner for bagging (the optimal bag size was 50% of the training set and 10 iterations were performed) and boosting (AdaBoost, again 10 iterations were performed). Bagging leads to the best f-score for the negative class: 65.3%.

5 Application to Steganography

Steganography is the art of embedding hidden information in unremarkable cover media in a way that does not arouse an eavesdropper's suspicion to the existence of hidden content underneath the surface message [11]. Hiding secret messages has always attracted the interest of communicating parties. Nowadays, information technologies take advantage of redundant information bits in the cover medium, and replace them with bits of the secret message, employing easy-to-use methodologies.

There are three important aspects to steganography: capacity, security and robustness. Capacity refers to the amount of information that can be hidden in the cover medium. The security level determines the (in)ability of an eavesdropper to 'wiretap' the hidden message, and robustness defines the amount of modification the cover medium can withstand without destroying the hidden information.

While several steganographic systems and methodologies have been developed for multimedia content, i.e. image, audio and video data [5], natural language (text) steganography is an emerging field [1]. Natural language steganography aims at hiding a message within a cover text by performing a set of linguistic alterations to the morphological, syntactic, or semantic structure of the text. Approaches have focused on three types of linguistic modifications: synonym substitution, syntactic and semantic transformations [15].

Synonym substitution replaces words in a sentence with their synonyms [15]. More specifically, the chosen synonyms are semantically more ambiguous than the initial words, i.e. they belong to several synonym sets (they may take several senses), making it harder for a third party (eavesdropper) to choose the correct sense.

Approaches performing syntactic transformations [9] modify the syntactic structure of a sentence (while preserving its original meaning). The plural number of syntactic structures that a sentence can appear in allows for the embedding of hidden information within the syntactic structure itself.

Semantic transformations identify noun phrases that refer to the same entity (co references). Upon co reference detection, a repeated noun phrase may be replaced by a referring expression (e.g. anaphora), or a pronoun referring to an aforementioned entity in a previous sentence may be replaced by the entity itself.

Though more widely employed, synonym substitution entails the danger of affecting the text semantics, sometimes altering its meaning. Moreover, synonym substitution presupposes the use of a word sense disambiguation tool that detects the correct sense of a word. Even if such a tool is available for a given language, its bounded performance will affect security.

5.1 The Final Pool of Paraphrases

A primary concern of the present work is the security of the steganographic process. The security level of the proposed system depends on the size of the final pool of paraphrases. If this size is non-negligible, then it is not trivial for an eavesdropper to detect the actual paraphrase that has taken place and break the system.

The positively labeled paraphrases from phase 2 constitute one part of the final pool of paraphrases. This part consists of paraphrases that have been produced by single phrase swaps, and not by combinations of swaps. The other part (due to the fact that the learning process does not allow for a paraphrase to be formed by combinations of phrase swaps) is formed by those paraphrases derived from phase 1 that are combinations of two or more correct phrase swaps (the positively labeled individual phrase swaps defined by phase 2, i.e. the first part of the final pool).

5.2 Message Embedding and Extracting

Once the final pool of paraphrases is formed for every sentence in the input (cover) text, the steganographic process starts. A secret message, i.e. a sequence of bits, is to be hidden underneath the cover text. The embedding process is completed in 3 stages.

First, every rule is marked with one bit value, depending on its condition. By condition we mean the right or the left-hand side of the rule (right or left-hand side of the arrow in Table 1). For example, for Rule 1 a bit value "0" could mean the left hand side of the rule, and then a bit value "1" would indicate its right-hand side. In the case of symmetrical rules (Rules 3 and 4), the condition may be determined, for example, by considering as NP_1 the noun phrase which starts with a letter closer to the beginning of the alphabet that NP_2. This rule marking results from a prior understanding between the communicating parties.

In the next stage, for every sentence, a paraphrase is selected from its final pool. The selection may be performed in a round-robin fashion (i.e. to choose the paraphrase of each rule one at a time), or based on a secret (e.g. a symmetric cryptographic) key shared between the two communicating parties. We leave the details of establishing such a key out of the scope of this paper. In case the size of the pool is zero, the sentence remains unchanged, and it is not used for information embedding. If, however, the pool size is greater than zero, a selection is possible and the sentence is useful for information embedding.

In the third stage, depending on the condition of the selected rule, a secret bit is embedded as follows: if the bit to be hidden is the same as the condition of the rule, the rule is not applied and the sentence remains unchanged, otherwise it is applied and the sentence is paraphrased. For example, a subject-verb sequence in the input sentence would mean a condition "0" for Rule 1. If the bit to be hidden is also "0", Rule 1 is not applied and the sentence is transmitted as it is. If the hidden bit were "1", the rule is applied and the sequence in the transmitted sentence now reads verb-subject, instead of subject-verb.

On the other end, the watermark extractor receives the final text. Having at his disposal the same rule set, (s)he is able to identify the rules that may be applied to each sentence. Sharing the same secret key used in the embedding process, (s)he is able to select the same rule used in the insertion process. In the previous example that

was Rule 1. Reading a subject-verb sequence, and knowing that this sequence indicates a bit value "0" for the condition of Rule 1, (s)he decides on "0" to be the first secret bit. Reading a verb-subject sequence would have meant a condition value "1" and (s)he would have decided on "1" to be the first secret bit.

5.3 An Example

Let the following three sentences constitute the initial message. The brackets indicate the chunk boundaries.

```
[Είσαι] [καλά];                    ([Are you] [well]?)        (A)
[Πήγα] [στην εκδρομή] [χτες].
([I went] [to the excursion] [yesterday].)                    (B)
[Να τα πούμε].                     ([Let's talk].)            (C)
```

The applicable rules for the given text are: For sentence A, rule 9, for sentence B rules 5 and 6, and for sentence C no rule. So the final pool of paraphrases is:

```
[Καλά] [είσαι];                    (A1)
[Πήγα][χτες][στην εκδρομή].        (B1 - after applying rule 5)
[Στην εκδρομή][πήγα][χτες].        (B2 - after applying rule 6)
```

Suppose that the hidden message is the bit sequence 10. For embedding the secret message, the rule condition of the first sentence is being checked. According to rule 9, a verb-adverb sequence corresponds to a condition of value 0. The fist hidden bit is 1. The two bits don't match, so rule 9 is applied, the paraphrase is taking place and the first sentence to be sent is A1.

Given the secret key, or in a round-robin fashion, the message sender decides on one of the two applicable rules for sentence B. Suppose that rule 6 is chosen. According to rule 6, a sequence of a verb and a PP introduced by the preposition *σε* corresponds to a condition of value 0. The next message bit to be embedded is 0. The two bits match, so rule 6 is not applied, and the second sentence is sent as it is. So the sent message is A1 B C.

The receiver gets this text. He looks at the rules to decide which one can have possibly been applied to sentence A1. It is only rule 9. In A1 (s)he detects the sequence adverb-verb. According to rule 9, this indicates a condition value 1. So, (s)he chooses 1, which is the first hidden bit. The applicable rules for the second sentence are 5 and 6. Using his/her secret key, (s)he chooses rule 6, the correct rule. According to this rule, a sequence of a verb and a PP introduced by the preposition *σε* corresponds to a rule condition 0. (S)he chooses 0, the second hidden bit.

5.4 Security Aspects

In the described process, security relies heavily on two factors. First, it is very important for the final text to be natural. Nothing should raise the eavesdropper's suspicion to any sort of hidden information underneath the surface text. Therefore, the final text must be grammatically correct, have the same meaning as the original text, and not present any anomalies (unnaturalness) in its stylistic properties.

A second important security aspect is the final pool size of the sentences. Even if the eaves-dropper does suspect the existence of hidden information, if the average pool size is large enough, it will be non-trivial to decide upon the correct paraphrase.

Regarding the first security aspect, a series of experiments was conducted to test the grammaticality and naturalness of the initial and the final pools of paraphrases. The experimental setup is described in detail in the following section. Regarding the second security aspect, Figure 2 proves that, unlike [9] (where reportedly a sentence can have at most seven paraphrases – one for every rule), the average pool size in this work is large enough to ensure inability in detecting the correct paraphrase for someone who is not familiar with the secret key.

In the final pool the average number of paraphrases per sentence is reduced (it still remains significantly higher than in [9]), affecting negatively the second security aspect and potentially steganographic capacity. The first security aspect, however, is strengthened considerably by the learning process of the second phase.

6 Paraphrasing Evaluation

Table 5 presents statistical information regarding rule applicability. The frequency column represents the applicability value for each rule (the number of times each rule is applicable in the corpus sentences) divided by the sum of the applicability values of all the rules. As can be seen, the subject-verb displacement (rule 1) and the adverb displacement (rule 5) rules constitute together around 70% of rule applications.

The initial and the final pool of paraphrases of the 193 sentences were checked for grammaticality and naturalness by two native language experts. Table 6 shows the effect of the paraphrases on the language experts. The first error rate indicates the percentage of rule applications that have forced the experts to make modifications in order for the paraphrases to become linguistically correct and natural within the initial pool. Modifications entail swaps in the ordering of the chunks. The second error rate is the same percentage for the final pool. Inter-expert agreement exceeded 94%.

Due to the automatic chunking process, neither the detection of chunk boundaries nor the detection of chunk types is error-free. Chunking errors affect paraphrasing performance. One significant type of chunking errors is excessive phrase cut-up. Looking up only the word suffix in the suffix lexicon, the chunker may assign an erroneous part-of-speech tag to the word, leading, thereby, to mistakes in the detection of phrase boundaries. For example, in the following sentence the word *πλήρες* (full) is erroneously tagged as a noun and not as an adjective, leading to false phrase splitting.

```
NP[Το πλήρες] NP[κείμενο της ανακοίνωσης]
(NP[The full] NP[text of the announcement])
```

As can be seen in Table 6, the coordination rules present the highest rate error (21.8%), mainly due to unnaturalness rather than ungrammaticality. For instance, in the following coordination example from the corpus, swapping the NPs would result

Table 5. Frequency of rule applicability

Rule	Frequency
1	0.3
2	0.08
3,4	0.07
5	0.38
6	0.025
7	0.028
8	0.02
9	0.1

Table 6. The rules' error rate

Rule	Error Rate 1	Error Rate 2
1	10.8%	7.2%
2	14.3%	13.3%
3,4	21.8%	14.1%
5	15.9%	11.9%
6	0%	0%
7	0%	0%
8	0%	0%
9	3.9%	3.1%
Avg	8.2%	5.5%

in a grammatically correct, but unnatural paraphrase, due to the presence of the genitive pronoun *του* (his). The feature *gen* helps focus on such cases.

```
NP[ο Μπρετόν] CON[και] NP[οι φίλοι του]
NP[Breton] CON[and] NP[his friends]
```

Given the low level of paraphrasing, the results of Table 6 are quite satisfactory, when compared to results of approaches that utilize more sophisticated resources, like [9], where an average error rate of 12.7% on the applied rules is reported, or the work in [12], where a minimum error rate of 10.5% is reported, when applying phrasal replacement to create paraphrases.

An interesting notion is the ability to simultaneously apply more than one rules to a sentence, or the same rule more than once. This allows for embedding more than one bits within a single sentence, increasing thereby steganographic capacity.

7 Conclusion

The application of shallow (chunk level) paraphrasing rules to Modern Greek sentences for steganographic purposes has been presented. Among the pool of paraphrases of a sentence, one paraphrase is chosen, in a manner that is se-cure and known only to the authorized communicating parties, and its presence can encode a secret message bit. The low paraphrasing level, as well as the absence of any kind of external linguistic resources, enables the easy portability of the methodology to other inflectional languages that are poor in resources. The large average size of the paraphrase pools, makes it non-trivial for an unauthorized party to detect the correct paraphrase. Security also depends on the correctness and naturalness of the paraphrase rules, which is quite satisfactory, taking into account the low-level linguistic processing. An interesting future direction of the current approach would be to take more advantage of the pool size in order to further increase the steganographic capacity of the input text.

References

1. Atallah, M., McDonough, C., Raskin, V., Nirenburg, S.: Natural Language Processing for Information Assurance and Security: An Overview and Implementations. In: Workshop on New Security Paradigms, Ballycotton, County Cork, Ireland, pp. 51–65 (2000)
2. Barzilay, R., McKeown, K.R.: Extracting Paraphrases from a Parallel Corpus. In: 39th Annual Meeting and the 10th Conference of the European Chapter of the Association for Computational Linguistics, Toulouse, France, pp. 50–57 (2001)
3. Barzilay, R., Lee, L.: Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In: Human Language Technology-NAACL Conference, Edmonton, Canada, pp. 16–23 (2003)
4. Brockett, C., Dolan, W.B.: Support Vector Machines for Paraphrase Identification and Corpus Construction. In: 3rd International Workshop on Paraphrasing (IWP), Korea (2005)
5. Cox, I., Miller, M.L., Bloom, J.A.: Digital Watermarking. Morgan Kaufmann, San Francisco (2002)
6. Duclaye, F., Yvon, F., Collin, O.: Learning Paraphrases to Improve a Question-Answering System. In: 10th Conference of EACL Workshop of Natural Language Processing for Question-Answering, Budapest, Hungary (2003)
7. Hatzigeorgiu, N., Gavrilidou, M., Piperidis, S., Carayannis, G., Papakostopoulou, A., Spiliotopoulou, A., Vacalopoulou, A., Labropoulou, P., Mantzari, E., Papageorgiou, H., Demiros, I.: Design and Implementation of the online ILSP Greek Corpus. In: 2nd International Conference on Language Resources and Evaluation, Athens, Greece, pp. 1737–1742 (2000), http://www.elda.fr/catalogue/en/text/W0022.html
8. Kozareva, Z., Montoyo, A.: Paraphrase identification on the basis of supervised machine learning techniques. In: Salakoski, T., Ginter, F., Pyysalo, S., Pahikkala, T. (eds.) FinTAL 2006. LNCS (LNAI), vol. 4139, pp. 524–533. Springer, Heidelberg (2006)
9. Meral, H.M., Sevinc, E., Unkar, E., Sankur, B., Ozsoy, A.S., Gungor, T.: Syntactic Tools for Text Watermarking. In: SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents IX (2007)
10. Pang, B., Knight, K., Marcu, D.: Syntax-based Alignment of Multiple Translations: Extracting Paraphrases and Generating New Sentences. In: Human-Language Technology Conference (NAACL-HLT), Edmonton, Canada, pp. 102–109 (2003)
11. Provos, N., Honeyman, P.: Hide and Seek: An Introduction to Steganography. In: IEEE Security and Privacy, Oakland, USA, pp. 32–44 (2003)
12. Quirk, C., Brockett, C., Dolan, W.B.: Monolingual Machine Translation for Paraphrase Generation. In: Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, pp. 142–149 (2004)
13. Shinyama, Y., Sekine, S., Sudo, K.: Automatic Paraphrase Acquisition from News Articles. In: Human-Language Technology Conference (NAACL-HLT), San Diego, California, pp. 313–318 (2002)
14. Stamatatos, E., Fakotakis, N.D., Kokkinakis, G.: A practical chunker for unrestricted text. In: Christodoulakis, D.N. (ed.) NLP 2000. LNCS, vol. 1835, pp. 139–150. Springer, Heidelberg (2000)
15. Topkara, M., Taskiran, C.M., Delp, E.: Natural Language Watermarking. In: SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents, San Jose, USA (2005)

BorderFlow: A Local Graph Clustering Algorithm for Natural Language Processing

Axel-Cyrille Ngonga Ngomo and Frank Schumacher

Department of Business Information Systems University of Leipzig
Johannisgasse 26, Leipzig D-04103, Germany
{ngonga,schumacher}@informatik.uni-leipzig.de
http://bis.uni-leipzig.de/

Abstract. In this paper, we introduce BorderFlow, a novel local graph clustering algorithm, and its application to natural language processing problems. For this purpose, we first present a formal description of the algorithm. Then, we use BorderFlow to cluster large graphs and to extract concepts from word similarity graphs. The clustering of large graphs is carried out on graphs extracted from the Wikipedia Category Graph. The subsequent low-bias extraction of concepts is carried out on two data sets consisting of noisy and clean data. We show that BorderFlow efficiently computes clusters of high quality and purity. Therefore, BorderFlow can be integrated in several other natural language processing applications.

1 Introduction

Graph-theoretical models and algorithms have been successfully applied to natural language processing (NLP) tasks over the past years. Especially, graph clustering has been applied to areas as different as language separation [3], lexical acquisition [9] and word sense disambiguation [12]. The graphs generated in NLP are usually large. Therefore, most global graph clustering approaches fail when applied to NLP problems. Furthermore, certain applications (such as concept extraction) require algorithms able to generate a soft clustering. In this paper, we present a novel local graph clustering algorithm called BorderFlow, which is designed especially to compute a soft clustering of large graphs. We apply BorderFlow to two NLP-relevant tasks, i.e., clustering large graphs and concept extraction. We show that our algorithm can be effectively used to tackle these two tasks by providing quantitative and qualitative evaluations of our results.

This paper is structured as follows: in the next section, we describe BorderFlow formally. Thereafter, we present our experiments and results. First, we present the results obtained using BorderFlow on three large similarity graphs extracted from the Wikipedia Category Graph (WCG). By these means, we show that BorderFlow can efficiently handle large graphs. Second, we use BorderFlow to extract domain-specific concepts from two different corpora and show that it computes concepts of high purity. Subsequently, we conclude by discussing possible extensions and applications of BorderFlow.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 547–558, 2009.

2 BorderFlow

BorderFlow is a general-purpose graph clustering algorithm. It uses solely local information for clustering and achieves a soft clustering of the input graph. The definition of cluster underlying BorderFlow was proposed by Flake et al. [5], who state that a cluster is a collection of nodes that have more links between them than links to the outside. When considering a graph as the description of a flow system, Flake et al.'s definition of a cluster implies that a cluster X can be understood as a set of nodes such that the flow within X is maximal while the flow from X to the outside is minimal. The idea behind BorderFlow is to maximize the flow from the border of each cluster to its inner nodes (i.e., the nodes within the cluster) while minimizing the flow from the cluster to the nodes outside of the cluster. In the following, we will specify BorderFlow for weighted directed graphs, as they encompass all other forms of non-complex graphs.

2.1 Formal Specification

Let G = (V, E, ω) be a weighted directed graph with a set of vertices V, a set of edges E and a weighing function ω, which assigns a positive weight to each edge $e \in E$. In the following, we will assume that non-existing edges are edges e such that $\omega(e) = 0$. Before we describe BorderFlow, we need to define functions on sets of nodes. Let $X \subseteq V$ be a set of nodes. We define the set $i(X)$ of inner nodes of X as:

$$i(X) = \{x \in X | \forall y \in V : \omega(xy) > 0 \rightarrow y \in X\}. \tag{1}$$

The set $b(X)$ of border nodes of X is then

$$b(X) = \{x \in X | \exists y \in V \backslash X : \omega(xy) > 0\}. \tag{2}$$

The set $n(X)$ of direct neighbors of X is defined as

$$n(X) = \{y \in V \backslash X \ | \exists x \in X : \omega(xy) > 0\}. \tag{3}$$

In the example of a cluster depicted in Figure 2.1, $X = \{3, 4, 5, 6\}$, the set of border nodes of X is $\{3, 5\}$, $\{6, 4\}$ its set of inner nodes and $\{1, 2\}$ its set of direct neighbors.

Let Ω be the function that assigns the total weight of the edges from a subset of V to another one to these subsets (i.e., the flow between the first and the second subset). Formally:

$$\begin{array}{c} \Omega : 2^V \times 2^V \rightarrow \mathbb{R} \\ \Omega(X, Y) = \sum_{x \in X, y \in Y} \omega(xy). \end{array} \tag{4}$$

We define the border flow ratio $F(X)$ of $X \subseteq V$ as follows:

$$F(X) = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), V \backslash X\big)} = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), n(X)\big)}. \tag{5}$$

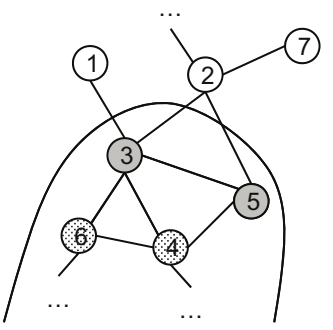


Fig. 1. An exemplary cluster. The nodes with relief are inner nodes, the grey nodes are border nodes and the white are outer nodes. The graph is undirected.

Based on the definition of a cluster by [5], we define a cluster X as a node-maximal subset of V that maximizes the ratio $F(X)$[1], i.e.:

$$\forall X' \subseteq V,\ \forall v \notin X : X' = X + v \rightarrow F(X') < F(X). \tag{6}$$

The idea behind BorderFlow is to select elements from the border $n(X)$ of a cluster X iteratively and insert them in X until the border flow ratio $F(X)$ is maximized, i.e., until Equation (6) is satisfied. The selection of the nodes to insert in each iteration is carried out in two steps. In a first step, the set $C(X)$ of candidates $u \in V \backslash X$ which maximize $F(X + u)$ is computed as follows:

$$C(X) := \underset{u \in n(X)}{\arg\max}\ F(X + u). \tag{7}$$

By carrying out this first selection step, we ensure that each candidate node u which produces a maximal flow to the inside of the cluster X and a minimal flow to the outside of X is selected. The flow from a node $u \in C(X)$ can be divided into three distinct flows:

- the flow $\Omega(u, X)$ to the inside of the cluster,
- the flow $\Omega(u, n(X))$ to the neighbors of the cluster and
- the flow $\Omega(u, V \backslash (X \cup n(X)))$ to the rest of the graph.

Prospective cluster members are elements of $n(X)$. To ensure that the inner flow within the cluster is maximized in the future, a second selection step is necessary. During this second selection step, BorderFlow picks the candidates $u \in C(X)$ which maximize the flow $\Omega(u, n(X))$. The final set of candidates $C_f(X)$ is then

$$C_f(X) := \underset{u \in C(X)}{\arg\max}\ \Omega(u, n(X)). \tag{8}$$

All elements of $C_f(X)$ are then inserted in X if the condition

[1] For the sake of brevity, we shall utilize the notation $X + c$ to denote the addition of a single element c to a set X. Furthermore, singletons will be denoted by the element they contain, i.e., $\{v\} \equiv v$.

$$F(X \cup C_f(X)) \geq F(X) \tag{9}$$

is satisfied.

2.2 Heuristics

One drawback of the method proposed above is that it demands the simulation of the inclusion of each node in $n(X)$ in the cluster X before choosing the best ones. Such an implementation can be time-consuming as nodes in terminology graphs can have a high number of neighbors. The need is for a computationally less expensive criterion for selecting a nearly optimal node to optimize $F(X)$. Let us assume that X is large enough. This assumption implies that the flow from the cluster boundary to the rest of the graph is altered insignificantly when adding a node to the cluster. Under this condition, the following two approximations hold:

$$\begin{aligned} \Omega(b(X), n(X)) &\approx \Omega(b(X+v), n(X+v)), \\ \Omega(b(X), v) - \Omega(d(X, v), X+v) &\approx \Omega(b(X), v). \end{aligned} \tag{10}$$

Consequently, the following approximation holds:

$$\Delta F(X, v) \approx \frac{\Omega(b(X), v)}{\Omega(b(X+v), n(X+v))}. \tag{11}$$

Under this assumption, one can show that the nodes that maximize $F(X)$ maximize the following:

$$f(X, v) = \frac{\Omega(b(X), v)}{\Omega(v, V \backslash X)} \text{ for symmetrical graphs.} \tag{12}$$

Now, BorderFlow can be implemented in a two-step greedy fashion by ordering all nodes $v \in n(X)$ according to $1/f(X, v)$ (to avoid dividing by 0) and choosing the node v that minimizes $1/f(X, v)$. Using this heuristic, BorderFlow is easy to implement and fast to run.

3 Experiments and Results

In this section, we present two series of experiments carried out using Border-Flow.

3.1 Clustering Large Graphs

Experimental Setup
The global aim of this series of experiments was to generate a soft clustering of paradigmatically similar nodes from the WCG. The WCG[2] is a directed cyclic graph whose edges represent the *is-a* relation [15]. Therefore, we needed to define a similarity metric for the categories ζ of the WCG. We chose to use the Jaccard metric

[2] We used the version of July 2007.

$$\sigma_r(\zeta, \zeta') = \frac{2|R(\zeta, r) \cap R(\zeta', r)|}{|R(\zeta, r) \cup R(\zeta', r)|} \tag{13}$$

on the sets

$$R(x, r) = \{y : r(x, y)\}, \tag{14}$$

where r was one of the following three relations:

- *parent-of*: *parent-of*$(\zeta, \zeta') \Leftrightarrow \zeta'$ *is-a* ζ.
- *child-of*: *child-of*$(\zeta, \zeta') \Leftrightarrow \zeta$ *is-a* ζ'.
- *shared-article*: *shared-article*(ζ, ζ') holds iff there exists a Wikipedia article that was tagged using both ζ and ζ'.

To ensure that we did not generate polysemic clusters, we did not use hubs as seeds. In the context of our experiments, we defined hubs as nodes which displayed a connectivity above the average connectivity of the graph. In the graphs at hand, the average connectivity for *parent-of* was 295, 8 for *child-of* and 60 for *shared-article*. It is important to notice that nodes with a connectivity above average could be included in clusters. The quantitative evaluation of our clustering was carried out by using the silhouette index [11].

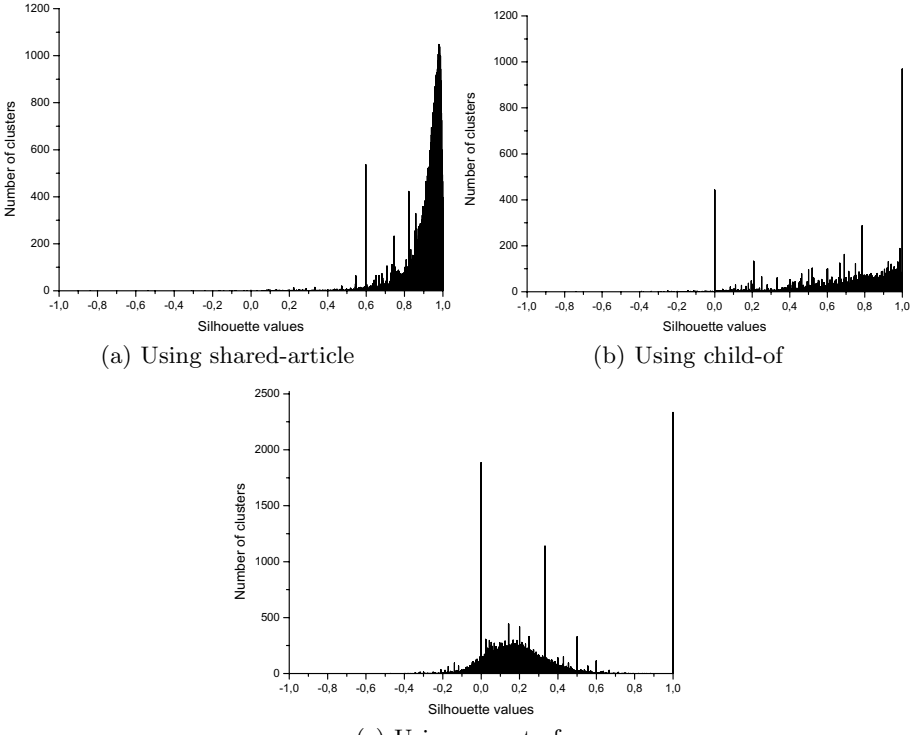
Results

We tried clustering the three resulting similarity graphs using the MCL algorithm [14] but had to terminate the run after seven days without results. Table 1 sums up the results we obtained by using BorderFlow. There were 244,545 initial categories. The clustering based on *child-of* covered solely 31.63% of the categories available because a high percentage of the categories of the WCG do not have any descendant. The other two relations covered approximately the same percentage of categories (82.21% for *shared-article* and 82.07% for *parent-of*).

Figure 2 shows the distribution of the silhouette over all the clusters computed on the three graphs. The best clustering was achieved on the *shared-article*-graph (see Figure 2(a)). We obtained the highest mean (0.92) with the smallest standard deviation (0.09). An analysis of the silhouettes of the clusters computed by using the *parent-of*-graph revealed that the mean of the silhouette lied around 0.74 with a standard deviation of 0.24 (see Figure 2(b)). The smaller average

Table 1. Results on the WCG

	shared-article	*child-of*	*parent-of*
Categories	201,049	77,292	200,688
Coverage	82.21%	31.61%	82.07%
Clusters	93,331	28,568	90,418
Average number of nodes per clusters	3.59	2.29	8.63
Average number of clusters per node	7.74	6.20	19.15
Mean silhouette	0.92	0.20	0.74
Standard deviation	0.09	0.19	0.24



(a) Using shared-article

(b) Using child-of

(c) Using parent-of

Fig. 2. Distribution of silhouette values

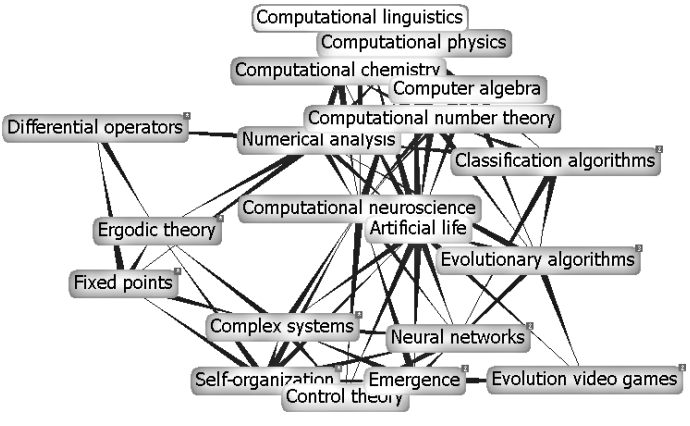


Fig. 3. An example of a cluster containing "Computational Linguistics"

silhouette value was mainly due to the high connectivity of the simila-rity graph generated using this relation, resulting into large clusters and thus a higher flow to the outside (see Table 1). Clustering the *child-of*-graph yielded the worst

results, with a mean of 0.20 and a standard deviation of 0.19. Figure 3 shows an example of a cluster containing "Computational Linguistics".

The high average number of clusters per node (i.e., the number of cluster to which a given node belongs) show how polysemic the categories contained in the WCG are. By using the clustering resulting from our experiments with the *shared-article* relation, one can subdivide the WCG into domain-specific categories and use these to extract domain-specific corpora from Wikipedia. Furthermore, BorderFlow allows the rapid identification of categories similar to given seed categories. Thus, BorderFlow can be used for other NLP applications such as query expansion [2], topic extraction [8] and terminology expansion [7].

3.2 Low-Bias Concept Extraction

In our second series of experiments, we used BorderFlow for the low-bias extraction of concepts (i.e., semantic classes) from word similarity graphs. We were especially interested in providing a qualitative evaluation of the results of BorderFlow. For this purpose, we measured the purity of the clusters computed by BorderFlow against a reference data set.

Experimental Setup

The input to our approach to concept extraction consisted exclusively of domain-specific corpora. Our approach was subdivided into two main steps. First, we extracted the domain-specific terminology from the input without using any a-priori knowledge on the structure of the language to process or the domain to process. Then, we clustered this terminology to domain-specific concepts. The low-bias terminology extraction was carried out by using the approach described in [9] on graphs of the size 20,000. Our experiments were conducted on two data sets: the TREC corpus for filtering [10] and a subset of the articles published by BioMed Central (BMC[3]). Henceforth, we will call the second corpus *BMC*. The TREC corpus is a test collection composed of 233,445 abstracts of publications from the bio-medical domain. It contained 38,790,593 running word forms. The *BMC corpus* consists of full text publications extracted from the BMC Open Access library. The original documents were in XML. We extracted the text entries from the XML data using a SAX[4] Parser. Therefore, it contained a large amount of impurities that were not captured by the XML-parser. The main idea behind the use of this corpus was to test our method on real life data. The 13,943 full text documents contained 70,464,269 running word forms.

For the extraction of concepts, we represented each of the domain-specific terms included in the terminology extracted priorly by its most significant co-occurrences [6]. These were computed in two steps. In a first step, we extracted function words by retrieving the f terms with the lowest information content according to Shannon's law [13]. Function words were not considered as being significant co-occurrences. Then, the s best scoring co-occurrences of each term

[3] http://www.biomedcentral.com

[4] SAX stands for Simple Application Programming Interface for XML.

that were not function words were extracted and stored as its feature vector. The similarity values of the features vectors were computed by using the cosine metric. The resulting similarity values were used to compute a term similarity graph, which was utilized as input for BorderFlow. We evaluated our results quantitavely and qualitatively. In the quantitative evaluation, we compared the clustering generated by BorderFlow with that computed using kNN, which is the local algorithm commonly used for clustering tasks. In the qualitative evaluation, we computed the quality of the clusters extracted by using BorderFlow by comparing them with the controlled MEdical Subject Headings (MESH[5]) vocabulary.

Quantitative Evaluation

In this section of the evaluation, we compared the average silhouettes [11] of the clusters computed by BorderFlow with those computed by kNN on the same graphs. To ensure that all clusters had the same maximal size k, we used the following greedy approach for each seed: first, we initiated the cluster X with the seed. Then, we sorted all $v \in n(X)$ according to their flow to the inside of the cluster $\Omega(v, X)$ in the descending order. Thereafter, we sequentially added all v until the size of the cluster reached k. If it did not reached k after adding all neighbors, the procedure was iterated with $X = X \cup n(X)$ until the size k was reached or no more neighbors were found.

One of the drawbacks of kNN lies in the need for specifying the right value for k. In our experiments, we used the average size of the clusters computed using BorderFlow as value for k. This value was 7 when clustering the TREC data. On the BMC corpus, the experiments with $f = 100$ led to $k = 7$, whilst the experiments with $f = 250$ led to $k = 9$. We used exactly the same set of seeds for both algorithms.

The results of the evaluation are shown in Table 2. On both data sets, BorderFlow significantly outperformed kNN in all settings. On the TREC corpus, both algorithms generated clusters with high silhouette values. BorderFlow outperformed kNN by 0.23 in the best case ($f = 100$, $s = 100$). The greatest difference between the standard deviations, 0.11, was observed when $f = 100$ and $s = 200$. In average, BorderFlow outperformed kNN by 0.17 with respect to the mean silhouette value and by 0.08 with respect to the standard deviation. In the worst case, kNN generated 73 erroneous clusters, while BorderFlow generated 10. The distribution of the silhouette values across the clusters on the TREC corpus for $f = 100$ and $s = 100$ are shown in Figure 4(a) for BorderFlow and Figure 4(b) for kNN.

The superiority of BorderFlow over kNN was better demonstrated on the noisy BMC corpus. Both algorithms generated a clustering with lower silhouette values than on TREC. In the best case, BorderFlow outperformed kNN by 0.57 with respect to the mean silhouette value ($f = 250$, $s = 200$ and $s = 400$). The greatest difference between the standard deviations, 0.18, was observed when $f = 250$ and $s = 400$. In average, BorderFlow outperformed kNN by 0.5 with

[5] http://www.nlm.nih.gov/mesh/

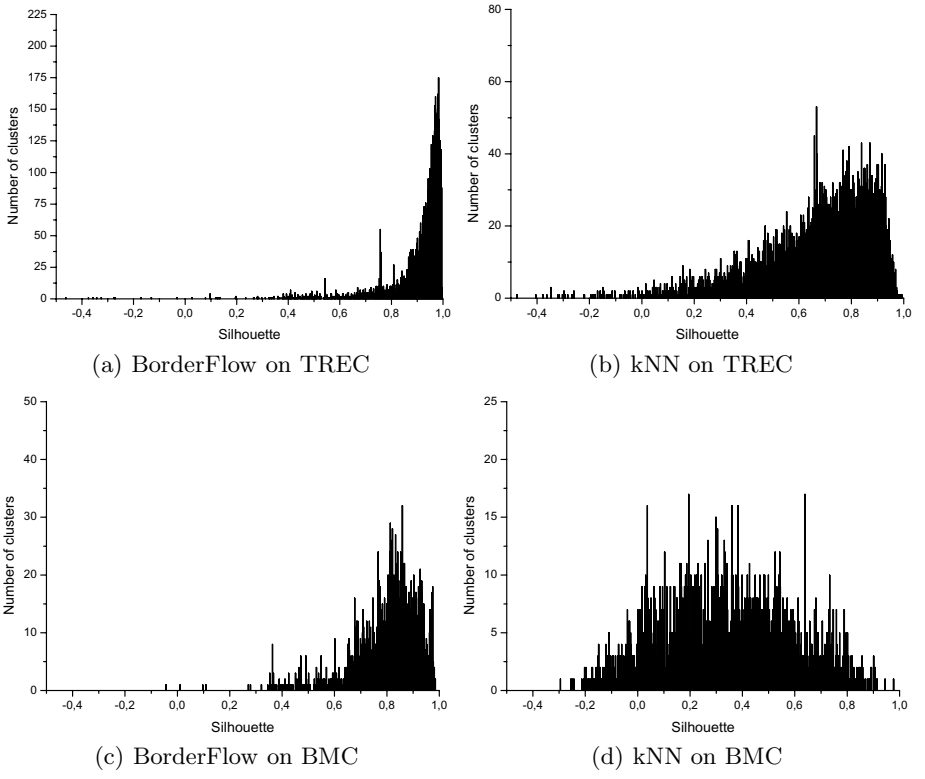


Fig. 4. Distribution of the average silhouette values obtained by using BorderFlow and kNN on the TREC and BMC data set with f=100 and s=100

Table 2. Comparison of the distribution of the silhouette index over clusters extracted from the TREC and BMC corpora. BF stands for BorderFlow. μ the mean of silhouette values over the clusters and σ the standard deviation of the distribution of silhouette values. Erroneous clusters are cluster with negative silhouettes. Bold fonts mark the best results in each experimental setting.

		$\mu \pm \sigma$				Erroneous clusters			
f	s	TREC		BMC		TREC		BMC	
		kNN	BF	kNN	BF	kNN	BF	kNN	BF
100	100	0.68±0.22	**0.91**±0.13	0.37±0.28	**0.83**±0.13	73	**10**	214	**1**
100	200	0.69±0.22	**0.91**±0.11	0.38±0.27	**0.82**±0.12	68	**1**	184	**1**
100	400	0.70±0.20	**0.92**±0.11	0.41±0.26	**0.83**±0.12	49	**1**	142	**1**
250	100	0.81±0.17	**0.93**±0.09	0.23±0.31	**0.80**±0.14	10	**2**	553	**0**
250	200	0.84±0.13	**0.94**±0.08	0.23±0.31	**0.80**±0.14	5	**2**	575	**0**
250	400	0.84±0.12	**0.94**±0.08	0.24±0.32	**0.80**±0.14	2	**1**	583	**0**

respect to the mean silhouette value and by 0.16 with respect to the standard deviation. Whilst BorderFlow was able to compute a correct clustering of the data set, generating maximally 1 erroneous cluster, using kNN led to large sets of up to 583 erroneous clusters ($f = 100$, $s = 400$). Figures 4(c) and 4(d) show the distribution of the silhouette values across the clusters on the BMC corpus for $f = 100$ and $s = 100$.

3.3 Qualitative Evaluation

The goal of the qualitative evaluation was to determine the quality of the content of our clusters. We focused on elucidating whether the elements of the clusters were labels of semantically related categories. To achieve this goal, we compared the content of the clusters computed by BorderFlow with the MESH taxonomy [1]. It possesses manually designed levels of granularity. Therefore, it allows to evaluate cluster purity at different levels. The purity $\varphi(X)$ of a cluster X was computed as follows:

$$\varphi(X) = \max_C \left(\frac{|X \cap M|}{|X \cap C^*|} \right), \tag{15}$$

where M is the set of all MESH category labels, C is a MESH category and C^* is the set of labels of C and all its sub-categories. For our evaluation, we considered only clusters that contained at least one term that could be found in MESH.

The results of the qualitative evaluation are shown in Table 3. The best cluster purity, 89.23%, was obtained when clustering the vocabulary extracted from the TREC data with $f = 250$ and $s = 100$. In average, we obtained a lower cluster purity when clustering the BMC data. The best cluster purity using BMC was 78.88% ($f = 100$, $s = 200$). On both data sets, the difference in cluster quality at the different levels was low, showing that BorderFlow was able to detect fine-grained cluster with respect to the MESH taxonomy. Example of clusters computed with $f = 250$ and $s = 400$ using the TREC corpus are shown in Table 4.

Table 3. Cluster purity obtained using BorderFlow on TREC and BMC data. The upper section of the table displays the results obtained using the TREC corpus. The lower section of the table displays the same results on the BMC corpus. All results are in %.

Level	f=100 s=100	f=100 s=200	f=100 s=400	f=250 s=100	f=250 s=200	f=250 s=400
1	86.81	81.84	81.45	89.23	87.62	87.13
2	85.61	79.88	79.66	87.67	85.82	86.83
3	83.70	78.55	78.29	86.72	84.81	84.63
1	78.58	78.88	78.40	72.44	73.85	73.03
2	76.79	77.28	76.54	71.91	73.27	72.39
3	75.46	76.13	74.74	69.84	71.58	70.41

Table 4. Examples of clusters extracted from the TREC corpus

Cluster members	**Seeds**	**Hypernym**
b_fragilis, c_albicans, l_pneumophila, candida_albicans,	c_albicans	Etiologic agents
mouse_embryos, oocytes, embryo, embryos	mouse_embryos, embryo, oocytes, embryos	Egg cells
leukocytes, macrophages, neutrophils, platelets, pmns	platelets	Blood cells
bolus_doses, intramuscular_injections, intravenous_infusions, intravenous_injections, *developmental_stages*	intravenous_injections	General anesthesia
albuterol, carbamazepine, deferoxamine, diuretic, diuretics, fenoldopam, hmg, inh, mpa, nedocromil_sodium, osa, phenytoin, pht	albuterol	Drugs
atropine, atropine_sulfate, cocaine, *epinephrine*, morphine, *nitroglycerin*, scopolamine, verapamil	atropine_sulfate	Alkaloids
leukocyte, monocyte, neutrophil, polymorphonuclear_leukocyte	polymorphonuclear _leukocyte	White blood cells

4 Conclusion

We presented the novel local graph clustering algorithm BorderFlow and showed how it can be applied to two NLP-relevant tasks. We were able to show that our algorithm can compute clusters with high silhouette indexes. The second series of experiments also showed that BorderFlow is able to compute clusters of high purity. Our algorithm can be extended to produce a hierarchical clustering.

The soft clustering generated by our algorithm can be used to generate a new graph, the nodes of this graph being the clusters computed in the previous step. The edge weights could be computed by making use of a cluster membership function. Thus, BorderFlow can be used for general hierarchical clustering tasks in general and taxonomy and ontology extraction [4] in particular. In addition to these tasks, we will use BorderFlow for tasks including query expansion, topic extraction and lexicon expansion in future work.

References

1. Ananiadou, S., Mcnaught, J.: Text Mining for Biology and Biomedecine, Norwood, MA, USA (2005)
2. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press / Addison-Wesley (1999)

3. Biemann, C.: Chinese whispers - an efficient graph clustering algorithm and its application to natural language processing problems. In: Proceedings of the HLT-NAACL 2006 Workshop on Textgraphs, New York, USA (2006)
4. Fernández-López, M., Gómez-Pérez, A.: Overview and analysis of methodologies for building ontologies. Knowledge Engineering Review 17(2), 129–156 (2002)
5. Flake, G., Lawrence, S., Giles, C.L.: Efficient identification of web communities. In: Proceedings of the 6th ACM SIGKDD, Boston, MA, pp. 150–160 (2000)
6. Heyer, G., Luter, M., Quasthoff, U., Wittig, T., Wolff, C.: Learning relations using collocations. In: Workshop on Ontology Learning. CEUR Workshop Proceedings, vol. 38, CEUR-WS.org. (2001)
7. Jacquemin, C., Klavans, J., Tzoukermann, E.: Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In: Proceeding of ACL-35, pp. 24–31 (1997)
8. Maguitman, A., Leake, D., Reichherzer, T., Menczer, F.: Dynamic extraction topic descriptors and discriminators: towards automatic context-based topic search. In: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, pp. 463–472. ACM, New York (2004)
9. Ngonga Ngomo, A.-C.: SIGNUM: A graph algorithm for terminology extraction. In: Gelbukh, A. (ed.) CICLing 2008. LNCS, vol. 4919, pp. 85–95. Springer, Heidelberg (2008)
10. Robertson, S.E., Hull, D.: The TREC 2001 filtering track report. In: Proceedings of the Text REtrieval Conference (2001)
11. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20(1), 53–65 (1987)
12. Schütze, H.: Automatic word sense discrimination. Computational Linguistics 24(1), 97–123 (1998)
13. Shannon, C.E.: A mathematic theory of communication. Bell System Technical Journal 27, 379–423 (1948)
14. van Dongen, S.: Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht (2000)
15. Zesch, T., Gurevych, I.: Analysis of the Wikipedia Category Graph for NLP Applications. In: Proceedings of the NAACL-HLT 2007 Workshop on TextGraphs, pp. 1–8 (2007)

Generalized Mongue-Elkan Method for Approximate Text String Comparison

Sergio Jimenez[1], Claudia Becerra[1], Alexander Gelbukh[2], and Fabio Gonzalez[1]

[1] Intelligent Systems Laboratory (LISI)
Systems and Industrial Engineering Department
National University of Colombia
[2] Natural Language Laboratory
Center for Computing Research (CIC)
National Polytechnic Institute (IPN), Mexico

Abstract. The Mongue-Elkan method is a general text string comparison method based on an internal character-based similarity measure (e.g. edit distance) combined with a token level (*i.e.* word level) similarity measure. We propose a generalization of this method based on the notion of the generalized arithmetic mean instead of the simple average used in the expression to calculate the Monge-Elkan method. The experiments carried out with 12 well-known name-matching data sets show that the proposed approach outperforms the original Monge-Elkan method when character-based measures are used to compare tokens.

1 Introduction

Approximate string similarity measures are used in many natural language processing (NLP) tasks such as term identification in information extraction, information retrieval, word sense disambiguation, etc. Probably the most well known character-level measure is the *edit distance* proposed by Levenshtein in 1965 [11]. The character-based measures consider the strings to be compared merely as character sequences, which makes this approach affordable when the strings to be compared are single words having misspellings, typographical errors, OCR errors, or even some morphological variations. However, in most human languages, character sequences are split into words (tokens). This property of natural language texts is exploited by token-based measures such as the resemblance coefficients [1] (e.g., *Jaccard*, *Dice*, overlap). The token-based measures compare text strings as sequences of tokens instead of sequences of characters. Such an approach is successful when it is used to compare text strings with many tokens and with different order of the tokens or missing tokens.

All previously mentioned measures based on character or tokens are static. Static string-similarity metrics as they were defined by Bilenko *et al.* [3] are those that compare two character or token sequences in an algorithmic way using solely the information contained into the sequences. Most of the character-based measures are static, that is, the characters into two strings are compared among them with a strategy in order to return a final similarity value. Some

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 559–570, 2009.

adaptive approaches [19,4] use labeled corpus as additional information to affine the parameters of static metrics such as edit distance.

The static token-based measures compare the words between and into the strings in order to return compute similarity. However, there are a wide set of approaches that uses additional information about words in order to compare the sequences. For instance, the use of corpus statistics such as TF-IDF, or combinations with static character-based metrics as it was proposed by Cohen *et al.* [7]. Although, the similarity between tokens is mainly measured using other word-space methods [2] and information sources such as WordNet [17], the static methods have importance in pattern matching, text searching, record linkage, information integration, dictionary and name matching among others particular tasks.

Additionally to the resemblance coefficients, there are others static token-based measures that compare tokens using an internal static character-based measure [5,15,12,13,16]. Monge and Elkan [15] proposed one of those hybrid measures that has been used in many name-matching and record linkage comparative studies [4,7,6,18] obtaining a competitive performance versus other non-static measures. The Monge-Elkan method preserves the properties of the internal character-based measure (*e.g.* ability to deal with misspellings, typos, OCR errors) and deals successfully with missing or disordered tokens. In fact, the Monge-Elkan method is a general and recursive token similarity method that can combine any token comparison measure, which captures semantics, translations, etc.

In this paper, we propose a generalization to the Monge-Elkan method, based on the notion of the generalized arithmetic mean, in order to control the balance between higher and lower values of the similarity between the pairs of tokens being compared. The basic Monge-Elkan method uses a simple arithmetic average to combine internal similarities, the proposed generalization provides the ability to apply heuristics that promotes more similar token pairs. Our experiments with twelve name-matching data sets show that this heuristic leads to improvement of the results.

This paper is organized as follows. In Section 2, the original Monge-Elkan method is presented. Section 3 introduces the proposed generalization. Experimental evaluation is discussed in Section 4, and concluding remarks are given in Section 5.

2 The Monge-Elkan Method

Monge and Elkan [15] proposed a simple but effective method to measure the similarity between two text strings that contains several tokens, using an internal similarity function $sim'(a, b)$ able to measure the similarity between two individual tokens a and b. Given two texts A, B, with $|A|$ and $|B|$ being their respective number of tokens, and an external inter-token similarity measure sim', the Monge-Elkan measure is computed as follows:

$$sim_{MongeElkan}(A,B) = \frac{1}{|A|}\sum_{i=1}^{|A|} \max\{sim'(a_i,b_j)\}_{j=1}^{|B|} \quad (1)$$

Informally, this measure is the average of the similarity values between the more similar token pairs in both strings. In fact, the Monge-Elkan measure approximates the solution to the *optimal assignment problem* in combinatorial optimization [10] with time complexity of $O(|A| \times |B|)$. This approximation is a reasonable tradeoff between accuracy and complexity, because known exact solutions to this combinatorial problem have the time complexity of $O(min(|A|,|B|)^3)$ and require much more sophisticated algorithms [10].

Consider the following example using as internal measure sim' a normalized edit distance converted to a similarity measure (subscripts denote tokens within the strings):

$A =$"*Lenovo inc.*"; $a_1 =$"*Lenovo*"; $a_2 =$"*inc.*"
$B =$"*Lenovo corp.*"; $b_1 =$"*Lenovo*"; $b_2 =$"*corp.*"
$sim'(a_1,b_1) = 1 - \frac{0}{6} = 1$; $sim'(a_1,b_2) = 1 - \frac{5}{6} = 0.1666$
$sim'(a_2,b_1) = 1 - \frac{5}{6} = 0.1666$; $sim'(a_2,b_2) = 1 - \frac{4}{4} = 0$
$sim_{MongeElkan}(A,B) = \frac{1}{2}(\max(sim'(a_1,b_1), sim'(a_1,b_2)) +$
$\max(sim'(a_2,b_1), sim'(a_2,b_2)))$
$sim_{MongeElkan}(A,B) = \frac{1}{2}(1 + 0.1666) = 0.5833$

The Monge-Elkan measure it is not symmetrical, consider the example shown in Figure 1-1. The method iterates the tokens in string A looking for the most similar token in string B in order to make pairs (linked with arrows in Figure 1), then their similarities are averaged. Clearly, the averaged similarity values are different when the content of the strings A and B are swapped. This asymmetry is also evident when the number of tokens is different in both strings (see Figure 1-2). However, this behavior captures the redundancy of the string *"aaaa xaaa yaaa"* at the first case in Figure 1-2. Additionally, at the second case in the same figure, the tokens *"xaaa"* and *"yaaa"* are ignored because of the high similarity between the two tokens *"aaaa"*. Both cases unintentionally implements heuristics convenient for matching. Monge noticed [14] that symmetry may be

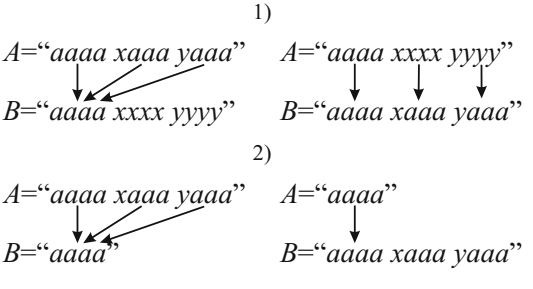


Fig. 1. Asymmetry examples of the Monge-Elkan measure

a natural requirement for many matching applications but not for all. For instance, the name "Alvaro E. Monge" matched "A. E. Monge" while the reverse is not necessarily true.

In the following subsections, for the sake of completeness some character-based string measures that we used as the internal measure sim' in our experimental evaluation are briefly described.

2.1 Bigrams

Q-grams [20] are substrings of length q of a longer string. Depending on q, q-grams are called unigrams, bigrams (or 2-grams), trigrams, and so on. For instance, all bigrams of the string *laptop* are *la*, *ap*, *pt*, *to*, *op*. One way to compute the similarity between two strings is the Dice coefficient

$$sim'(a,b) = 2\frac{|B(a) \cap B(b)|}{|B(a)| + |B(b)|}$$

where $B(x)$ is the set of bigrams of a string x, i.e., the measure is obtained by dividing the number of bigrams common to the two strings by the average number of bigrams in each string. There are several variants of the measures in the q-grams family such as skip-grams, positional q-grams, etc.; see [6] for a comparative study. We used bigrams with one padding character at the beginning and ending of the string because empirical results [9] have shown that padding increase the matching performance.

2.2 Edit Distance

The *edit distance* was originally proposed by Levenshtein [11]. It is equal to the minimum number of editing operations required to transform one sequence into the other. The three basic editing operations are insertion, deletion, and substitution. Several modifications to the original edit distance have been proposed, varying cost schemes and adding more edit operations such as transpositions, opening, and extending gaps; see [8] for a recent survey. The solution [21] for computing the edit distance is a dynamic programming algorithm that stores in a matrix the counts of edit operations for all possible prefixes of both strings. This algorithm computes the edit distance between two strings a and b of length $|a|$ and $|b|$ with a time complexity of $O(|a| \times |b|)$ and space complexity of $O(\min(|a||b|))$.

The edit distance measure can be normalized in the range [0,1] dividing the total number of operations by the number of characters in the longer string. Once normalized, the edit distance can be converted to similarity subtracting the distance value to the number 1.

2.3 Jaro Similarity

The Jaro similarity measure [22] between two strings of length $|a|$ and $|b|$ characters, has the space and time complexity of only $O(|a| + |b|)$. It considers the

number c of characters in common in the two strings and the number of transpositions t as follows:

$$sim_{Jaro}(a,b) = \frac{1}{3}\left(\frac{c}{|a|} + \frac{c}{|b|} + \frac{c-t}{c}\right).$$

If $c = 0$ then $sim_{Jaro} = 0$ by definition. The common characters are considered only in a sliding window of size $\max(|a|,|b|)/2$. In addition, the common characters cannot be shared and are assigned with a greedy strategy. In order to compute the transpositions t between the two strings, the common characters are ordered according to their occurrences in the strings being compared. The number of non-coincident characters at the same positions in both strings is the number of transpositions t. The values returned by the Jaro similarity are in the range $[0, 1]$.

3 Proposed Method

Since the Monge-Elkan method uses the arithmetic mean to average all the similarities measured between the selected token pairs, those similarities have the same weight in the final measure. However, it is possible to think that higher values in the internal sim' values may be more informative as lower ones. We believe that promoting the members with higher similarities in the average calculated according to the equation 1 leads to a more natural similarity measure, more suitable for at least some applications. One way of implementation of such promoting can be the use of a generalized mean instead of the arithmetic mean:

$$\bar{x}(m) = \left(\frac{1}{n} \cdot \sum x_i^m\right)^{\frac{1}{m}} \tag{2}$$

Different values of m result in different known "means": $m = 1$ gives the arithmetic mean; $m = 2$, quadratic mean; $m \to 0$, geometric mean; $m \to -1$, harmonic mean, $m \to \infty$, maximum, and $m \to -\infty$, minimum. The arithmetic mean assigns equal weights to all values of x_i in the mean; values of $m > 1$ promote greater values in the mean. The higher the value of m, the stronger the promotion of higher values of x_i in the mean. The generalized-mean concept is closely related to the Minkowski distance in Euclidean spaces and the values of $m = 1$ and 2 are known as City-block and Euclidean distances respectively.

The proposed generalized Monge-Elkan measure is then expressed as follows:

$$sim_{MongeElkan_m}(a,b) = \left(\frac{1}{|a|}\sum_{i=1}^{|a|}\left(\max\left\{sim'(a_i,b_j)\right\}_{j=1}^{|b|}\right)^m\right)^{\frac{1}{m}} \tag{3}$$

Other approaches implement similar heuristics such as soft-TF-IDF [7,16] and the combination of the Dice coefficient with the Jaro-Winkler measure proposed by Michelson and Knoblock [12]. Those approaches uses also an internal measure

sim' in order to determine the "soft" intersection $A \cap B$ between two sets of tokens A and B, selecting pairs of tokens $[a_i, b_i]$ having $sim'(a_i, b_i) > \theta$. The proposed generalization to the Monge-Elkan method avoids the need to establish an *a priori* threshold for the internal measure sim', implementing the promoting heuristic without assumptions about the range of values of sim'.

Consider again the proposed example in Section 2. Whereas the final measure of 0.5833 obtained with the original Monge-Elkan method seems to be very low taking into account that $sim'(a_1, b_1)$ is equal to 1. Consider using $m = 2$ (*i.e.* quadratic mean or Euclidean distance) in the same pair of strings:

$$sim_{MongeElkan_2}(A, B) = \left(\tfrac{1}{2}\left(1^2 + 0.1666^2\right)\right)^{\frac{1}{2}} = 0.7168$$

This quadratic mean gives greater importance to the number 1 than to 0.1666. The intuition behind the proposed approach is that values of m greater than 1 can improve the performance of the matching measure giving greater importance to those pairs of tokens $[a_i, b_i]$ that are more similar.

4 Experimental Evaluation

The aim of the experiments is to determine which values of m improve the performance of the basic Monge-Elkan measure in a specific string-matching task. For this, we take several name-matching data sets, establish a performance metric for the matching task, select character-based measures, experiment with different m, and evaluate the statistical evidence to assess the possible improvement in the matching performance.

4.1 Experimental Setup

The name-matching task consists of comparing two strings that contain names, addresses, telephone numbers, etc., in order to decide whether the two strings refer to the same entity. A data set used for testing the name matching techniques is usually represented as two sets of strings and a subset of their Cartesian product that defines valid matches. Table 1 describes twelve data sets we used. For each set, we give the total number of different strings in it, the size of the two sets of strings (set_1 and set_2), the size of the Cartesian product of the two sets, the number of pairs that are valid matches, and the total number of tokens.[1] The *Animal* data set was not originally separated into two relations, so the relations were obtained using the 327 valid matches and the 689 strings involved in those matches.

The strings in the data sets were not explicitly separated into tokens, therefore a tokenizing process was performed. We considered as separators of tokens (words) the following characters: blank space, parentheses, equality sign, dash,

[1] The data sets are from http://www.cs.cmu.edu/∼wcohen/match.tar.gz, except for the two last ones, which are from http://www.isi.edu/∼michelso/data/bft_data.zip and https://sourceforge.net/projects/febrl, respectively.

Table 1. Data sets used for our experiments.

Name	# records	$\|set_1\|$	$\|set_2\|$	$\|set_1\| \times \|set_2\|$	# matches	# tokens
Birds-Scott1	38	15	23	345	15	122
Birds-Scott2	719	155	564	87,420	155	3,215
Birds-Kunkel	336	19	315	5,985	19	1,302
Birds-Nybird	985	67	918	61,506	54	1,957
Business	2,139	976	1,163	1,135,088	296	6,275
Game-Demos	911	113	768	86,784	49	3,263
Parks	654	258	396	102,168	250	2,119
Restaurants	863	331	532	176,062	112	9,125
UCD-people	90	45	45	2,025	45	275
Animals	1,378	689	689	474,721	327	3,436
Hotels	1,257	1,125	132	148,500	1,028	9,094
Census	841	449	392	176,008	327	4,083

slash, coma, colon, and semicolon, as well as their sequences; leading and trailing blank spaces were removed. The number of tokens obtained with this tokenizing procedure in each data set is reported in Table 1. Finally, all letters were converted to uppercase, and French diacritics were removed .

We carried out 21 experiments for each data set, combining the three measures explained in Section 2 (bigrams, edit distance, and Jaro similarity) with seven fixed values for he exponent m in equation 3: m =0.00001, 0.5, 1 (*i.e.* the standard Monge-Elkan measure), 1.5, 2, 5, and 10.

4.2 Performance Metrics

The problem of matching between two sets of strings can be viewed as a classification problem over the Cartesian product of the sets. The training or testing data set (a *gold standard*) provides a subset of the Cartesian product of the two sets of strings judged by human annotators as valid matches (positive*s*); its complement is the set of non-valid matches (negatives). In each experiment, a similarity measure value in the range $[0, 1]$ is computed for each possible string pair in the data set. In order to decide whether a pair is a valid match, it is necessary to establish a threshold θ; the pairs of strings with the similarity value greater than or equal to θ are labeled as positive and the rest as negative. For a specific value of θ and a data set, four result sets are defined:

True positives: String pairs marked as valid matches in the gold standard and labeled by the algorithm as positive.

True negatives: String pairs not marked as valid matches in the gold standard and labeled by the algorithm as negative.

False positives: String pairs not marked in the gold standard as valid matches but labeled by the algorithm as positive.

False negatives: String pairs marked as valid matches in the gold standard but labeled by the algorithm as negative.

We measure the performance of a classification task is in terms of *precision*, *recall*, and *F-measure* (the harmonic mean between precision and recall) given by the following expressions:

$$Precision = \frac{|True\,positives|}{|True\,positives| + |False\,positives|}$$

$$Recall = \frac{|True\,positives|}{|True\,positives| + |False\,negatives|}$$

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

For each experiment (a combination of a string matching method and a dataset) the number of true positives, false positives, and false negatives were counted ranging the threshold θ form 0 to 1 with 0.01 increment, thus obtaining 101 values of precision, recall, and F-measure. Figure 2 plots the three measures obtained in a typical experiment, showing the trade-off between precision and recall: recall has a decreasing tendency, while precision is generally increasing. The maximum value reached by the F-measure is known as *F1 score*; it is obtained at the threshold θ with the best balance between precision and recall. F1 score can also be seen as a general performance measure of the experiment. F1 score is close to the intersection point of the precision and recall curves, which is another general performance measure for the experiment; however, the F1 score measure is more commonly used. Finally, the F1 score metric reflects the maximum balanced performance that a string measure can reach in a specific matching task.

The same data can be used to plot a precision vs. recall curve, as shown for a typical experiment in Figure 3. From this curve, it is possible to obtain

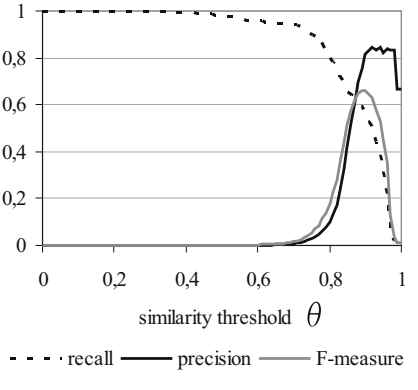


Fig. 2. Precision/recall/F-measure vs. threshold curves for a typical experiment

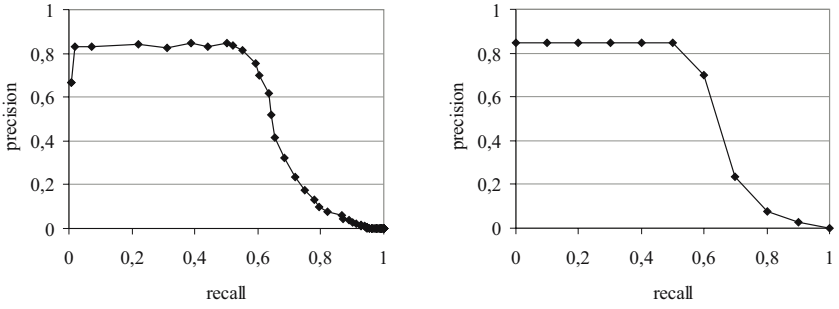


Fig. 3. Example of precision-recall curve and its interpolated version

another general performance measure called *interpolated average precision* (IAP) commonly used in information retrieval [2]. The interpolated precision at recall point r is the maximum precision obtained at points with recall greater than or equal to r. One method for computing IAP is to interpolate the precision vs. recall curve at 11 evenly distributed recall points (*i.e.,* 0, 0.1, 0.2, ...,1); the area under the interpolated precision vs. recall curve is the IAP.

IAP is a good performance measure. Indeed, consider a classifier that obtains precision and recall values of 1 at some θ. The interpolated precision-recall curve becomes a step (*i.e.* 1×1 square) with its elbow at the point $(1, 1)$; the area under that curve is 1. Such a result is the best possible performance, which is achieved by a perfect classifier.

IAP reflects the general performance of the string matching technique considering all possible values for the θ threshold. Although, in practical applications only one value of the θ threshold is used, the performance metrics obtained considering the whole range of values of θ give an assessment of the convenience of the measure for a specific matching problem. In summary, the F1 score metric assesses the maximum possible performance at a fixed threshold and the IAP metric assesses the ability of the classifier to separate the valid matches from non-valid matches regardless the θ threshold.

In order to report a single value for both performance metrics (F1 score and IAP) the results obtained from each dataset are averaged with weights according with the total number of records on each dataset.

4.3 Results

The IAP and F1 score for each internal character-level similarity measure obtained matching each one of the 12 datasets are plotted in Figure 4 and Figure 5, respectively. The natural baseline for the proposed generalization for the Monge-Elkan method is the original method, obtained when $m = 1$ (highlighted in the plots with a dotted vertical line). The results show clearly that values of m below 1 obtain lower performance than the baseline in both metrics. All curves reach

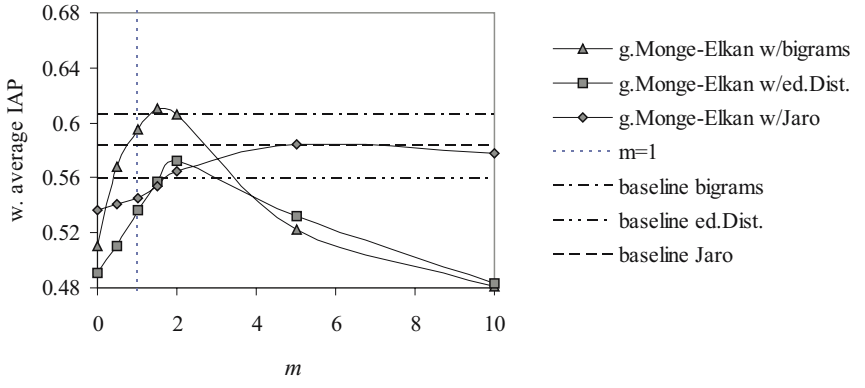


Fig. 4. Weighted Average IAP behavior of the exponent m in extended Monge-Elkan method for each internal character-level string similarity measure

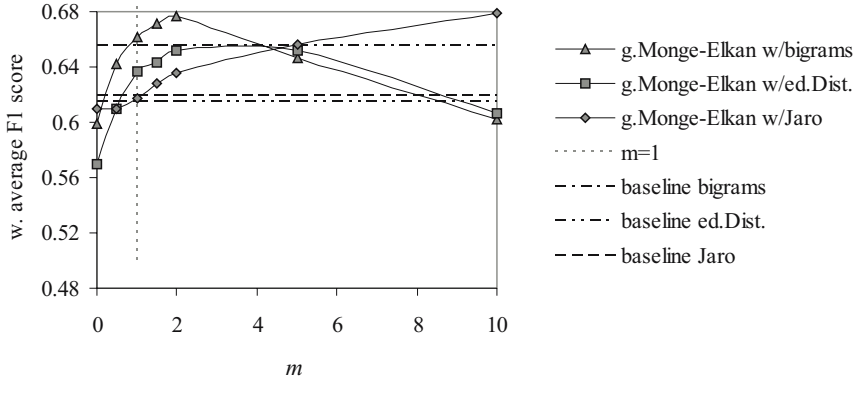


Fig. 5. Weighted average F1 score by the exponent m in generalized Monge-Elkan method for different internal character-level string similarity measures

their maximum performance at values of m above 1, showing that the original method can be improved with the proposed generalization.

Additional baselines are provided in Figures 4 and 5 using directly (without the Monge-Elkan method) the string measures edit distance, bigrams and Jaro similarity for comparing the whole records disregarding any token splitting. Those baselines allow comparing the used character-based measures against the proposed method using the same measure as internal sim' measure. The results obtained with the IAP metric show that whereas the performance of the original Monge-Elkan method is below those baselines, their generalized versions reach the three baselines. Regarding F1 score, those baselines are clearly outperformed.

5 Conclusions

We proposed a generalization of the Monge-Elkan method integrating the arithmetic generalized mean concept to the original expression. That generalization implements a heuristic of giving greater importance in the combined measure to the pairs of tokens whose similarity is higher in comparison with the similarity of other pairs. The proposed method was tested on 12 name-matching data sets with three representative character-based string measures: bigrams, edit distance, and the Jaro similarity. The results showed that the performance of the original Monge-Elkan method can be improved for values of the generalized mean exponent (m) above 1. Particularly, the best results were obtained with $m = 2$ (*i.e.* Euclidean distance) for the measures bigrams and edit distance, and $m = 5$ for the Jaro similarity when they are used as internal similarity measure in the proposed method.

Additionally, the proposed method reached (and clearly outperformed in some cases) the matching performance of the three character-based measures used independently. The original Monge-Elkan method fails to reach that baseline but provides robustness against disordered and missing tokens. The proposed generalization keeps that robustness without compromising the matching performance.

In future work, we plan to incorporate another non-static internal similarity measures to the proposed method and evaluate its application to other tasks such as word sense disambiguation, information retrieval and textual entailment.

References

1. De Baets, B., De Meyer, H.: Transitivity-preserving fuzzification schemes for cardinality-based similarity measures. European Journal of Operational Research 160, 726–740 (2005)
2. Baeza-Yates, R., Ribero-Neto, B.: Modern Information Retrieval. Addison Wesley / ACM Press (1999)
3. Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., Fienberg, S.: Adaptive name matching in information integration. IEEE Intelligent Systems 18 (5), 16–23 (2003)
4. Bilenko, M., Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2003)
5. Chaudhuri, S., Ganjam, K., Ganti, V., Motwani, R.: Robust and efficient fuzzy match for online data cleaning. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (2003)
6. Christen, P.: A comparison of personal name matching: Techniques and practical issues. Technical report, The Australian National University, Department of Computer Science, Faculty of Engineering and Information Technology (2006)
7. Cohen, W.W., Ravikumar, P., Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. In: Proceedings of the IJCAI 2003 Workshop on Information Integration on the Web (2003)
8. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering 19(1), 1–16 (2007)

9. Keskustalo, H., Pirkola, A., Visala, K., Leppänen, E., Järvelin, K.: Non-adjacent digrams improve matching of cross-lingual spelling variants. In: Nascimento, M.A., de Moura, E.S., Oliveira, A.L. (eds.) SPIRE 2003. LNCS, vol. 2857, pp. 252–265. Springer, Heidelberg (2003)
10. Kuhn, H.W.: The hungarian method for the assignment problem. Naval Research Logistics Quarterly 2 2, 83–97 (1955)
11. Levenshtein, V.: Bynary codes capable of correcting deletions, insertions and reversals. Doklady Akademii Nauk SSSR 163(4), 845–848 (1965)
12. Michelson, M., Knoblock, C.A.: Unsupervised information extraction from unstructured, ungrammatical data sources on the world wide web. International Journal on Document Analysis and Recognition 10(3), 211–226 (2007)
13. Minton, S.N., Nanjo, C., Knoblock, C.A., Michalowski, M., Michelson, M.: A heterogeneous field matching method for record linkage. In: Proceedings of the Fifth IEEE International Conference on Data Mining (2005)
14. Monge, A.: An adaptive and efficient algorithm for detecting approximately duplicate database records. International Journal on Information Systems Special Issue on Data Extraction, Cleaning, and Reconciliation (2001)
15. Monge, A., Elkan, C.: The field matching problem: Algorithms and applications. In: Proceedings of The Second International Conference on Knowledge Discovery and Data Mining, (KDD) (1996)
16. Moreau, E., Yvon, F., Cappé, O.: Robust similarity measures for named entities matching. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008) (2008)
17. Pedersen, T., Pakhomov, S.V.S., Patwardhan, S., Chute, C.G.: Measures of semantic similarity and relatedness in the biomedical domain. Journal of Biomedical Informatics 40(3), 288–299 (2007)
18. Piskorski, J., Sydow, M.: Usability of string distance metrics for name matching tasks in polish. In: Proceedings of the 3rd Language and Technology Conference, Poznan (2007)
19. Ristad, E.S., Yianilos, P.N.: Learning string edit distance. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(5) (1998)
20. Ullmann, J.R.: A binary n-gram technique for automatic correction of substitution deletion, insertion and reversal errors in words. The Computer Journal 20(2), 141–147 (1977)
21. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. Journal of the Association for Computing Machinery 21(1), 168–173 (1974)
22. Winkler, W., Thibaudeau, Y.: An application fo the fellegi-sunter model of record linkage to the 1990 us decenial census. Technical report, Bureau of the Census, Washington, D.C. (1991)

Estimating Risk of Picking a Sentence for Document Summarization

Chandan Kumar, Prasad Pingali, and Vasudeva Varma

Language Technologies Research Centre,
International Institute of Information Technology,
Hyderabad, India
chandan_kumar@research.iiit.ac.in,
{pvvpr,vv}@iiit.ac.in

Abstract. Automatic Document summarization is proving to be an increasingly important task to overcome the information overload. The primary task of document summarization process is to pick subset of sentences as a representative of whole document set. We treat this as a decision making problem and estimate the risk involve in making this decision. We calculate the risk of information loss associated with each sentence and extract sentences based on ascending order of their risk. The experimental result shows that the proposed approach performs better than various state of the art approaches.

1 Introduction

Automatic document summarization is extremely helpful in saving time and efforts of the users by helping in tackling the information overload problems. The focus in automatic summarization has been shifted from single document summarization to more complex and challenging problem of multi-document summarization. The goal here is to produce a single text as a compressed version of a given set of documents related to a particular topic with all and only the relevant information.

There are two kinds of approaches to document summarization: abstraction and extraction. Even though efforts have been put to generate an abstract summary that requires using heavy machinery from natural language processing, including grammars and lexicons for parsing and generation, extraction is still the most feasible approach, and most of recent works in this area are based on extraction. Extraction is the process of selecting important units from the original document and concatenating them into a shorter form as summary. Extractive approach to summarization can employ various levels of granularity, e.g., keyword, sentence, or paragraph. Most research concentrates on sentence-extraction because the readability of a list of keywords is typically low while paragraphs are unlikely to cover the information content of a document given summary space constraints.

In this paper, we address the problem of generic multi-document summarization through a sentence extractive procedure. Here the task is to pick subset of

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 571–581, 2009.

sentences from the document cluster(set of documents to be summarized) and present them to user in form of summary that provides an overall sense of the documents content.

We treat sentence extractive summarization as a decision making problem. Given a document or set of documents to summarize, either to a human or system, the selection of few sentences as a representative to whole document set(which contains hundreds of sentence) is a critical decision making problem. As per bayesian decision theory we define sentence selection in terms of risk of information loss, and sentences with minimum expected risk will be chosen as part of summary.

Through this formulation of minimizing risk we come up with a very light-weight function to generate more informative summary than the earlier approaches which uses very complex algorithm for summary generation.

We evaluated our system on DUC-2004 corpus and are able to achieve better reported results for DUC-2004 summarization task as well as for MSE-2005 for all three metric ROUGE-1 ROUGE-2 ROUGE-SU4. Comparison on DUC-2007 dataset supports our results on DUC-2004 and MSE-2005.

The rest of the paper is organized as follows. In section 2 we discuss different approaches to sentence extraction summarization. In section 3 we discuss concept of Bayesian decision theory and formulate sentence extractive summarization as a decision problem where the decision is based on risk analysis. Section 4 describes our method of risk estimation for picking a sentence. In section 5 we explain summary generation process. Evaluation procedure explained in section 6. We discuss the relation of information loss and summarization in section 7, and finally section 8 concludes the paper.

2 Related Work

A variety of extractive summarization techniques have been developed. One of the most popular extractive summarization methods, MEAD [6] is a centroid based method that assigns scores to sentences based on sentence-level and inter-sentence features, including cluster centroids, position, TF*IDF, etc. Lin and hovy [4] selects important content using sentence position, term frequency, topic signature and term clustering. Yih etc all. [19] uses machine learning to find content terms using frequency and position information and followed by a search algorithm to find the best set of sentences that can maximize the content term scores. CLASSY [12,13] uses a learned HMM model to identify summary and non-summary sentence based on some signature terms. Graph based approaches also been explored. Mani and erkan [8,9] uses a graph-connectivity model for sentence extraction with assumption that the nodes which are connected to many other nodes are likely to carry salient information. LexPageRank [8] tried the similar type of approach for computing sentence importance based on the concept of eigenvector centrality. Hardy etc al. [10] uses passage clustering to detect the topic themes and then extracts sentences which reflect these main themes. Harabagiu and Lacatusu [17] have investigated five different topic

representations for extraction. These approaches requires extensive computation of topic themes or signatures, and also they rely on various feature estimation and their parameterization which makes them domain and language dependent. Supervised approaches also been tried extensively [14,15], where the sentence classifiers are trained using human-generated summaries as training examples for feature extraction and parameter estimation. The major drawbacks of the supervised approaches are domain dependency and the problems caused by the inconsistency of human generated summaries.

In contrast to these complex systems, we use a basic information theoretic approach which generates sentence extract on the fly without extensive computation or training or the estimation and parameterization of multiple features which seems to be used in various state of the art algorithms. We formally define sentence extraction as decision theoretic problem, and our approach tries to estimate the risk while picking a sentence and generation of summary with minimum information loss. None of the previous approaches are able to model the loss of information while generating summary. Also while comparing sentence and document cluster models, we actually compare sentence relationship with whole document cluster rather than few topical signatures or centroid themes. Sentence is a function of whole document and its relationship with the entire document cluster should be considered.

3 Decision Theory and Summarization

Bayesian decision theory [11] provides a theoretical foundation to deal with problems of action and inference under uncertainty. The basic idea can be explained by considering the following formulation of a decision problem. Suppose θ is the parameter representing the true state of the nature on which distribution of any random variable depends on. Let $A = \{a_1, a_2, .., a_n\}$ be all the possible actions about θ. In general framework of bayesian decision theory, to each such action a there is associated a loss $L(a, \theta)$ which specifies our decision preferences. The task is to make a decision on which action to take. In order to evaluate each action, we consider the Bayesian expected risk (or loss) associated with taking action a_i

$$Risk(a_i) = \int L(a_i, \theta) d\theta \tag{1}$$

In decision problem the action space, in principle, consists of all the possible actions that the system can take. Bayesian decision theory states that the optimal decision is to choose a Bayes action, i.e., an action a^* that minimizes the conditional expected risk.

$$a^* = arg_n min Risk(a_i) \tag{2}$$

Sentence extractive summarizer can be regarded as a system, where, given a document or set of documents, system needs to choose a subset of sentences and present them to the user to convey the information contained in the document

set. We consider the sentence extraction process as a decision making task, where each sentence is a possible action that can be taken, and the system has to make a decision on which sentences to pick as part of summary.

Consider a document cluster, D, to summarize $D = \{D_1, ..., D_n\}$, where each document contain a set of sentences $D_i = \{s_1, ..., s_n\}$. For simplicity, we represent the document cluster as the set of all sentences from all the documents present in cluster, i.e., $D = \{s_1, ..., s_k\}$. An extractive summary $S = \{s_i, ..., s_j\}$ contains a subset of sentences from the original document set D.

The system has to make a decision on which sentences to choose as part of summary from all k sentences. As mentioned above, the action space consists of all the possible actions that the system can take. In sentence extraction scenario system can pick any of the k sentences present in the documents set towards summary. Hence the action space is the entire sentence collection and the system has to make a decision about which sentence to pick, so $\{s_1, s_2..., s_k\}$ are the possible actions about Document set D. So from (1), the risk of an action i.e. picking a sentence s_i when the information available to system is D, is given as

$$Risk(s_i) = \int L(s_i, D)dD \tag{3}$$

The optimal sentence s^* will be the one with minimum expected risk.

$$s^* = arg_k minRisk(s_i) \tag{4}$$

So sentences with minimum risk are optimal to be chosen as part of summary, we consider sentence selection as a sequential decision making process, and pick sentences towards summary in the order of their risk value. From (3) it is clear that the expected risk of picking a sentence is measured in terms of a loss function, i.e. risk of picking a sentence is proportional to the amount of information lost when we pick a particular sentence s_i to represent the whole document cluster D.

Here the sentence s and document D are the text units, that can be represented as probabilistic distribution of terms. So we need a function to compute loss between two distributions when one tries to predict the other. One such loss function that has proven to be of importance in several fields, including information theory, data compression, mathematical finance, computational learning theory and statistical mechanics is the relative entropy function [1]. It is an information theoretic measure that can measure the extra amount of information required to model one probability distribution using other. Relative entropy (RE) between two probability mass functions $P(x)$ and $Q(x)$ is defined as

$$L(P, Q) = RE(P, Q) = \sum_x P(x) log \frac{P(x)}{Q(x)} \tag{5}$$

Relative entropy has an intuitive interpretation, since it is either zero when the probability distributions are identical or has a positive value, quantifying the difference between the distributions. It gives the number of bits which are wasted

by encoding events from the distribution P with a code based on distribution Q. Relative entropy is a loss function between two probability distributions which measures how bad a given probability distribution is in modeling the other one. So using relative entropy as a loss function between a sentence and document distribution, we can estimate the risk involve in picking that sentence.

4 Estimating Risk

The risk estimation process is modeled into three basic components: (1) A sentence can be viewed as an observation from a probabilistic sentence model (2) A document set can be viewed as an observation from a probabilistic document model (3) The risk of picking a sentence is a loss function between sentence model and document model, i.e. amount of information loss when we consider sentence to represent the document model. It is measured using relative entropy between the sentence and document distribution.

4.1 Document and Sentence Representation

As mentioned sentence s and Document cluster D can be represented as a probability distribution of terms. For the document cluster D, we estimate $P(w|D)$,

$$P(w|D) = \frac{tf(w, D)}{|D|} \tag{6}$$

where $tf(w, D)$ is the frequency of word w in the document D and $|D| = \sum_w D(w)$ is total number of times all words occur in the document set D, it is essentially the length of the document cluster D.

Sentences also modeled in the same manner.

$$P(w|s) = \frac{tf(w, s)}{|s|} \tag{7}$$

here $tf(w, s)$ is the frequency of word w in the sentence s and $|s| = \sum_w d(w)$ is the sentence's length.

4.2 Sentence Expected Risk

Risk of picking a sentence is proportional to loss of information when we consider sentence as a unit to represent the whole document cluster. We measure this according to the relative entropies of the estimated sentence distribution with respect to the estimated document distribution, i.e. relative entropy has been used as a loss function to calculate how much information is required to reconstruct the whole document cluster D from sentence s. We measure relative entropy(RE) of the sentence model with the document model.

$$Risk(s_i) \simeq RE(s_i, D)$$

$$= \sum_{w} p(w|s_i) log \frac{p(w|s_i)}{p(w|D)} \quad (8)$$

The computation of the above formula involves a sum over all the words in D that have a non-zero probability according to $P(w|s_i)$, because if there is a term $w \in D$ not in s_i, its contribution will be zero in the sentence information loss computation. The contribution of any term w in sentence information loss can be defined as how much we lose information by assuming the term is drawn from the sentence model instead of the document model.

Each sentence s_i gets a expected risk $Risk(s_i)$ according to its relative entropy in comparison to document cluster D.

5 Summary Generation

For summary generation, we select sentences with minimum risk while keeping the redundancy minimum. Sentence selection is a sequential decision making process, so until the length of summary is reached we pick the sentence sequentially in the order of their risk value.

1. Identify sentence boundaries in the given set of documents to decompose the document set into individual sentences and form the candidate sentence set S=$\{s_i|i = 1, 2, .., n\}$.
2. For each sentence $s_i \in S$ compute its expected risk value $Risk(s_i)$ using proposed mechanism, then sort the sentences in ascending order based on their risk.
3. Select sentence s_i with minimum $Risk(s_i)$, and move it to the summary set F and remove it from S.
4.
while $|F|$ <required summary length **do**
  Pick the next sentence s_k with minimum $Risk(s_k)$ from set S
  if term overlap between F and $s_k < r$ where r is redundancy threshold **then**
    add s_k to F, remove s_k from S
  else
    remove s_k from S
  end if
end while
5. arrange the sentences in F in chronological order (between documents i.e. based on the time stamp) and order of occurrence (within the document).

Algorithm 1. Summary Generation Steps

For redundancy identifications, we use the measure of number of terms overlapping between the already generated summary and the new sentence being considered. We observed that a 50 percent term overlap between sentences is a good heuristic to estimate redundancy and hence used this redundancy threshold. Once sufficient number of sentences are picked to make the required length of summary, they are arranged based on chronological ordering (between documents i.e. based on the time stamp) and order of occurrence (within the document). Thus, sentences coming from different document will be ordered based

on their source documents date of publication and if two sentences originate from the same document their original order in the source document will be considered order they were found in the original documents to generate the final summary. Any additional words than the required length of summary are truncated. Algorithm 1 shows the operation flow.

6 Evaluation

In order to evaluate the performance of our approach, we use three data sets that have been used in recent multi-document summarization: DUC-2004, MSE-2005, DUC 2007. For evaluation we used the automatic summary evaluation metric, ROUGE [2] which is the standard way of evaluation of summaries. ROUGE is a recall based metric for fixed-length summaries which is based on n-gram cooccurrence. It measures summary quality by counting overlapping units such as the n-gram, word sequences and word pairs between the candidate summary and the reference summary(human written summaries) . We show three of the ROUGE metrics in our experiment results: ROUGE-1 (unigram-based), ROUGE-2 (bigram-based), and ROUGE-SU4 (skip bigram) metric[1]

DUC 2004: The generic multi-document summarization task in DUC 2004 (task 2) involves summarization of 50 TDT (Topic Detection and Tracking) English clusters where each cluster has 10 news articles discussing the same topic.The task was to generate a 100 word summary for each cluster. Four different human judges produced model summaries (reference summaries) for any given cluster for comparison.

The proposed approaches are compared with the top 3 performing systems and two baseline systems (i.e. the lead baseline and the coverage baseline) on task 2 of DUC 2004. The top three systems are the systems with the highest ROUGE scores, chosen from the performing systems in the tasks of DUC 2004. The lead baseline and coverage baseline are two baselines employed in the multi-document summarization tasks at DUC. Lead baseline simply takes the first 100 words of the most recent news article in the document cluster as the summary. And the coverage baseline takes the first sentence from the first document, the first sentence from the second document, and the first sentence from the third document, and so on, until the summary reaches the length limit.

We can see from The table 1-3 that our system outperforms the top performing systems and baseline systems on DUC 2004 tasks over all three ROUGE metrics.

MSE 2005: In 2005, a multi-document summarization task was conducted as part of the Machine Translation and Summarization Workshop at ACL. 25 document sets containing a mixture of English and machine translated Arabic news to English provided and 100 word summary needs to be generated. The news articles are generally shorter than those used in DUC-2004. Ignoring the potential

[1] ROUGE version 1.5.5, with arguments -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -d -e.

Table 1. ROUGE-1 systems comparison on DUC 2004

System	ROUGE-1	95% conf. interval
Our system	0.38664	0.37967 - 0.39372
peer65	0.37950	0.37345 - 0.38535
peer104	0.37115	0.36536 - 0.37691
peer35	0.37567	0.36908 - 0.38181
Coverage baseline	0.34542	0.33297 - 0.35769
Lead baseline	0.32100	0.31468 - 0.32777

Table 2. ROUGE-2 systems comparison on DUC 2004

System	ROUGE-2	95% conf. interval
Our system	0.09423	0.08840 - 0.09986
peer65	0.09171	0.08733 - 0.09642
peer104	0.08489	0.08082 - 0.08903
peer35	0.08389	0.07888 - 0.08879
Coverage baseline	0.07452	0.06734 - 0.08208
Lead baseline	0.06375	0.05915 - 0.06852

Table 3. ROUGE-SU4 systems comparison on DUC 2004

System	ROUGE-SU4	95% conf. interval
Our system	0.13550	0.13063 - 0.14035
peer65	0.13238	0.12845 - 0.13628
peer104	0.12805	0.12447 - 0.13152
peer35	0.12907	0.12516 - 0.13273
Coverage baseline	0.11396	0.10781 - 0.12011
Lead baseline	0.10225	0.09877 - 0.10592

mistakes introduced by the machine translator, we ran our systems without any modifications for this unusual setting.

As shown in Table 4, on this data set, our system outperforms the top performing systems over all three metrics.

DUC 2007: Unlike DUC2004 and MSE 2005, DUC 2007 is not a Generic multi-document summarization task. Instead it is a Query-focused multi document summarization task, with 50 English document clusters generated from AQUANT corpus, where each cluster has 25 news articles discussing the same topic. The participants were asked to generate a 250 word summary for each cluster using the given query. As each cluster had documents relating to the same topic, a generic summarization is also expected to produce satisfactory results without using queries. In DUC 2007 two baselines are given. First is the leadline same as DUC 2004 which simply takes the first 250 words of the most recent news article in the document cluster as the summary. The second baseline is the system CLASSY, a generic multi-document summarizer, the best performer

Table 4. Comparison on MSE 2005

System	ROUGE-1	ROUGE-2	ROUGE-SU4
Our system	.4366	.1652	.1873
peer28	.4342	.1603	.1862
peer29	.4231	.1435 .	.1696
peer30	.4188	.1397	.1690

Table 5. Comparison on DUC 2007

System	ROUGE-1	ROUGE-2	ROUGE-SU4
Our system	.4267	.1096	.1628
CLASSY	.4005	.0938	.1464
Lead baseline	.3347	.0603 .	.1050

in both DUC 2004(peer65) and MSE 2005(peer28). both of this baselines are generic summaries. We compare our result with both these baselines on this dataset. Even with DUC 2007 our system gives better results than CLASSY as is the case with DUC2004 and MSE 2005. The results for DUC 2007 is shown in Table 5.

7 Discussion

When analyzing the reasons for the effective performance of our approach, we found that information loss depicts the quality of the summaries generated.

Automatic Summarization is defined as a process whose goal is to produce a condensed representation of the content of its input for human consumption [7]. In automatic document summarization, input is documents which are viewed as information sources. So we can view summarization as a condensation or compression process of an information source. We know the compression or condensation process of any information source suffers from loss of information. In proposed approach while modeling the risk of picking a sentence towards summary, we are minimizing the loss of information. So we are able to generate summary which preserves the semantic informativeness of original source document i.e. the summary is informative and correlates well with the human written summaries.

With relative entropy metric we pick the sentences towards summary based on its capability to reconstruct the source document. In information theoretic terms, a summary could be viewed as informative if it allows one to reconstruct the source document based on it. This means that is one is asked to guess the content of the source text based on reading the summary, the best summary would be one which allowed him to correctly guess the full text of the source document.

8 Conclusion

In this paper we present a risk minimization framework for sentence extraction. Here we treat summarization as a decision making problem and derive a general extraction mechanism for picking sentences based on an ascending order of the expected risk of information loss, which can be computed separately for each sentence. We evaluated our approach on DUC-2004 and MSE-2005 and DUC 2007 corpus and it outperforms all the reported systems for all three metrics ROUGE-1 ROUGE-2 and ROUGE-SU4.

Our algorithm represents sentence and documents as probability distribution of terms and uses relative entropy as a loss function for sentence extraction. The proposed risk estimation framework opens up the possibility of exploring different sentence extraction methods by considering different loss functions and data representations in the framework.

References

1. Cover, T.M., Thomas, J.A.: Elements of Information Theory. Wiley-Interscience, New York (1991)
2. Lin, C.Y., Hovy, E.H.: Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In: Proceedings of HLT-NAACL 2003(2003)
3. Lin, C., Hovy, E.: The automatic acquistion of topic signatures for text summarization. In: Proc. of COLING (2000)
4. Lin, C.Y., Hovy, E.H.: From Single to Multidocument Summarization: A Prototype System and its Evaluation. In: Proceedings of ACL 2002 (2002)
5. Daume, H., Marcu, D.: Bayesian query-focused summarization. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pp. 05–312 (2006)
6. Radev, D.R., Jing, H.Y., Stys, M., Tam, D.: Centroid-based summarization of multiple documents. Information Processing and Management 40, 919–938 (2004)
7. Mani, I., Maybury, M.: Advances in Automatic Text Summarization. MIT Press, Cambridge (1999)
8. Mani, I., Bloedorn, E.: Summarizing Similarities and Differences Among Related Documents. Journal of Information Retrieval (2000)
9. Erkan, G., Radev, D.: LexPageRank: prestige in multidocument text summarization. In: Proceedings of EMNLP 2004 (2004)
10. Hardy, H., Shimizu, N., Strzalkowski, T., Ting, L., Wise, G.B., Zhang, X.: Cross-document summarization by concept classification. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland (2002)
11. Berger, J.: Statistical decision theory and Bayesian analysis. Springer, Heidelberg (1985)
12. Conroy, J., Schlesinger, J., Goldstein, J., OLeary, D.: Left-brain/right-brain multi-document summarization. In: Proceedings of DUC (2004)
13. Conroy, J., Schlesinger, J., Goldstein, J.: Three classy ways to perform arabic and english multidocument summarization. In: Proc. of MSE (2005)
14. Kupiec, J., Pederson, J., Chen, F.A.: Trainable Document Summarizer. In: Proceedings of the 18th ACM SIGIR, pp. 68–73 (1995)

15. Amini, M.-R., Gallinari, P.: The Use of unlabeled data to improve supervised learning for text summarization. In: Proceedings of the 25th ACM SIGIR, pp. 105–112 (2002)
16. Over, P., Yen, J.: An introduction to DUC 2004 intrinsic evaluation of generic news text summarization systems. In: Proceedings of DUC (2004)
17. Harabagiu, S., Lacatusu, F.: Topic themes for multidocument summarization. In: Proceedings of SIGIR, Salvador, Brazil, pp. 202–209 (2005)
18. Manning, C., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
19. Yih, W.T., Goodman, J., Vanderwende, L., Suzuki, H.: Multi-document summarization by maximizing informative content words. In: IJCAI 2007: 20th International Joint Conference on Artificial Intelligence (January 2007)

The Decomposition of Human-Written Book Summaries

Hakan Ceylan and Rada Mihalcea

University of North Texas
Computer Science Department
{hakan,rada}@unt.edu

Abstract. In this paper, we evaluate the extent to which human-written book summaries can be obtained through cut-and-paste operations from the original book. We analyze the effect of the parameters involved in the decomposition algorithm, and highlight the distinctions in coverage obtained for different summary types.

1 Introduction

Books represent one of the oldest forms of written communication. Despite this fact, given that a large fraction of the electronic documents available online and elsewhere consist of short texts such as Web pages or news articles, the focus of natural language processing techniques to date has been on the automation of methods targeting short documents. We are witnessing however a change: an increasingly larger number of books become available in electronic format, in projects such as Gutenberg, Google Books, or the Million Books project. Similarly, a large number of the books published in recent years are often available in electronic format. Thus, the need for language processing techniques able to handle very large documents such as books is becoming increasingly important.

In this paper, we focus on the problem of book summarization. In particular, we address the first step in automatic summarization, and analyze the extent to which human-written summaries of books can be obtained through extractive methods. We use a decomposition algorithm to automatically identify matches between sentences in the summary and sentences in the book, and thus determine the potential coverage of extractive summarization.

Our work is inspired by the decomposition algorithm previously proposed by Jing & McKeown for single-document summarization [4]. In this paper, we study the applicability of the algorithm to book summaries, and analyze the effect of its various parameters on the coverage of the decomposition. We show that even for long documents such as books, a significant number of summary sentences can be obtained through cut-and-paste operations from the original book. In turn, this coverage depends on the type of summaries being analyzed, with significant differences observed between objective (plot) summaries and interpretative summaries.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 582–593, 2009.

2 Related Work

To our knowledge, with the exception of [6], no research work to date was specifically concerned with the automatic summarization of books. There is, however, a large and growing body of work concerned with the summarization of short documents, with evaluations typically focused on news articles. In particular, a significant number of summarization systems have been proposed during the recent Document Understanding (DUC) / Text Analysis (TAC) conferences – evaluations that usually draw the participation of 20–30 teams every year.

In terms of analysis of the composition of human-written summaries, the work most closely related to ours is the decomposition algorithm proposed in [4], which analyzed the extent to which single-document summaries of news articles are created by using cut-and-paste operations from the text. More recently, their technique has been applied to the analysis of human summaries of Japanese broadcast news [7], and has also been adapted for the analysis of multi-document human summaries [1].

A related study [5], explored the usefulness and coverage of extractive algorithms for single-document summarization. The study found that the best extractive systems are still 15%-24% below the upper bound obtained through agreement calculated over manual summaries, suggesting that there is still room for improving the quality of methods for extractive summarization.

3 Decomposition Algorithm

In order to analyze the sentences in human-written summaries, we used the summary sentence decomposition methodology proposed by Jing & McKeown in [4]. This methodology is based on the assumption that the human summarizers often extract phrases from the original source, and then make further editions to compose the summary sentence. The technique, also referred to as *cut-and-paste*, is supported by the studies in [2] and [3].

The goal of the decomposition analysis of a summary sentence is to discover the original sentences from which the cut-and-paste is done. The task is very difficult, as the extracted phrases of the source sentence can go through several transformations as a result of the editions that human summarizers perform. The transformations may make the summary sentence become quite different as compared to the phrases extracted from the source text.

3.1 Cut-and-Paste Operations

Jing & McKeown analyze 120 sentences from 15 different human-written summaries, and identify six major operations performed during cut-and-paste summarization. These operations are sentence reduction, sentence combination, syntactic transformation, lexical paraphrasing, generalization/specification, and reordering. Human summarizers often use one or a combination of these operations.

Determining whether a phrase in a summary sentence is a result of lexical paraphrasing or a generalization/specification of phrase(s) of the original text is a difficult problem, and it is omitted in both our and Jing & McKeown's studies. Hence, in the decomposition algorithm, we only consider the sentence reduction, sentence combination, syntactic transformation, and reordering operations.

The sentence reduction operation refers to extracting a sentence from the original source and then removing certain words or phrases from it. Sentence combination is the process of combining two or more sentences from the original source and merging them into one sentence. Note that it is possible to combine only parts of the sentences, hence this operation is often used together with the sentence reduction operation. Syntactic transformation refers to modification of the syntactic structure of a sentence, such as word reordering or passive transformations. Finally, the reordering operation is concerned with the position of the sentence in the summary with respect to the sentences in the original text that are used to construct it.

3.2 Problem Formulation

Based on the assumptions discussed in the previous section, the sentence decomposition problem translates into finding the words of a summary sentence inside the original text. If the words come from a single sentence, then we can conclude that either the original sentence is included as-is in the summary, or that the sentence reduction operation is used to eliminate of some of the words. If the position of the words is changed with respect to the original sentence, then syntactic transformations are also involved. Further, if some of the words come from different sentences, then we can conclude that the sentence combination operation is used.

Thus the problem is formulated as follows. Each summary sentence is represented as a sequence of words $(w_1, w_2, ..., w_n)$, and each word w_i can be represented as a set S_i of tuples (P, L), where P is the position of the sentence in the source document which contains w_i, and L is the position of w_i within the sentence. The tuple (P, L) is also referred to as the document position of the word. For example, the tuple $(4, 12)$ for a word w means that w appears in the 12th position of the 4th sentence of the source document. Hence, for each summary sentence, there are $M = \prod_{i=1}^{n} |S_i|$ possible ways to compose it, where $|S_i|$ denotes the cardinality of the set S_i. We are interested in finding the set that will select the most likely document position for each word. The next section describes the algorithm that attempts to do this task in an efficient way.

3.3 Algorithm

The relation between the document position of a word and the position of the preceding words can be used to assign a likelihood probability to the current document position. This likelihood can be estimated using an N-gram model. In our study, we follow [4] and use a bigram model. Hence, we specify the probability $P(w_i = (P, L)_j | w_{i-1}(P, L)_k)$ where $(P, L)_j \in S_i$ and $(P, L)_k \in S_{i-1}$, as the

probability of the word w_i coming from document position $(P, L)_j$ given that the word w_{i-1} comes from position $(P, L)_k$.

Intuitively, given two adjacent words in a summary sentence, we want to find these two words in the original text. We identify the following cases. First, we can find the words in the same sentence, next to each other, and following the same order. This would be the ideal case, and we refer to it as case 1. We can still find these words in the same sentence, in the same order, but not necessarily consecutively as there might be other words in between. This is case 2. The third option, case 3, is where we find the words in the same sentence but in different order. In the fourth option, case 4, the order is retained, however we fail to find them in the same sentence but we find them in neighboring sentences, where the neighboring is determined via a window parameter. Case 5 is the same as case 4 but this time the order is also reversed. The final option, case 6, is when we find the words in non-neighboring sentences in any order.

Each document position can be seen as a state, and each of the cases defined above can be seen as a transition from one of the states of the word w_i to one of the states of the adjacent word w_{i+1} (if the adjacent word has no states, then we consider the states of the next word, w_{i+2}). These transitions can be represented using the bigram model, and by assigning a probability to each of the cases defined above, we define a probability for each possible transition. By building these states and transition probabilities for the entire summary sentence, we are in fact constructing a first-order Hidden Markov Model (HMM). We can use the Viterbi algorithm to find the most likely sequence of states in this HMM. The algorithm associates a probability for the current state based on the probability of the previous states and the transition probability from the previous state to the current state. Hence we define a probability for state $w_i = (P, L)_j$ as $P(w_i = (P, L)_j) = \max_{(P,L)_k \in S_{i-1}} P(w_{i-1} = (P, L)_k) \times P(w_i = (P, L)_j | w_{i-1} = (P, L)_k)$. We also define $P(w_1 = (P, L)_j) = 1$ for all j, which makes sure that every first word of a sentence has an equal chance of being selected. Now we can approximate the probability $\max P(w_1, w_2, ..., w_n)$ as $\max_{(P,L)_j \in S_i} P(w_n = (P, L)_j)$. In order to find the most likely sequence, we keep a back pointer to the previous state that is selected in each step.

For cases 1 through 5, we use the same probabilities as given in [4], namely for case 1 the probability is 1.0, for case 2 is 0.9, for case 3 is 0.8, for case 4 is 0.7 and for case 5 is 0.6. For case 6, we use the probability as a parameter, and change it during our analysis. Note that these probabilities are not tuned for a particular task, but rather just assigned intuitively.

4 Data Set

Unlike the summarization of short documents, which benefits from the data sets made available through the annual DUC/TAC evaluations, there are no publicly available data sets that can be used for the evaluation of methods for book summarization. The lack of such data sets is not surprising since even for humans the summarization of books is more difficult and time consuming than the summarization of short news documents.

We use a strategy similar to [6], and construct a data set starting from the observation that several English and literature courses make use of books that are sometimes also available in the form of abstracts, meant to ease the access of students to the content of the books. Many of these books are classics that are already in the public domain, and thus for most of them we are able to find the online electronic version of the books on sites such as Gutenberg or Online Literature. Unlike [6], who only used two publishers of summaries, we have identified and used nine different publishers that make summaries available online for books studied in the U.S. high-school and college systems.

For our analysis, we target two kinds of summaries. First, we are interested in summaries that describe the plot of the book, in a manner as objective as possible, without any interpretation from the summary writer. We refer to these summaries as *objective summaries*. Second, we are also interested in analysis summaries that describe, in a subjective manner, the interpretation of the facts and of the main story in the book. The publishers often provide these summaries under the *Notes/Analysis/Interpretation* sections. We refer to these summaries as *interpretative summaries*. Objective and interpretative summaries are collected at chapter level for each book.

Figure 1 shows two samples drawn from an objective and an interpretative summary for "The Red Badge of Courage," as made available by Cliff's Notes.

OBJECTIVE SUMMARY: *As the novel opens, the soldiers of a regiment are waiting for battle. After one of the men, a tall soldier, suggests that a battle is imminent, other soldiers argue against the notion. One of the young soldiers, Henry, a private, returns to the hut where the regiment is camped and thinks about war. He recalls his desire to enlist in the army, his mother's refusal to support the idea, and his eventual decision to enlist over her objections.*

INTERPRETATIVE SUMMARY: *The overriding impression of this first chapter is one of conflict. The Union soldiers await a physical battle with the Confederate troops in the area. The eminent external conflict is paralleled by the fight raging in Henry's mind. As the book opens, the reader sees the main character, a soldier waiting for his first battle, ironically engaged in an internal conflict with his own thoughts.*

Fig. 1. Sample objective and interpretative summaries

From the entire set of 64 books that we found with summaries available from at least two publishers, we selected only those that had both objective and interpretative summaries available online. Moreover, we also placed a constraint on the length of these summaries, and require that the interpretative summaries be at least as long as the corresponding objective summaries, when considering the same source.

This left us with a final data set of 31 books, which is used in the analyses described below. The books in this collection have an average length of 87,000

words. The average length for the objective and interpretative summaries is 6,800 words per summary.

5 Parameter Analysis for the Decomposition Algorithm

The data set used in [4] is composed of news articles and their summaries, and the parameters were selected intuitively based on this data genre. In this study, rather than taking the same parameters for granted, we decided to see the effects of the parameters when the decomposition algorithm is applied on book summaries. Therefore we start our evaluation by analyzing these parameters. Note that our goal is not to tune these parameters for a specific application, but rather to see how different parameter values affect the decomposition algorithm.

There are three parameters whose value can affect the algorithm. First, the window size parameter w, which is the number of neighboring sentences that can be considered during a sentence combination operation alongside the current document sentence. Second, the probability p of finding a word outside the current document sentence or outside the neighboring sentences that are within the window size. Finally, the last parameter is a rule that specifies the number of stop words, sw, and/or the number of non-stop words, nsw, that need to be found from a document sentence in order to consider that sentence as being involved in the cut-and-paste process. The values used in [4] for these parameters are $w = 3$, $p = 0.5$, and for the final rule, they enforced that a document sentence has to contribute to the summary sentence with either two or more non-stop words or a non-stop word plus one or more stop words, which in logical form can be written as $nsw \geq 2 \vee (nsw \geq 1 \wedge sw \geq 1)$.

Moreover, in their analysis Jing & McKeown [4] assumed that a summary sentence is formed as a result of cut-and-paste operations on document sentences if and only if at least half of the words in the summary sentence can be found in document sentences. Rather than using the same assumption, we instead compute a *coverage* for each summary sentence, which is simply the percentage of the words in the summary sentence that are found in the source document. This allows us to create plots and visualize the effect of varying coverage. In addition, we can always specify a cutoff value for the coverage and discard the sentences that fall below the cutoff. Note that when we have a cutoff value of 0.5, we would be making the same assumption as [4].

5.1 Number of Words in Contributing Document Sentences

We start by analyzing the third parameter, namely the number of stop words and non-stop words in a contributing document sentence. We note that even at chapter level, the books have significantly greater length than news articles. Since there is a substantially larger number of sentences, the probability of finding a summary word or phrase in multiple sentences is also greatly increased. Therefore we start our experiments by gradually strengthening the conditions, making them more restrictive. Meanwhile, we keep the parameters w, and p fixed to 3

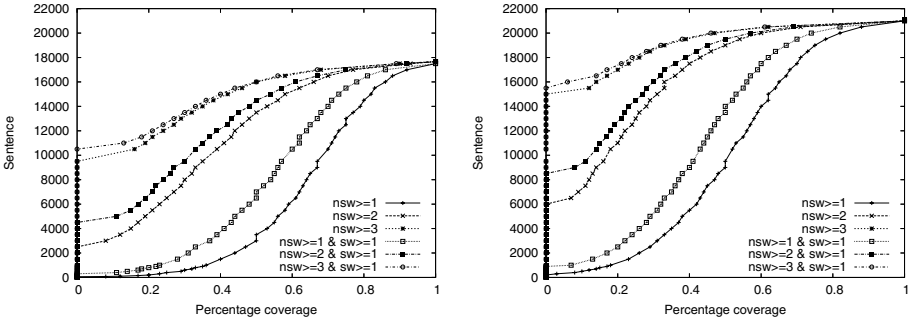


Fig. 2. (a) Objective summaries (b) Interpretative summaries

and 0.5 respectively, which are the values used in [4]. As our least restrictive rule, we start with the simple condition, $(nsw \geq 1)$. The next rules in order of strength are: $(nsw \geq 2) \vee (nsw \geq 1 \wedge sw \geq 1)$, $(nsw \geq 2)$, $(nsw \geq 3) \vee (nsw \geq 2 \wedge sw \geq 1)$, $(nsw \geq 3)$, $(nsw \geq 4) \vee (nsw \geq 3 \wedge sw \geq 1)$, and $(nsw \geq 4)$.

We show the results in Figure 2. The figure plots the coverage for both the objective and interpretative summaries in our collection. For each summary sentence, we measure its coverage as the percentage of words in the sentence that are obtained from the original text through cut-and-paste operations. The Y axis shows all the sentences in all the summaries in our collection, in increasing order of their coverage; we analyze 17,665 objective and 21,077 interpretative summary sentences from 55 sources.

Table 1. Percentage of sentences in the objective and interpretative summaries that are constructed by cut-and-paste operations, as a function of cutoff value and word restriction rules

	Cutoff					
Rules	0.2	0.3	0.4	0.5	0.6	0.7
Objective summaries						
$(nsw \geq 1)$	98.9	96.7	92.4	83.9	66.0	41.9
$(nsw \geq 2) \vee (nsw \geq 1 \wedge sw \geq 1)$	95.9	90.0	80.0	65.5	42.1	20.8
$(nsw \geq 2)$	71.7	55.6	39.7	26.3	14.2	0.69
$(nsw \geq 3) \vee (nsw \geq 2 \wedge sw \geq 1)$	62.2	46.3	31.7	20.1	10.6	0.54
$(nsw \geq 3)$	37.2	25.9	17.0	10.9	0.61	0.35
$(nsw \geq 4) \vee (nsw \geq 3 \wedge sw \geq 1)$	33.6	23.6	15.6	10.0	0.57	0.33
$(nsw \geq 4)$	19.9	15.0	10.4	0.71	0.42	0.26
Interpretative summaries						
$(nsw \geq 1)$	94.7	87.5	74.9	57.7	35.0	17.0
$(nsw \geq 2) \vee (nsw \geq 1 \wedge sw \geq 1)$	88.5	75.5	57.5	37.8	18.2	0.78
$(nsw \geq 2)$	48.2	30.7	17.9	10.2	0.53	0.30
$(nsw \geq 3) \vee (nsw \geq 2 \wedge sw \geq 1)$	40.5	24.7	14.1	0.81	0.44	0.27
$(nsw \geq 3)$	19.5	12.1	0.73	0.48	0.31	0.20
$(nsw \geq 4) \vee (nsw \geq 3 \wedge sw \geq 1)$	17.8	11.2	0.68	0.46	0.30	0.20
$(nsw \geq 4)$	10.5	0.75	0.51	0.37	0.26	0.18

For both the objective and interpretative summaries, increasing the restrictiveness of the conditions results in less coverage. Further, although the results are similar for both objective and interpretative summaries, as the coverage rate increases, the drop in the number of sentences is much quicker for the interpretative summaries. The curves for the objective summaries are less steep because the cutoff affects these summaries to a lesser extent than the interpretative ones.

In order to see this difference, we apply different cutoffs on these results, and report in Table 1 the percentage of the sentences that exceed the cutoff value. This value represents the percentage of the sentences in the summary that are constructed by cut-and-paste operations based on the assumptions.

The difference between the objective and interpretative summaries is clear from the results in Table 1. For example, by applying the second rule and a cutoff of 0.5 (the same assumptions as in [4]), we can identify 65.5% of the objective summary sentences as constructed by cut-and-paste operations. Using the same assumptions, the number dramatically reduces to 37.8% for interpretative summaries. For the remainder of the analyses in this section, for consistency with [4], we fix this parameter to the second rule, $(nsw \geq 2) \vee (nsw \geq 1 \wedge sw \geq 1)$.

5.2 Probability Assigned to Distant Words

Next, we analyze the parameter p, which specifies the bias of the algorithm to consider combinations of phrases from sentences that are distant from each other, i.e., outside the window size w. In order to see this bias, we vary p in 0.1 increments from 0.0 to 0.5 while keeping the other parameters fixed.

We plot the results in Figure 3. Once again, we show the coverage of the sentences in the summaries, for both the objective and the interpretative summaries in our data set. A summary of the results are also displayed in Table 2, which shows the percentage of sentences in the objective and interpretative summaries that are constructed by cut-and-paste operations.

As seen in the figure and in the table, the coverage is greatly effected when p is reduced to 0, which only allows for the combination of sentences found within the

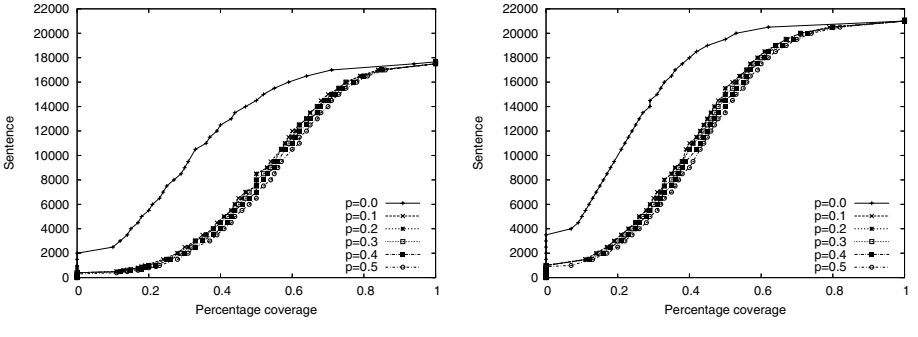


Fig. 3. (a) Objective summaries (b) Interpretative summaries

Table 2. Percentage of sentences in the objective and interpretative summaries that are constructed by cut-and-paste operations varying with cutoff and p

	Cutoff					
Probability	0.2	0.3	0.4	0.5	0.6	0.7
Objective summaries						
$p = 0.0$	70.1	49.4	31.1	18.8	0.91	0.44
$p = 0.1$	94.3	86.5	74.2	57.8	34.6	16.3
$p = 0.2$	94.4	86.8	74.6	58.4	35.1	16.6
$p = 0.3$	94.8	87.5	75.9	56.9	36.5	17.2
$p = 0.4$	95.4	88.7	77.7	62.5	38.8	18.8
$p = 0.5$	95.9	90.0	80.0	65.5	42.1	20.8
Interpretative summaries						
$p = 0.0$	54.0	30.8	15.6	0.75	0.34	0.20
$p = 0.1$	85.5	70.3	50.0	30.7	14.1	0.60
$p = 0.2$	85.7	70.7	50.6	31.2	14.4	0.61
$p = 0.3$	86.3	71.8	52.2	32.5	15.1	0.64
$p = 0.4$	87.4	73.4	54.8	34.9	16.5	0.69
$p = 0.5$	88.5	75.5	57.5	37.8	18.2	0.78

specified window size. This shows that there is a substantial amount of sentences in the summary that are formed by the combination of *distant* sentences.

5.3 Window Size

The final parameter that we are investigating is the window size w. This parameter is related to p so with increasing w we expect to gain some of the sentences that are lost when p is reduced. Thus, for this experiment, we only consider the cases where $p = 0$, the third parameter set to the rule $(nsw \geq 2) \vee (nsw \geq 1 \wedge sw \geq 1)$, and we set w to increasingly larger values for a randomly selected delta of 6, namely $3, 9, 15, 21, 27$. The results are presented in Figure 4.

As expected, increasing the window size allows the algorithm to combine the sentences that are distant from each other. For example, for objective summaries,

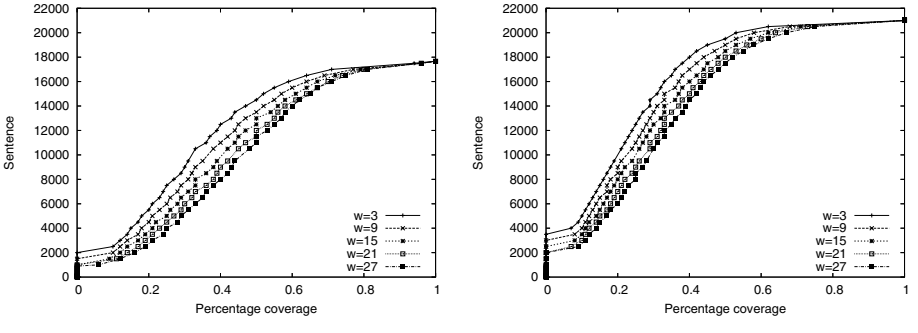


Fig. 4. (a) Objective summaries (b) Interpretative summaries

with a window size of 27, and a cutoff value of 0.5, the percentage of the sentences that are found by the algorithm is increased to 39.8% compared to 18.8% when $p = 0$ with the same parameters and cutoff value.

5.4 The Decomposition of Book Summaries

In their analysis of 1,642 summary sentences, Jing & McKeown found that 42% of these sentences match to a single sentence in the document, 36% of them match to two sentences, and only 3% match to three or more sentences. They conclude that 78% of the summary sentences are constructed by cut-and-paste operations by only counting the ones that use one or two document sentences. Their study was however limited to news articles. In comparison, for book summaries, even at chapter level, we typically have much larger compression rates, so we believe that it is reasonable for a summary sentence to use three or more document sentences. See for instance the example in Figure 5, which is a sentence from a human-written summary for Moby Dick by Herman Melville, along with three sentences in the original book that contribute to this summary sentence.

SUMMARY SENTENCE (51 WORDS): *Flask is the last person down at the table and the first one to leave; since Flask had become an officer he had never known what it was to be otherwise than hungry, more or less, for what he eats does not relieve his hunger as keep it immortal in him.*

BOOK SENTENCES: *Sentence 33 (8 words): Flask was the last person down at the dinner, and Flask is the first man up.*
Sentence 38 (20 words): Therefore it was that Flask once admitted in private, that ever since he had arisen to the dignity of an officer, from that moment he had never known what it was to be otherwise than hungry, more or less.
Sentence 39 (13 words): For what he ate did not so much relieve his hunger, as keep it immortal in him.

Fig. 5. Sample summary sentence and three original sentences used to compose it

Not only this example demonstrates the fact that a summary sentence can be constructed from three document sentences, but it also shows the necessity for either a non-zero probability p or a larger window-size w.

We analyze 17,665 objective, and 21,077 interpretative summary sentences from a total of 55 sources. In order to be able to compare our results with [4], we use the parameters $w = 3$, $p = 0.5$, and the rule $(nsw \geq 2) \vee (nsw \geq 1 \wedge sw \geq 1)$. We show the results in Table 3. For objective summaries, 34.5% of the sentences cannot make the cutoff and hence are not identified as constructed by cut-and-paste operations. On the other hand, 48.4% of the sentences use one to four document sentences, and 17.1% of them use five or more. For interpretative summaries, 62.2% of them cannot make the cutoff, and 25.3% use one to four document sentences, and 12.5% use five or more.

Table 3. Percentage of the number of document sentences used to compose the summary sentences

	Number of document sentences used				
Summary type	1	2	3	4	5 or more
Objective	5.3%	13.8%	15.6%	13.7%	17.1%
Interpretative	3.1%	6.3%	8.6%	7.3%	12.5%

6 Conclusions and Future Work

In this paper, we applied the sentence decomposition algorithm to book summaries, and analyzed the extent to which human-written book summaries can be obtained through cut-and-paste operations from the original book. Specifically, we concentrated on two separate chapter-level summary sources for books: one that attempts to give a summary by redescribing the events in the book in a compact form, which we refer to as objective summaries, and one that attempts to give a summary by capturing the deep meaning of the story by describing the author's ideas and thoughts, which we refer to as interpretative summaries.

As a result of our analysis, based on certain assumptions concerning the value of the parameters used in the algorithm, we found that about 48.4% of the objective summary sentences can be reconstructed from the original document by using cut-and-paste from one to four document sentences. The same analysis leads to only 25.3% cut-and-paste sentences in the interpretative summaries. We can therefore conclude that humans use very little extraction from the original document when writing interpretative summaries, and thus extractive summarization is not-suitable for this summary type. On the other hand, about half of the human-written objective summary sentences are constructed from the original document, which indicates that extractive summarization can be used to create objective book summaries.

Our results differ from those reported by Jing & McKeown, who performed their analysis on short articles from the news domain. First, the decomposition algorithm fails to find a match for 34.5% of the objective summary sentences compared to only 19% in the study of Jing & McKeown. Although some of those sentences are still constructed by cut-and-paste, the algorithm fails to find them due to other transformations applied to the sentences. This demonstrates that heavy editing operations such as paraphrasing or generalization/specification are more frequently encountered in the construction of sentences in book summaries. Further, while Jing & McKeown found that only 3% of the summary sentences are constructed by using three or more document sentences, we find that sentences in book summaries tend to use a larger number of source sentences from the document. This is due to the lengthy nature of the books and the larger compression ratio of their summaries. In the data set we analyzed, the average compression ratio per chapter was 92.5% for objective summaries, and 89% for interpretative summaries, compared to a ratio of only 50-80% typical for shorter documents.

A decomposition analysis for book summaries not only helps us make the distinction between two summary styles (e.g., objective vs. interpretative), but it also helps us see how humans transform the document sentences into summary sentences, and which document sentences are selected for inclusion into a summary through cut-and-paste operations. In future work, we plan to use these results to train a machine learning model for book summarization. Furthermore, we also plan to extend our work to analyze the subjectivity of the objective and interpretative summary sources, and measure the chapter agreements across different summary sources.

Acknowledgments

This work was partially supported by an award from Google Inc. We are grateful to Stan Szpakowicz for useful comments on an earlier draft of this paper.

References

1. Banko, M., Vanderwende, L.: Using n-grams to understand the nature of summaries. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics, Boston, Massachusetts, USA (2004)
2. Endres-niggemeyer, B., Neugebauer, E.: Professional summarising: No cognitive simulation without observation. In: Proceedings of the International Conference in Cognitive Science (1995)
3. Fries, U.: Summaries in newspapers: A textlinguistic investigation. In: Fries, U. (ed.) The Structure of Texts, pp. 47–63. Gunter Narr Verlag Tübingen (1997)
4. Jing, H., McKeown, K.R.: The decomposition of human-written summary sentences. In: SIGIR 1999: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 129–136. ACM, New York (1999)
5. Lin, C., Hovy, E.: The potential and limitations of sentence extraction for summarization. In: Proceedings of the HLT/NAACL Workshop on Automatic Summarization, Edmonton, Canada (May 2003)
6. Mihalcea, R., Ceylan, H.: Explorations in automatic book summarization. In: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007), Prague, Czech Republic (2007)
7. Tanaka, H., Kumano, T., Nishiwaki, M., Itoh, T.: Analysis and modeling of manual summarization of japanese broadcast news. In: Proceedings of the International Joint Conference on Natural Language Processing, Jeju Island, Korea (2005)

Linguistic Ethnography: Identifying Dominant Word Classes in Text

Rada Mihalcea[1,2] and Stephen Pulman[2]

[1] Computer Science Department, University of North Texas
rada@cs.unt.edu
[2] Computational Linguistics Group, Oxford University
sgp@clg.ox.ac.uk

Abstract. In this paper, we propose a method for "linguistic ethnography" – a general mechanism for characterising texts with respect to the dominance of certain classes of words. Using humour as a case study, we explore the automatic learning of salient word classes, including semantic classes (e.g., person, animal), psycholinguistic classes (e.g., tentative, cause), and affective load (e.g., anger, happiness). We measure the reliability of the derived word classes and their associated dominance scores by showing significant correlation across different corpora.

1 Introduction

Text classification is an area in natural language processing that has received a significant amount of interest from both the research and industrial communities, with numerous applications ranging from spam detection and Web directory categorization [4], to sentiment and subjectivity classification [17], emotion recognition [14], gender identification [3] or humour recognition [6]. The task is defined as the automatic identification and labeling of texts that share certain properties, be that a common topic (e.g., "arts"), a common author (e.g., female-authored texts), or a certain feature of the text (e.g., humorous texts).

While there are a number of text classification algorithms that have been proposed to date, there are only a handful of techniques that have been developed to identify the characteristics that are shared by the texts in a given class. Most of the work in this area has focused on the use of weights associated with the words in the text, by using metrics such as tf.idf or information gain, but no attempts have been made to systematically identify broader patterns or word classes that are common in these texts. The relatively small amount of work in this area is understandable since, from a practical perspective, the accurate classification of texts is more important than the identification of general word classes that are specific to the texts in one category.

When the goal however is to *understand the characteristics* of a certain type of text, in order to gain a better understanding of the properties or behaviours modeled by those texts (such as happiness, humour, or gender), then the systematic identification of broad word classes characteristic to the given type of text is considerably more insightful than a mere figure reflecting the accuracy of a text classifier.

A. Gelbukh (Ed.): CICLing 2009, LNCS 5449, pp. 594–602, 2009.

Given a collection of texts, characterised by a certain property that is shared by all the texts in the collection, we introduce a method to automatically discover the classes of words that are dominant in the given type of text. For instance, given a collection of texts that are either humorous, or that reflect the happy mood of the writer, or the specifics of the gender of the author, the method can be used to identify those word classes that are typical to the given texts. For example, the method can find that words that describe *humans* are more often encountered in humorous texts, and thus suggest the human-centeredness of humour. Or, it can find that words that are used to characterize *novelty* are frequently used in texts describing happy moods, and thus indicate a possible connection between novelty and happiness.

In the following, we first introduce the method to automatically assign a dominance score to word classes to indicate their saliency in a type of text. We then describe three lexical resources that define word classes, including Roget's Thesaurus, Linguistic Inquiry and Word Count, and WordNet Affect. We then illustrate the application of the method to humorous texts, we show the classes that are derived by using the dominance score, and evaluate the consistency of the classes using correlation measures.

2 Identifying Dominant Word Classes in Text

In this section, we describe a method to calculate a score associated with a given class of words, as a measure of saliency for the given word class inside a collection of texts that share a common property.

We define the *foreground* corpus to be the collection of texts for which we want to determine the dominant word classes. All the texts in the foreground corpus are assumed to share a certain property, e.g., humorous texts, female-authored texts, etc.

We define the *background* corpus as a collection of texts that are neutral and do not have the property shared by the texts in the foreground. The background corpus plays the role of a baseline, with respect to which we can determine the saliency of the word classes in the foreground corpus. A good background corpus should consist of a mix of texts balanced with respect to genre and source, all of which lack the property of the foreground texts. The purpose of seeking different sources for the construction of the background dataset is to avoid the bias that could be introduced by a specific source or genre. We want to model the characteristics of the foreground corpus, and thus we do not want to learn features that could be specific to a single-source background collection.

Given a class of words $C = \{W_1, W_2, ..., W_N\}$, we define the class coverage in the foreground corpus F as the percentage of words from F belonging to the class C:

$$Coverage_F(C) = \frac{\sum_{W_i \in C} Frequency_F(W_i)}{Size_F}$$

where $Frequency_F(W_i)$ represents the total number of occurrences of word W_i inside the corpus F, and $Size_F$ represents the total size (in words) of the corpus F.

Similarly, we define the class C coverage for the background corpus B:

$$Coverage_B(C) = \frac{\sum_{W_i \in C} Frequency_B(W_i)}{Size_B}$$

The *dominance score* of the class C in the foreground corpus F is then defined as the ratio between the coverage of the class in the corpus F with respect to the coverage of the same class in the background corpus B:

$$Dominance_F(C) = \frac{Coverage_F(C)}{Coverage_B(C)} \quad (1)$$

A dominance score close to 1 indicates a similar distribution of the words in the class C in both the foreground and the background corpus. Instead, a score significantly higher than 1 indicates a class that is dominant in the foreground corpus, and thus likely to be a characteristic of the texts in this corpus. Finally, a score significantly lower than 1 indicates a class that is unlikely to appear in the foreground corpus. Note that if the background corpus is compiled so that it is balanced and mixed across different genres and sources, a score lower than 1 does not indicate a class that is characteristic to the background corpus, but a class that is *not characteristic* to the foreground corpus.

3 Word Classes

We use classes of words as defined in three large lexical resources: Roget's Thesaurus, Linguistic Inquiry and Word Count, and the six main emotions from WordNet Affect. For each lexical resource, we only keep the words and their corresponding class. Note that some resources include the lemmatised form of the words (e.g., Roget), while others include an inflected form (e.g., LIWC); we keep the words as they originally appear in each resource. Any other information such as morphological or semantic annotations are removed for consistency purposes, since not all the resources have such annotations available.

3.1 Roget

Roget is a thesaurus of the English language, with words and phrases grouped into hierarchical classes. A word class usually includes synonyms, as well as other words that are semantically related. Classes are typically divided into sections, subsections, heads and paragraphs, allowing for various granularities of the semantic relations used in a word class. We only use one of the broader groupings, namely the heads. The most recent version of Roget (1987) includes about 100,000 words grouped into close to 1,000 head classes. Table 1 shows three classes, together with a sample of the words included in these classes.

3.2 Linguistic Inquiry and Word Count (LIWC)

LIWC was developed as a resource for psycholinguistic analysis, by Pennebaker and colleagues [10,11]. The 2001 version of LIWC includes about 2,200 words and word stems grouped into about 70 broad categories relevant to psychological processes (e.g., emotion, cognition). The LIWC lexicon has been validated by showing significant correlation between human ratings of a large number of written texts and the rating obtained through LIWC-based analyses of the same texts. Table 1 shows three LIWC classes along with a set of sample words included in these classes.

Table 1. Three word classes from each lexical resource, along with sample words

Class	Words
	Roget
PERFECTION	perfection, faultlessness, lawlessness, impeccability, purity, integrity, chastity
MEDIOCRITY	mediocrity, dullness, indifference, normality, commonness, inferiority
SAFETY	safety, security, surety, assurance, immunity, safeguard, protect, insured
	LIWC
OPTIM(ISM)	accept, best, bold, certain, confidence, daring, determined, glorious, hope
TENTAT(IVE)	any, anyhow, anytime, bet, betting, depending, doubt, fuzzy, guess, hesitant
SOCIAL	adult, advice, affair, anyone, army, babies, band, boy, buddies, calling, comrade
	WordNet Affect
ANGER	wrath, umbrage, offense, temper, irritation, lividity, irascibility, fury, rage
JOY	worship, adoration, sympathy, tenderness, regard, respect, pride, preference, love
SURPRISE	wonder, awe, amazement, astounding, stupefying, dazed, stunned, amazingly

3.3 WordNet Affect

WordNet Affect [15] is a resource that was created starting with WordNet [8], by annotating synsets with several emotions. It uses several resources for affective information, including the emotion classification of Ortony [9]. WordNet Affect was constructed in two stages. First, a core resource was built based on a number of heuristics and semi-automatic processing, followed by a second stage where the core synsets were automatically expanded using the semantic relations available in WordNet.

We extracted the words corresponding to the six basic emotions defined by [9], namely anger, disgust, fear, joy, sadness, surprise. We show three of these classes and a few sample words in Table 1.

4 Analysing Humorous Text

As a case study for our method, we analyse the dominant word classes found in humorous text. This follows on previous work on humour recognition using large collections of humorous texts [7], as well as on more recent work including an analysis of the features found in humorous texts [5]. Unlike previous work, where the words found in verbal humour were manually investigated in an attempt to identify more general word classes, the method proposed here is more general and systematic.

4.1 Foreground Corpus: Two Collections of Humorous Texts

There have been only a relatively small number of previous attempts targeting the computational modeling of humour. Among these, most of the studies have relied on small datasets, e.g. 195 jokes used for the recognition of knock-knock jokes [16], or 200 humorous headlines analysed in [2]. Such small collections may not suffice for the robust learning of features of humorous text.

More recently, we proposed a Web-based bootstrapping method that automatically collects humorous sentences starting with a handful of manually selected seeds, which allowed us to collect a large dataset of 16,000 one-liners [6].

In this paper, we use the corpus of one-liners, as well as a second dataset consisting of humorous news articles [5].

One-liners. A one-liner is a short sentence with comic effects and an interesting linguistic structure: simple syntax, deliberate use of rhetoric devices (e.g. alliteration, rhyme), and frequent use of creative language constructions meant to attract the readers' attention. While longer jokes can have a relatively complex narrative structure, a one-liner must produce the humorous effect "in one shot," with very few words. These characteristics make this type of humor particularly suitable for use in an automatic learning setting, as the humor-producing features are guaranteed to be present in the first (and only) sentence.

Starting with a short seed set consisting of a few one-liners manually identified, the algorithm proposed in [6] automatically identifies a list of webpages that include at least one of the seed one-liners, via a simple search performed with a Web search engine. Next, the webpages found in this way are HTML parsed, and additional one-liners are automatically identified and added to the seed set. The process is repeated several times, until enough one-liners are collected.

Take my advice; I don't use it anyway.
I get enough exercise just pushing my luck.
I took an IQ test and the results were negative.
A clean desk is a sign of a cluttered desk drawer.
Beauty is in the eye of the beer holder.

Fig. 1. Sample examples of one-liners

Two iterations of the bootstrapping process, started with a small seed set of ten one-liners, resulted in a large set of about 24,000 one-liners. After removing the duplicates using a measure of string similarity based on the longest common subsequence, the resulting dataset contains 16,000 one-liners, which are used in the experiments reported in this paper. The one-liners humor style is illustrated in Figure 1, which shows five examples of such one-sentence jokes.

Humorous News Articles. In addition to the one-liners, we also use a second dataset consists of daily stories from the newspaper "The Onion" – a satiric weekly publication with ironic articles about current news, targeting in particular stories from the United States. It is known as "the best satire magazine in the U.S." (Andrew Hammel, German Joys, http://andrewhammel.typepad.com) and "the best source of humour out there" (Jeff Grienfield, CNN senior analyst, http://www.ojr.org/).

All the articles published during August 2005 – March 2006 were collected, which resulted in a dataset of approximately 2,500 news articles. After cleaning the HTML tags, all the news articles that felt outside the 1000–10,000 character length range were removed. This process led to a final dataset of 1,125 news stories with humorous content. Figure 2 shows a sample article from this dataset. This data set was previously used in [5].

Canadian Prime Minister Jean Chrétien and Indian President Abdul Kalam held a subdued press conference in the Canadian Capitol building Monday to announce that the two nations have peacefully and sheepishly resolved a dispute over their common border. Embarrassed Chrétien and Kalam restore diplomatic relations. "We are – well, I guess proud isn't the word – relieved, I suppose, to restore friendly relations with India after the regrettable dispute over the exact coordinates of our shared border," said Chrétien, who refused to meet reporters' eyes as he nervously crumpled his prepared statement. "The border that, er... Well, I guess it turns out that we don't share a border after all."

Fig. 2. Sample news article from "The Onion"

4.2 Background corpus

In order to create a background corpus, we compiled a dataset consisting of a mix of non-humorous sentences from four different sources: (1) *Reuters* titles, extracted from news articles published in the Reuters newswire over a period of one year (8/20/1996 – 8/19/1997); (2) *Proverbs* extracted from an online proverb collection; (3) *British National Corpus (BNC)* sentences; and (4) sentences from the *Open Mind Common Sense* collection of commonsense statements.

Table 2. Dominant word classes from each lexical resource, along with sample words

Class	Score	Sample words
		Roget
ANONYMITY	3.48	you, person, cover, anonymous, unknown, unidentified, unspecified
ODOR	3.36	nose, smell, strong, breath, inhale, stink, pong, perfume, flavor
SECRECY	2.96	close, wall, secret, meeting, apart, ourselves, security, censorship
WRONG	2.83	wrong, illegal, evil, terrible, shame, beam, incorrect, pity, horror
UNORTHODOXY	2.52	error, non, err, wander, pagan, fallacy, atheism, erroneous, fallacious
PEACE	2.51	law, rest, order, peace, quiet, meek, forgiveness, soft, calm, spirit
OVERESTIMATION	2.45	think, exaggerate, overestimated, overestimate, exaggerated,
INTUITION INSTINCT	2.45	drive, feel, idea, sense, blind, feeling, knowledge, natural, tact
INTELLECTUAL	2.41	woman, brain, student, genius, amateur, intellect, pointy, clerk
DISARRANGEMENT	2.18	trouble, throw, ball, bug, insanity, confused, upset, mess, confuse
		LIWC
YOU	3.17	you, thou, thy, thee, thin
I	2.84	myself, mine
SWEAR	2.81	hell, ass, butt, suck, dick, arse, bastard, sucked, sucks, boobs
SELF	2.23	our, myself, mine, lets, ourselves, ours
SEXUAL	2.07	love, loves, loved, naked, butt, gay, dick, boobs, cock, horny, fairy
GROOM	2.06	soap, shower, perfume, makeup
CAUSE	1.99	why, how, because, found, since, product, depends, thus, cos
SLEEP	1.96	bed, wake, asleep, woke, nap, wakes, napping, waking
PRONOUN	1.84	you, they, his, them, she, her, him, nothing, our, its, themselves
HUMANS	1.79	man, men, person, children, human, child, kids, baby, girl, boy
		WordNet Affect
SURPRISE	3.31	stupid, wonder, wonderful, beat, surprised, surprise, amazing, terrific

4.3 Dominant Word Classes in Humorous Text

All the word classes from the resources described in Section 3 were ranked according to the dominance score calculated with formula 1. Those classes that have a high score are the classes that are dominant in humorous text. Table 2 shows the top classes found according to each lexical resource, along with their dominance score and a few sample words.

5 Evaluation

To evaluate the dominance scores obtained for the word classes, we measure the correlation between the scores derived by using different humorous data sets. Since we are interested in a consistent ranking for the dominance scores when derived from different corpora, we use the Spearman correlation metric to measure ranking consistency.

We evaluate the correlation for three data pairs. First, the one-liners data set is randomly split into two non-intersecting data sets consisting of 8,000 one-liners each. In Table 3, this data set pair is labeled *one-liners vs. one-liners*. Second, the humorous news articles set is split into two separate data sets of approximately 550 news articles each (*news articles vs. news articles*). Finally, the last data set pair measures correlation across corpora: dominance scores derived from the entire corpus of 16,000 one-liners compared to the scores obtained for the entire corpus of 1,125 news articles (*one-liners vs. news articles*).

Table 3. Spearman correlation between word class dominance scores derived for different humorous corpora

	Roget	LIWC
one-liners vs. one-liners	0.95	0.96
news articles vs. news articles	0.84	0.88
one-liners vs. news articles	0.63	0.42

Table 3 shows the Spearman correlation measured for the three data set pairs, for the dominance scores obtained for the Roget and LIWC word classes. Not surprisingly, the correlation within the same genre (e.g., one-liners vs. one-liners or news articles vs. news articles) is higher than across genres. However, despite the genre and source differences between the one-liners and the news articles corpora, the correlation is still strong, significant at $p < 0.01$ level using a two-tailed t-test.

For WordNet Affect, because it includes only six classes, we could not calculate the Spearman correlation, since at least 12 points are required for a reliable correlation metric. Instead, the dominance scores obtained for the six emotion classes are listed in Table 4. As seen in the table, the dominance score rankings obtained for the two different data sets (one-liners and humorous news articles) are similar, with *surprise* being by far the most dominant emotion, with a score of 3.31 obtained for the one-liners and 1.91 for the humorous news articles. The *disgust* emotion has also a score larger than 1, but not as significant as the surprise emotion.

Table 4. Dominance scores for the six emotions in WordNet Affect

Emotion	One-liners	News articles
ANGER	0.81	0.73
DISGUST	1.33	1.16
FEAR	1.12	0.97
JOY	1.13	0.83
SADNESS	0.97	0.85
SURPRISE	3.31	1.91

For a second evaluation, we also compare the high dominance classes obtained with our method with the observations made in previous work concerning the features of humorous text. For instance, [7] observed that sexual vocabulary was frequently used in humour. This matches the SEXUAL class that we also identified as dominant. Similarly, [5] found human-centered vocabulary and negative polarity as important characteristics of humorous texts. These features correspond to several dominant classes that we automatically identified: YOU, I, SELF, HUMANS (human-centered vocabulary), and WRONG, UNORTHODOXY, DISARRANGEMENT (negative polarity). Swearing vocabulary (among our classes: SWEAR) was also found useful for humour recognition [13]. Finally, surprise [12,1] was previously identified as one of the elements most frequently encountered in humour. We also found this class as having a high dominance score in humorous texts.

Those observations however were mostly empirical, based on a manual analysis of the words frequently encountered in humour. Instead, our method allows us to systematically identify the word classes that are dominant in humorous texts, which implies increased coverage (a larger number of word classes can be identified), robustness (the same method can be applied to corpora of different sizes), and portability (besides humour, the method can be used to characterize any other types of texts).

6 Conclusions

In this paper, we proposed a method for "linguistic ethnography," which automatically identifies the most dominant word classes in text. By using this method, we can take a step further toward the systematic characterization of texts sharing a common property, such as humorous texts or texts authored by same gender writers.

Using humour as a case study, we showed that the automatically learned word classes are reliable, and they correlate well across different corpora sharing the same humorous property. Moreover, we showed that several of the classes automatically identified correspond to previous empirical observations that were based on manual analysis of humorous texts.

Despite its simplicity, the method proposed is systematic, robust, and portable, and can be used to automatically characterize any types of texts. In future work, we plan to integrate the automatically derived dominant word classes into a classifier for humour recognition. We also plan to test the applicability of the method to other types of texts.

References

1. Attardo, S., Raskin, V.: Script theory revis(it)ed: Joke similarity and joke representation model. Humor: International Journal of Humor Research 4, 3–4 (1991)
2. Bucaria, C.: Lexical and syntactic ambiguity as a source of humor. Humor 17, 3 (2004)
3. Liu, H., Mihalcea, R.: Of men, women, and computers: Data-driven gender modeling for improved user interfaces. In: International Conference on Weblogs and Social Media (2007)
4. McCallum, A., Nigam, K.: A comparison of event models for Naive Bayes text classification. In: Proceedings of AAAI 1998 Workshop on Learning for Text Categorization (1998)
5. Mihalcea, R., Pulman, S.: Characterizing humour: An exploration of features in humorous texts. In: Proceedings of the Conference on Intelligent Text Processing and Computational Linguistics, Mexico City (2007)
6. Mihalcea, R., Strapparava, C.: Making computers laugh: Investigations in automatic humor recognition. In: Proceedings of the Human Language Technology / Empirical Methods in Natural Language Processing conference, Vancouver (2005)
7. Mihalcea, R., Strapparava, C.: Technologies that make you smile: Adding humor to text-based applications. IEEE Intelligent Systems 21, 5 (2006)
8. Miller, G., Leacock, C., Randee, T., Bunker, R.: A semantic concordance. In: Proceedings of the 3rd DARPA Workshop on Human Language Technology, Plainsboro, New Jersey (1993)
9. Ortony, A., Clore, G.L., Foss, M.A.: The referential structure of the affective lexicon. Cognitive Science, 11 (1987)
10. Pennebaker, J., Francis, M.: Linguistic inquiry and word count: LIWC. Erlbaum Publishers, Mahwah
11. Pennebaker, J., King, L.: Linguistic styles: Language use as an individual difference. Journal of Personality and Social Psychology 77, 1296–1312 (1999)
12. Raskin, V.: Semantic Mechanisms of Humor. Kluwer Academic Publications, Dordrecht (1985)
13. Sjobergh, J., Araki, K.: Recognizing humor without recognizing meaning. In: Proceedings of the Workshop on Cross-Language Information Processing (2007)
14. Strapparava, C., Mihalcea, R.: Learning to identify emotions in text. In: Proceedings of the ACM Conference on Applied Computing ACM-SAC 2008, Fortaleza, Brazile (2008)
15. Strapparava, C., Valitutti, A.: Wordnet-affect: an affective extension of wordnet. In: Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon (2004)
16. Taylor, J., Mazlack, L.: Computationally recognizing wordplay in jokes. In: Proceedings of CogSci 2004, Chicago (August 2004)
17. Wiebe, J., Riloff, E.: Creating subjective and objective sentence classifiers from unannotated texts. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 486–497. Springer, Heidelberg (2005)

Author Index