ACL-IJCNLP 2009

Joint Conference of the
47th Annual Meeting of the
Association for Computational Linguistics
and
4th International Joint Conference on
Natural Language Processing
# of the AFNLP

Proceedings of the Student Research Workshop

4 August 2009
Suntec, Singapore

Production and Manufacturing by
World Scientific Publishing Co Pte Ltd
5 Toh Tuck Link
Singapore 596224

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

Introduction

Welcome to the ACL-IJCNLP 2009 Student Research Workshop! The Student Research Workshop is now an established tradition at ACL conferences and provides a venue for student researchers investigating topics in Computational Linguistics and Natural Language Processing to present their work and receive feedback. This year we received a total of 25 submissions coming from 15 different countries, and accepted 12 of them. 5 will be presented orally, and 7 as posters, during a common poster session with the main conference. A total of 43 students and senior researchers agreed to serve on the program committee, which allowed us to assign 3 reviewers per paper. We would like to thank the reviewers for understanding the spirit of the Student Research Workshop and giving careful and constructive reviews. We hope their comments will be helpful to all the students who submitted their work. All presenters were offered travel grants to assist them in their travel to Singapore, thanks to generous support from the U.S. National Science Foundation, The Asian Federation of Natural Language Processing, The Nagao Fund of the AFNLP, and The Walker Fund of the Association for Computational Linguistics.

We are very grateful to Brian Roark and Grace Ngai, our faculty advisors, for their advice, constant support (and reminders!), and obtaining of funding. Finally, we would like to thank the general chair of ACL-IJCNLP 2009, Keh-Yih Su, the program chairs, Jian Su and Janyce Wiebe, the publications chairs Regina Barzilay and Jing-Shin Chang, Haizhou Li and the local organization committee, and Priscilla Rasmussen.

Davis Dimalen, Jenny Rose Finkel, and Blaise Thomson
The ACL-IJCNLP 2009 Student Research Workshop co-chairs

Faculty advisors:

Grace Ngai
Hong Kong Polytechnic University
Kowloon, Hong Kong

Brian Roark
Oregon Health and Science University
Beaverton, Oregon, USA

Chairs:

Davis Muhajereen D. Dimalen
CLCLP, Taiwan International Graduate Program
Academia Sinica, Taiwan

Jenny Rose Finkel
Stanford University
Stanford, California, USA

Blaise Thomson
Cambridge University
Cambridge, UK

Program Committee:

Galen Andrew, Microsoft, USA
Eva Banik, Open University, UK
Shane Bergsma, University of Alberta, Canada
Dan Bohus, Microsoft, USA
Don Erick Bonus, Jose Rizal University, Philippines
Wauter Bosma, Vrije Universiteit, Netherlands
Bill Byrne, University of Cambridge, UK
Colin Cherry, Microsoft , USA
Huang Chu-Ren, Academica Sinica, Taiwan
Shay Cohen, Carnegie Mellon University, USA
Editha D. Dimalen, Mindanao State University, Philippines
Mark Dredze, University of Pennsylvania, USA
Jacob Eisenstein, University of Illinois at Urbana-Champaign, USA
Sharon Goldwater, University of Edinburgh, UK
Mark Greenwood, University of Sheffield, UK
Masato Hagiwara, Nagoya University, Japan
David Hall, Stanford University, USA

LI Haizhou, Institute for Infocomm Research, Singapore
Aurelie Herbelot, University of Cambridge, UK
Samar Husain, IIIT Hyderabad, India
Pei-Yun Hsueh, University of Edinburgh, UK and IBM, USA
Sanaz Jabbari, University of Sheffield, UK
Maggie LI (Li Wenjie Maggie), Hong Kong Polytechnic University, China
Yuji Matsumoto, NAIST, Japan
David McClosky, Brown University, USA
Roser Morante, University of Antwerp, Belgium
Teruhisa Misu, NICT/ATR, Japan
Vincent Ng, University of Texas at Dallas, USA
Patrick Pantel, Yahoo, USA
Jong C. Park, KAIST, Korea
JIN Peng, Peking University, China
Le Hong Phuong, INRIA Lorraine, France
Emily Pitler, University of Pennsylvania, USA
Daniel Ramage, Stanford University, USA
Antti-Veikko Rosti, BBN, USA
Rachel Edita Roxas, De LaSalle University-Manila, Philippines
Philipp Spanger, Tokyo Institute of Technology, Japan
Reut Tsarfaty, University of Amsterdam, Netherlands
Joseph Turian, University of Montreal, Canada
Lonneke van der Plas, University of Geneva, Switzerland
Sumithra Velupillai, Stockholm University/KTH, Sweden
Andreas Vlachos, University of Cambridge, UK
Liang-Chih Yu, Yuan-Ze University, China
JIA Yuxiang, Peking University, China

Table of Contents

Conference Program

Student Research Workshop, 4 August, 2009

Oral Session (10:15–12:20)

10:15–10:40 *Sense-based Interpretation of Logical Metonymy Using a Statistical Method*
Ekaterina Shutova

10:40–11:05 *Insights into Non-projectivity in Hindi*
Prashanth Mannem, Himani Chaudhry and Akshar Bharati

11:05–11:30 *Annotating and Recognising Named Entities in Clinical Notes*
Yefeng Wang

11:30–11:55 *Paraphrase Recognition Using Machine Learning to Combine Similarity Measures*
Prodromos Malakasiotis

11:55–12:20 *A System for Semantic Analysis of Chemical Compound Names*
Henriette Engelken

Poster Session (12:20–14:20)

Sentence diagram generation using dependency parsing
Elijah Mayfield

Accurate Learning for Chinese Function Tags from Minimal Features
Caixia Yuan, Fuji Ren and Xiaojie Wang

Optimizing Language Model Information Retrieval System with Expectation Maximization Algorithm
Justin Liang-Te Chiu and Jyun-Wei Huang

Data Cleaning for Word Alignment
Tsuyoshi Okita

The Modulation of Cooperation and Emotion in Dialogue: The REC Corpus
Federica Cavicchio

Student Research Workshop, 4 August, 2009 (continued)

Clustering Technique in Multi-Document Personal Name Disambiguation
Chen Chen, Hu Junfeng and Wang Houfeng

Creating a Gold Standard for Sentence Clustering in Multi-Document Summarization
Johanna Geiss

Sense-based Interpretation of Logical Metonymy Using a Statistical Method

Ekaterina Shutova
Computer Laboratory
University of Cambridge
15 JJ Thomson Avenue
Cambridge CB3 0FD, UK
Ekaterina.Shutova@cl.cam.ac.uk

Abstract

The use of figurative language is ubiquitous in natural language texts and it is a serious bottleneck in automatic text understanding. We address the problem of interpretation of *logical metonymy*, using a statistical method. Our approach originates from that of Lapata and Lascarides (2003), which generates a list of non-disambiguated interpretations with their likelihood derived from a corpus. We propose a novel sense-based representation of the interpretation of logical metonymy and a more thorough evaluation method than that of Lapata and Lascarides (2003). By carrying out a human experiment we prove that such a representation is intuitive to human subjects. We derive a ranking scheme for verb senses using an unannotated corpus, WordNet sense numbering and glosses. We also provide an account of the requirements that different aspectual verbs impose onto the interpretation of logical metonymy. We tested our system on verb-object metonymic phrases. It identifies and ranks metonymic interpretations with the mean average precision of 0.83 as compared to the gold standard.

1 Introduction

Metonymy is defined as the use of a word or a phrase to stand for a related concept which is not explicitly mentioned. Here are some examples of metonymic phrases:

(1) The *pen* is mightier than the *sword*.

(2) He played *Bach*.

(3) He drank *his glass*. (Fass, 1991)

(4) He enjoyed *the book*. (Lapata and Lascarides, 2003)

(5) After *three martinis* John was feeling well. (Godard and Jayez, 1993)

The metonymic adage in (1) is a classical example. Here the *pen* stands for the press and the *sword* for military power. In the following example *Bach* is used to refer to the composer's music and in (3) the *glass* stands for its *content*, i.e. the actual *drink* (beverage).

The sentences (4) and (5) represent a variation of this phenomenon called *logical metonymy*. Here both *the book* and *three martinis* have eventive interpretations, i.e. the noun phrases stand for the events of *reading the book* and *drinking three martinis* respectively. Such behaviour is triggered by the type requirements the verb (or the preposition) places onto its argument. This is known in linguistics as a phenomenon of *type coercion*. Many existing approaches to logical metonymy explain systematic syntactic ambiguity of metonymic verbs (such as *enjoy*) or prepositions (such as *after*) by means of type coercion (Pustejovsky, 1991; Pustejovsky, 1995; Briscoe et al., 1990; Verspoor, 1997; Godard and Jayez, 1993).

Logical metonymy occurs in natural language texts relatively frequently. Therefore, its automatic interpretation would significantly facilitate the task of many NLP applications that require semantic processing (e.g., machine translation, information extraction, question answering and many others). Utiyama et al. (2000) followed by Lapata and Lascarides (2003) used text corpora to automatically derive interpretations of metonymic phrases.

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 1–9,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

Utiyama et al. (2000) used a statistical model for the interpretation of general metonymies for Japanese. Given a verb-object metonymic phrase, such as *read Shakespeare*, they searched for entities the object could stand for, such as *plays of Shakespeare*. They considered all the nouns co-occurring with the object noun and the Japanese equivalent of the preposition *of*. Utiyama and his colleagues tested their approach on 75 metonymic phrases taken from the literature and reported a precision of 70.6%, whereby an interpretation was considered correct if it made sense in some imaginary context.

Lapata and Lascarides (2003) extend Utiyama's approach to interpretation of logical metonymies containing aspectual verbs (e.g. *begin the book*) and polysemous adjectives (e.g. *good meal* vs. *good cook*). Their method generates a list of interpretations with their likelihood derived from a corpus.

Lapata and Lascarides define an interpretation of logical metonymy as a verb string, which is ambiguous with respect to word sense. Some of these strings indeed correspond to paraphrases that a human would give for the metonymic phrase. But they are not meaningful as such for automatic processing, since their senses still need to be disambiguated in order to obtain the actual meaning. For example, compare the *grab* sense of *take* vs. its *film* sense for the metonymic phrase *finish video*. It is obvious that only the latter sense is a correct interpretation.

We extend the experiment of Lapata and Lascarides by disambiguating the interpretations with respect to WordNet (Fellbaum, 1998) synsets (for verb-object metonymic phrases). We propose a novel ranking scheme for the synsets using a non-disambiguated corpus, address the issue of sense frequency distribution and utilize information from WordNet glosses to refine the ranking.

We conduct and experiment to show that our representation of a metonymic interpretation as a synset is intuitive to human subjects. In the discussion section we provide an overview of the constraints on logical metonymy pointed out in linguistics literature, as well as proposing some additional constraints (e.g. on the type of the metonymic verb, on the type of the reconstructed event, etc.)

Metonymic Phrase	Interpretations	Log-probability
finish video	film	-19.65
	edit	-20.37
	shoot	-20.40
	view	-21.19
	play	-21.29
	stack	-21.75
	make	-21.95
	programme	-22.08
	pack	-22.12
	use	-22.23
	watch	-22.36
	produce	-22.37

Table 1: Interpretations of Lapata and Lascarides (2003) for *finish video*

2 Lapata and Lascarides' Method

The intuition behind the approach of Lapata and Lascarides is similar to that of Pustejovsky (1991; 1995), namely that there is an event not explicitly mentioned, but implied by the metonymic phrase (*begin to read the book*, or *the meal that tastes good* vs. *the cook that cooks well*). They used the British National Corpus (BNC)(Burnard, 2007) parsed by the Cass parser (Abney, 1996) to extract events (verbs) co-occurring with both the metonymic verb (or adjective) and the noun independently and ranked them in terms of their likelihood according to the data. The likelihood of a particular interpretation is calculated using the following formula:

$$P(e, v, o) = \frac{f(v, e) \cdot f(o, e)}{N \cdot f(e)}, \qquad (1)$$

where e stands for the eventive interpretation of the metonymic phrase, v for the metonymic verb and o for its noun complement. $f(e)$, $f(v, e)$ and $f(o, e)$ are the respective corpus frequencies. $N = \sum_i f(e_i)$ is the total number of verbs in the corpus. The list of interpretations Lapata and Lascarides (2003) report for the phrase *finish video* is shown in Table 1.

Lapata and Lascarides compiled their test set by selecting 12 verbs that allow logical metonymy[1] from the lexical semantics literature and combining each of them with 5 nouns. This yields 60 phrases, which were then manually filtered, excluding 2 phrases as non-metonymic.

They compared their results to paraphrase judgements elicited from humans. The subjects were presented with three interpretations for each

[1] *attempt, begin, enjoy, finish, expect, postpone, prefer, resist, start, survive, try, want*

metonymic phrase (from high, medium and low probability ranges) and were asked to associate a number with each of them reflecting how good they found the interpretation. They report a correlation of 0.64, whereby the inter-subject agreement was 0.74. It should be noted, however, that such an evaluation scheme is not very informative as Lapata and Lascarides calculate correlation only on 3 data points for each phrase out of many more yielded by the model. It fails to take into account the quality of the list of top interpretations, although the latter is deemed to be the aim of such applications. In comparison the fact that Lapata and Lascarides initially select the interpretations from high, medium or low probability ranges makes the task significantly easier.

3 Alternative Interpretation of Logical Metonymy

The approach of Lapata and Lascarides (2003) produces a list of non-disambiguated verbs, essentially just strings, representing possible interpretations of a metonymic phrase. We propose an alternative representation of metonymy interpretation consisting of a list of senses that map to WordNet synsets. However, the sense-based representation builds on the list of non-disambiguated interpretations similar to the one of Lapata and Lascarides.

Our method consists of the following steps:

- **Step 1** Use the method of Lapata and Lascarides (2003) to obtain a set of candidate interpretations (strings) from a non-annotated corpus. We expect our reimplementation of the method to extract data more accurately, since we use a more robust parser (RASP (Briscoe et al., 2006)), take into account more syntactic structures (coordination, passive), and extract our data from a newer version of the BNC.

- **Step 2** Map strings to WordNet synsets. We noticed that good interpretations in the lists yielded by Step 1 tend to form coherent semantic classes (e.g. *take, shoot [a video]* vs. *view, watch [a video]*). We search the list for verbs, whose senses are in hyponymy and synonymy relations with each other according to WordNet and store these senses.

- **Step 3** Rank the senses, adopting Zipfian sense frequency distribution and using the initial string likelihood as well as the information from WordNet glosses.

Sense disambiguation is essentially performed in both Step 2 and Step 3. One of the challenges of our task is that we use a non-disambiguated corpus while ranking particular senses. This is due to the fact that there is no word sense disambiguated corpus available, which would be large enough to reliably extract statistics for metonymic interpretations.

4 Extracting Ambiguous Interpretations

4.1 Parameter Estimation

We used the method developed by Lapata and Lascarides (2003) to create the initial list of non-disambiguated interpretations. The parameters of the model were estimated from the British National Corpus (BNC) (Burnard, 2007) that was parsed using the RASP parser of Briscoe et al. (2006). We used the grammatical relations (GRs) output of RASP for BNC created by Andersen et al. (2008). In particular, we extracted all direct and indirect object relations for the nouns from the metonymic phrases, i.e. all the verbs that take the head noun in the compliment as an object (direct or indirect), in order to obtain the counts for $f(o, e)$. Relations expressed in the passive voice and with the use of coordination were also extracted. The verb-object pairs attested in the corpus only once were discarded, as well as the verb *be*, since it does not add any semantic information to the metonymic interpretation. In the case of indirect object relations, the verb was considered to constitute an interpretation together with the preposition, e.g. for the metonymic phrase *enjoy the city* the correct interpretation is *live in* as opposed to *live*.

As the next step we need to identify all possible verb phrase (VP) complements to the metonymic verb (both progressive and infinitive), which represent $f(v, e)$. This was done by searching for `xcomp` relations in the GRs output of RASP, in which our metonymic verb participates in any of its inflected forms. Infinitival and progressive complement counts were summed up to obtain the final frequency $f(v, e)$.

After the frequencies $f(v, e)$ and $f(o, e)$ were obtained, possible interpretations were ranked according to the model of Lapata and Lascarides (2003). The top interpretations for the metonymic

finish video		enjoy book	
Interpretations	Log-prob	Interpretations	Log-prob
view	-19.68	read	-15.68
watch	-19.84	write	-17.47
shoot	-20.58	work on	-18.58
edit	-20.60	look at	-19.09
film on	-20.69	read in	-19.10
film	-20.87	write in	-19.73
view on	-20.93	browse	-19.74
make	-21.26	get	-19.90
edit of	-21.29	re-read	-19.97
play	-21.31	talk about	-20.02
direct	-21.72	see	-20.03
sort	-21.73	publish	-20.06
look at	-22.23	read through	-20.10
record on	-22.38	recount in	-20.13

Table 2: Possible Interpretations of Metonymies Ranked by our System

phrases *enjoy book* and *finish video* together with their log-probabilities are shown in Table 2.

4.2 Comparison with the Results of Lapata and Lascarides

We compared the output of our reimplementation of Lapata and Lascarides' algorithm with their results, which we obtained from the authors. The major difference between the two systems is that we extracted our data from the BNC parsed by RASP, as opposed to the Cass chunk parser (Abney, 1996) utilized by Lapata and Lascarides. Our system finds approximately twice as many interpretations as theirs and covers 80% of their lists (our system does not find some of the low-probability range verbs of Lapata and Lascarides). We compared the rankings of the two implementations in terms of Pearson correlation coefficient and obtained the average correlation of 0.83 (over all metonymic phrases).

We also evaluated the performance of our system against the judgements elicited from humans in the framework of the experiment of Lapata and Lascarides (2003) (for a detailed description of the human evaluation setup see (Lapata and Lascarides, 2003), pages 12-18). The Pearson correlation coefficient between the ranking of our system and the human ranking equals to 0.62 (the intersubject agreement on this task is 0.74). This is slightly lower than the number achieved by Lapata and Lascarides (0.64). Such a difference is probably due to the fact that our system does not find some of the low-probability range verbs that Lapata and Lascarides included in their test set, and thus those interpretations get assigned a probability of 0. We conducted a one-tailed t-test to determine if our counts were significantly different from those of Lapata and Lascarides. The difference is statistically insignificant (t=3.6; df=180; p<.0005), and the output of the system is deemed acceptable to be used for further experiments.

5 Mapping Interpretations to WordNet Senses

The interpretations at this stage are just strings representing collectively all senses of the verb. What we aim for is the list of verb senses that are correct interpretations for the metonymic phrase. We assume the WordNet synset representation of a sense.

It has been recognized (Pustejovsky, 1991; Pustejovsky, 1995; Godard and Jayez, 1993) and verified by us empirically that correct interpretations tend to form semantic classes, and therefore, correct interpretations should be related to each other by semantic relations, such as synonymy or hyponymy. In order to select the right senses of the verbs in the context of the metonymic phrase we did the following.

- We searched the WordNet database for the senses of the verbs that are in synonymy, hypernymy and hyponymy relations.
- We stored the corresponding synsets in a new list of interpretations. If one synset was a hypernym (or hyponym) of the other, then both synsets were stored.

For example, for the metonymic phrase *finish video* the interpretations *watch, view* and *see* are synonymous, therefore a synset containing `(watch(3) view(3) see(7))` was stored. This means that sense 3 of *watch*, sense 3 of *view* and sense 7 of *see* would be correct interpretations of the metonymic expression.

The obtained number of synsets ranges from 14 (*try shampoo*) to 1216 (*want money*) for the whole dataset of Lapata and Lascarides (2003).

6 Ranking the Senses

A problem that arises with the lists of synsets obtained is that they contain different senses of the same verb. However, very few verbs have such a range of meanings that their two different senses could represent two distinct metonymic interpretations (e.g., in case of *take* interpretation of *finish video shoot* sense and *look at, consider* sense are both acceptable interpretations, the second obviously being dispreferred). In the vast majority of cases the occurrence of the same verb in different synsets means that the list still needs filtering.

In order to do this we rank the synsets according to their likelihood of being a metonymic interpretation. The sense ranking is largely based on the probabilities of the verb strings derived by the model of Lapata and Lascarides (2003).

6.1 Zipfian Sense Frequency Distribution

The probability of each string from our initial list represents the sum of probabilities of all senses of this verb. Hence this probability mass needs to be distributed over senses first. The sense frequency distribution for most words tends to be closer to Zipfian, rather than uniform or any other distribution (Preiss, 2006). This is an approximation that we rely on, as it has been shown to realistically describe the majority of words.

This means that the first senses will be favoured over the others, and the frequency of each sense will be inversely proportional to its rank in the list of senses (i.e. sense number, since word senses are ordered in WordNet by frequency).

$$P_{v,k} = P_v \cdot \frac{1}{k} \qquad (2)$$

where k is the sense number and P_v is the likelihood of the verb string being an interpretation according to the corpus data, i.e.

$$P_v = \sum_{k=1}^{N_v} P_{v,k} \qquad (3)$$

where N_v is the total number of senses for the verb in question.

The problem that arises with (2) is that the inverse sense numbers ($1/k$) do not add up to 1. In order to circumvent this, the Zipfian distribution is commonly normalised by the Nth generalised harmonic number. Assuming the same notation

$$P_{v,k} = P_v \cdot \frac{1/k}{\sum_{n=1}^{N_v} 1/n} \qquad (4)$$

Once we have obtained the sense probabilities $P_{v,k}$, we can calculate the likelihood of the whole synset

$$P_s = \sum_{i=1}^{I_s} P_{v_i,k} \qquad (5)$$

where v_i is a verb in the synset s and I_s is the total number of verbs in the synset s. The verbs suggested by WordNet, but not attested in the corpus in the required environment, are assigned the probability of 0. Some output synsets for the metonymic phrase *finish video* and their log-probabilities are demonstrated in Table 3.

In our experiment we compare the performance of the system assuming a Zipfian distribution of senses against the baseline using a uniform distribution. We expect the former to yield better results.

6.2 Gloss Processing

The model in the previous section penalizes synsets that are incorrect interpretations. However, it can not discriminate well between the ones consisting of a single verb. By default it favours the sense with a smaller sense number in WordNet. This poses a problem for the examples such as *direct* for the phrase *finish video*: our list contains several senses of it, as shown in Table 4, and their ranking is not satisfactory. The only correct interpretation in this case, sense 3, is assigned a lower likelihood than the senses 1 and 2.

The most relevant synset can be found by using the information from WordNet glosses (the verbal descriptions of concepts, often with examples). We searched for the glosses containing terms related to the noun in the metonymic phrase, here *video*. Such related terms would be its direct synonyms, hyponyms, hypernyms, meronyms or holonyms according to WordNet. We assigned more weight to the synsets whose gloss contained related terms. In our example the synset `(direct-v-3)`, which is the correct metonymic interpretation, contained the term *film* in its gloss and was therefore selected. Its likelihood was multiplied by the factor of 10.

It should be noted, however, that the glosses do not always contain the related terms; the expectation is that they will be useful in the majority of cases, not in all of them.

7 Evaluation

7.1 The Gold Standard

We selected the most frequent metonymic verbs for our experiments: *begin, enjoy, finish, try, start*. We randomly selected 10 metonymic phrases containing these verbs. We split them into the development set (5 phrases) and the test set (5 phrases)

Synset and its Gloss	Log-prob
(watch-v-1) - look attentively; “watch a basketball game”	-4.56
(view-v-2 consider-v-8 look-at-v-2) - look at carefully; study mentally; ”view a problem”	-4.66
(watch-v-3 view-v-3 see-v-7 catch-v-15 take-in-v-6) - see or watch; ”view a show on television”; ”This program will be seen all over the world”; ”view an exhibition”; ”Catch a show on Broadway”; ”see a movie”	-4.68
(film-v-1 shoot-v-4 take-v-16) - make a film or photograph of something; ”take a scene”; ”shoot a movie”	-4.91
(edit-v-1 redact-v-2) - prepare for publication or presentation by correcting, revising, or adapting; ”Edit a book on lexical semantics”; ”she edited the letters of the politician so as to omit the most personal passages”	-5.11
(film-v-2) - record in film; ”The coronation was filmed”	-5.74
(screen-v-3 screen-out-v-1 sieve-v-1 sort-v-1) - examine in order to test suitability; ”screen these samples”; ”screen the job applicants”	-5.91
(edit-v-3 cut-v-10 edit-out-v-1) - cut and assemble the components of; ”edit film”; ”cut recording tape”	-6.20

Table 3: Metonymy Interpretations as Synsets (for *finish video*)

Synset and its Gloss	Log-prob
(direct-v-1) - command with authority; “He directed the children to do their homework”	-6.65
(target-v-1 aim-v-5 place-v-7 direct-v-2 point-v-11) - intend (something) to move towards a certain goal; ”He aimed his fists towards his opponent’s face”; ”criticism directed at her superior”; ”direct your anger towards others, not towards yourself”	-7.35
(direct-v-3) - guide the actors in (plays and films)	-7.75
(direct-v-4) - be in charge of	-8.04

Table 4: Different Senses of *direct* (for *finish video*)

Development Set	Test Set
enjoy book	enjoy story
finish video	finish project
start experiment	try vegetable
finish novel	begin theory
enjoy concert	start letter

Table 5: Metonymic Phrases in Development and Test Sets

given in the table 5.

The gold standards were created for the top 30 synsets of each metonymic phrase after ranking. This threshold was set experimentally: the recall of correct interpretations among the top 30 synsets is 0.75 (average over metonymic phrases from the development set). This threshold allows to filter out a large number of incorrect interpretations.

The interpretations that are plausible in some imaginary context are marked as correct in the gold standard.

7.2 Evaluation Measure

We evaluated the performance of the system against the gold standard. The objective was to find out if the synsets were distributed in such a way that the plausible interpretations appear at the top of the list and the incorrect ones at the bottom. The evaluation was done in terms of *mean average precision* (MAP) at top 30 synsets.

$$MAP = \frac{1}{M}\sum_{j=1}^{M}\frac{1}{N_j}\sum_{i=1}^{N_j}P_{ji}, \qquad (6)$$

where M is the number of metonymic phrases, N_j is the number of correct interpretations for the metonymic phrase, P_{ji} is the precision at each correct interpretation (the number of correct interpretations among the top i ranks). First, the average precision was computed for each metonymic phrase independently. Then the mean values were calculated for the development and the test sets.

The reasoning behind computing MAP instead of precision at a fixed number of synsets (e.g. top 30) is that the number of correct interpretations varies dramatically for different metonymic phrases. MAP essentially evaluates how many good interpretations appear at the top of the list, which takes this variation into account.

7.3 Results

We compared the ranking obtained by applying Zipfian sense frequency distribution against that obtained by distributing probability mass over senses uniformly (baseline). We also considered the rankings before and after gloss processing. The results are shown in Table 6. These results demonstrate the positive contribution of both Zipfian distribution and gloss processing to the ranking.

7.4 Human Experiment

We conducted an experiment with humans in order to prove that this task is intuitive to people, i.e. they agree on the task.

We had 8 volunteer subjects altogether. All of

Dataset	Verb Probability Mass Distribution	Gloss Processing	MAP
Development set	Uniform	No	0.51
Development set	Zipfian	No	0.65
Development set	Zipfian	Yes	0.73
Test set	Zipfian	Yes	0.83

Table 6: Evaluation of the Model Ranking

Group 1	Group 2
finish video	finish project
start experiment	begin theory
enjoy concert	start letter

Table 7: Metonymic Phrases for Groups 1 and 2

them were native speakers of English and non-linguists. We divided them into 2 groups: 4 and 4. Subjects in each group annotated three metonymic phrases as shown in Table 7. They received written guidelines, which were the only source of information on the experiment.

For each metonymic phrase they were presented with a list of 30 possible interpretations produced by the system. For each synset in the list they had to decide whether it was a plausible interpretation of the metonymic phrase in an imaginary context.

We evaluated interannotator agreement in terms of Fleiss' kappa (Fleiss, 1971) and f-measure computed pairwise and then averaged across the annotators. The agreement in group 1 was 0.76 (f-measure) and 0.56 (kappa); in group 2 0.68 (f-measure) and 0.51 (kappa). This yielded the average agreement of 0.72 (f-measure) and 0.53 (kappa).

8 Linguistic Perspective on Logical Metonymy

There has been debate in linguistics literature as whether it is the noun or the verb in the metonymic phrase that determines the interpretation. Some of the accounts along with our own analysis are presented below.

8.1 The Effect of the Noun Complement

The interpretation of logical metonymy is often explained by the lexical defaults associated with the noun complement in the metonymic phrase. Pustejovsky (1991) models these lexical defaults in the form of the *qualia structure* of the noun. The qualia structure of a noun specifies the following aspects of its meaning:

- CONSTITUTIVE Role (the relation between an object and its constituents)
- FORMAL Role (that which distinguishes the object within a larger domain)
- TELIC Role (purpose and function of the object)
- AGENTIVE Role (how the object came into being)

For the problem of logical metonymy the telic and agentive roles are of particular interest. For example, the noun *book* would have *read* specified as its telic role and *write* as its agentive role in its qualia structure. Following Pustejovsky (1991; 1995) and others, we take this information from the noun qualia to represent the default interpretations of metonymic constructions. Nevertheless, multiple telic and agentive roles can exist and be valid interpretations, which is supported by the evidence derived from the corpus (Verspoor, 1997).

Such lexical defaults operate with a lack of pragmatic information. In some cases, however, lexical defaults can be overridden by context. Consider the following example taken from Lascarides and Copestake (1995).

(6) My goat eats anything. He really enjoyed your book.

Here it is clear that *the goat enjoyed eating the book* and not *reading the book*, which is enforced by the context. Thus, incorporating the context of the metonymic phrase into the model would be another interesting extension of our experiment.

8.2 The Effect of the Metonymic Verb

By analysing phrases from the dataset of Lapata and Lascarides (2003) we found that different metonymic verbs have different effect on the interpretation of logical metonymy. In this section we provide some criteria based on which one could classify metonymic verbs:

- ***Control* vs. *raising***. Consider the phrase *expect poetry* taken from the dataset of Lapata and Lascarides. *Expect* is a typical object raising verb and, therefore, the most obvious interpretation of this phrase would be *expect someone to learn/recite poetry*, rather than *expect to hear poetry* or *expect to learn poetry*, as suggested by the model of Lapata and Lascarides. Their model does not take into account raising syntactic frame and as such its interpretation of raising metonymic phrases will be based on the wrong kind of corpus evidence. Our expectation, however, is that control verbs tend to form logical metonymies more frequently. By analyzing the lists of control and raising verbs compiled by Boguraev and Briscoe (1987) we found evidence supporting this claim. Only 20% of raising verbs can form metonymic constructions (e.g. *expect, allow, command, request, require* etc.), while others can not (e.g. *appear, seem, consider* etc.). Due to both this and the fact that we build on the approach of Lapata and Lascarides (2003), we gave preference to control verbs to develop and test our system.

- ***Activity*** **vs.** ***result***. Some metonymic verbs require the reconstructed event to be an *activity* (e.g. *begin writing the book*), while others require a *result* (e.g. *attempt to reach the peak*). This distinction potentially allows to rule out some incorrect interpretations, e.g. a resultative *find* for *enjoy book*, as enjoy requires an event of the type *activity*. Automating this would be an interesting route for extension of our experiment.

- ***Telic*** **vs.** ***agentive*** **vs.** ***other*** events. Another interesting observation we made captures the constraints that the metonymic verb imposes on the reconstructed event in terms of its function. While some metonymic verbs require rather *telic* events (e.g., *enjoy, want, try*), others have strong preference for *agentive* (e.g., *start*). However, for some categories of verbs it is hard to define a particular type of the event they require (e.g., *attempt the peak* should be interpreted as *attempt to reach the peak*, which is neither telic nor agentive).

9 Conclusions and Future Work

We presented a system producing disambiguated interpretations of logical metonymy with respect to word sense. Such representation is novel and it is intuitive to humans, as demonstrated by the human experiment. We also proposed a novel scheme for estimating the likelihood of a WordNet synset as a unit from a non-disambiguated corpus. The obtained results demonstrate the effectiveness of our approach to deriving metonymic interpretations.

Along with this we provided criteria for discriminating between different metonymic verbs with respect to their effect on the interpretation of logical metonymy. Our empirical analysis has shown that control verbs tend to form logical metonymy more frequently than raising verbs, as well as that the former comply with the model of Lapata and Lascarides (2003), whereas the latter form logical metonymies based on a different syntactic frame. Incorporating such linguistic knowledge into the model would be an interesting extension of this experiment.

One of the motivations of the proposed sense-based representation is the fact that the interpretations of metonymic phrases tend to form coherent semantic classes (Pustejovsky, 1991; Pustejovsky, 1995; Godard and Jayez, 1993). The automatic discovery of such classes would require word sense disambiguation as an initial step. This is due to the fact that it is verb senses that form the classes rather than verb strings. Comparing the interpretations obtained for the phrase *finish video*, one can clearly distinguish between the meaning pertaining to the creation of the video, e.g., *film, shoot, take*, and those denoting using the video, e.g., *watch, view, see*. Discovering such classes using the existing verb clustering techniques is our next experiment.

Using sense-based interpretations of logical metonymy as opposed to ambiguous verbs could benefit other NLP applications that rely on disambiguated text (e.g. for the tasks of information retrieval (Voorhees, 1998) and question answering (Pasca and Harabagiu, 2001)).

Acknowledgements

I would like to thank Simone Teufel and Anna Korhonen for their valuable feedback on this project and my anonymous reviewers whose comments helped to improve the paper. I am also very grateful to Cambridge Overseas Trust who made this research possible by funding my studies.

References

S. Abney. 1996. Partial parsing via finite-state cascades. In J. Carroll, editor, *Workshop on Robust Parsing*, pages 8–15, Prague.

O. E. Andersen, J. Nioche, E. Briscoe, and J. Carroll. 2008. The BNC parsed with RASP4UIMA. In *Proceedings of the Sixth International Language Resources and Evaluation Conference (LREC'08)*, Marrakech, Morocco.

B. Boguraev and E. Briscoe. 1987. Large lexicons for natural language processing: utilising the grammar coding system of the *Longman Dictionary of Contemporary English*. *Computational Linguistics*, 13(4):219–240.

E. Briscoe, A. Copestake, and B. Boguraev. 1990. Enjoy the paper: lexical semantics via lexicology. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90)*, pages 42–47, Helsinki.

E. Briscoe, J. Carroll, and R. Watson. 2006. The second release of the rasp system. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 77–80.

L. Burnard. 2007. *Reference Guide for the British National Corpus (XML Edition)*.

D. Fass. 1991. met*: A method for discriminating metonymy and metaphor by computer. *Computational Linguistics*, 17(1):49–90.

C. Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database (ISBN: 0-262-06197-X)*. MIT Press, first edition.

J. L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

D. Godard and J. Jayez. 1993. Towards a proper treatment of coercion phenomena. In *Sixth Conference of the European Chapter of the ACL*, pages 168–177, Utrecht.

M. Lapata and A. Lascarides. 2003. A probabilistic account of logical metonymy. *Computational Linguistics*, 29(2):261–315.

A. Lascarides and A. Copestake. 1995. The pragmatics of word meaning. In *Journal of Linguistics*, pages 387–414.

M. Pasca and S. Harabagiu. 2001. The informative role of WordNet in open-domain question answering. In *Proceedings of NAACL-01 Workshop on WordNet and Other Lexical Resources*, pages 138–143, Pittsburgh, PA.

J. Preiss. 2006. Probabilistic word sense disambiguation analysis and techniques for combining knowledge sources. Technical report, Computer Laboratory, University of Cambridge.

J. Pustejovsky. 1991. The generative lexicon. *Computational Linguistics*, 17(4).

J. Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge, MA.

M. Utiyama, M. Masaki, and I. Hitoshi. 2000. A statistical approach to the processing of metonymy. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrucken, Germany.

C. M. Verspoor. 1997. Conventionality-governed logical metonymy. In *Proceedings of the Second International Workshop on Computational Semantics*, pages 300–312, Tilburg.

E. M. Voorhees. 1998. Using WordNet for text retrieval. In C. Fellbaum, editor, *WordNet: An Electornic Lexical Database*, pages 285–303. MIT Press.

Insights into Non-projectivity in Hindi

Prashanth Mannem, Himani Chaudhry, Akshar Bharati
Language Technologies Research Center,
International Institute of Information Technology,
Gachibowli, Hyderabad, India - 500032
{prashanth,himani}@research.iiit.ac.in

Abstract

Large scale efforts are underway to create dependency treebanks and parsers for Hindi and other Indian languages. Hindi, being a morphologically rich, flexible word order language, brings challenges such as handling non-projectivity in parsing. In this work, we look at non-projectivity in Hyderabad Dependency Treebank (HyDT) for Hindi. Non-projectivity has been analysed from two perspectives: graph properties that restrict non-projectivity and linguistic phenomenon behind non-projectivity in HyDT. Since Hindi has ample instances of non-projectivity (14% of all structures in HyDT are non-projective), it presents a case for an in depth study of this phenomenon for a better insight, from both of these perspectives.

We have looked at graph constriants like planarity, gap degree, edge degree and well-nestedness on structures in HyDT. We also analyse non-projectivity in Hindi in terms of various linguistic parameters such as the causes of non-projectivity, its *rigidity* (possibility of reordering) and whether the reordered construction is the *natural* one.

1 Introduction

Non-projectivity occurs when dependents do not either immediately follow or precede their heads in a sentence (Tesnire, 1959). These dependents may be spread out over a discontinuous region of the sentence. It is well known that this poses problems for both theoretical grammar formalisms as well as parsing systems. (Kuhlmann and Möhl, 2007; McDonald and Nivre, 2007; Nivre et al., 2007)

Hindi is a verb final, flexible word order language and therefore, has frequent occurrences of non-projectivity in its dependency structures. Bharati et al. (2008a) showed that a major chunk of errors in their parser is due to non-projectivity. So, there is a need to analyse non-projectivity in Hindi for a better insight into such constructions. We would like to say here, that as far as we are aware, there hasn't been any attempt to study non-projectivity in Hindi before this work. Our work is a step forward in this direction.

Non-projectivity can be analysed from two aspects. a) In terms of graph properties which restrict non-projectivity and b) in terms of linguistic phenomenon giving rise to non-projectivity. While a) gives an idea of the kind of grammar formalisms and parsing algorithms required to handle non-projective cases in a language, b) gives an insight into the linguistic cues necessary to identify non-projective sentences in a language.

Parsing systems can explore algorithms and make approximations based on the coverage of these graph properties on the treebank and linguistic cues can be used as features to restrict the generation of non-projective constructions (Shen and Joshi, 2008). Similarly, the analyses based on these aspects can also be used to come up with broad coverage grammar formalisms for the language.

Graph constraints such as *projectivity*, *planarity*, *gap degree*, *edge degree* and *well-nestedness* have been used in previous works to look at non-projective constructions in treebanks like PDT and DDT (Kuhlmann and Nivre, 2006; Nivre, 2006). We employ these constraints in our work too. Apart from these graph constraints, we also look at non-projective constructions in terms of various parameters like factors leading to non-projectivity, its rigidity (see Section 4), its approximate projective construction and whether its the natural one.

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 10–17,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

In this paper, we analyse dependency structures in Hyderabad Dependency Treebank (HyDT). HyDT is a pilot treebank containing dependency annotations for 1865 Hindi sentences. It uses the annotation scheme proposed by Begum et al. (2008), based on the Paninian grammar formalism.

This paper is organised as follows: In section 2, we give an overview of HyDT and the annotation scheme used. Section 3 discusses the graph properties that are used in our analysis and section 4 reports the experimental results on the coverage of these properties on HyDT. The linguistic analysis of non-projective constructions is discussed case by case in Section 5. The conclusions of this work are presented in section 6. Section 7 gives directions for future works on non-projectivity for Hindi.

2 Hyderabad Dependency Treebank (HyDT)

HyDT is a dependency annotated treebank for Hindi. The annotation scheme used for HyDT is based on the Paninian framework (Begum et al., 2008). The dependency relations in the treebank are syntactico-semantic in nature where the main verb is the central binding element of the sentence. The arguments including the adjuncts are annotated taking the meaning of the verb into consideration. The participants in an action are labeled with *karaka* relations (Bharati et al., 1995). Syntactic cues like case-endings and markers such as post-positions and verbal inflections, help in identifying appropriate *karakas*.

The dependency tagset in the annotation scheme has 28 relations in it. These include six basic karaka relations (adhikarana [*location*], apaadaan [*source*], sampradaan [*recipient*], karana [*instrument*], karma [*theme*] and karta [*agent*]). The rest of the labels are non-karaka labels like vmod, adv, nmod, rbmod, jjmod etc...[1] The tagset also includes special labels like *pof* and *ccof*, which are not dependency relations in the strict sense. They are used to handle special constructions like conjunct verbs (ex:- *prashna kiyaa* (`question did`)), coordinating conjunctions and ellipses.

In the annotation scheme used for HyDT, relations are marked between chunks instead of words. A chunk (with boundaries marked) in HyDT, by definition, represents a set of adjacent words which are in dependency relation with each other, and are connected to the rest of the words by a single incoming dependency arc. The relations among the words in a chunk are not marked. Thus, in a dependency tree in HyDT, each node is a chunk and the edge represents the relations between the connected nodes labeled with the karaka or other relations. All the modifier-modified relations between the heads of the chunks (inter-chunk relations) are marked in this manner. The annotation is done using Sanchay[2] mark up tool in Shakti Standard Format (SSF) (Bharati et al., 2005). For the work in this paper, to get the complete dependency tree, we used an automatic rule based intra-chunk relation identifier. The rules mark these intra-chunk relations with an accuracy of 99.5%, when evaluated on a test set.

The treebank has 1865 sentences with a total of 16620 chunks and 35787 words. Among these, 14% of the sentences have non-projective structures and 1.87% of the inter-chunk relations are non-projective. This figure drops to 0.87% if we consider the intra-chunk relations too (as all intra-chunk relations are projective). In comparison, treebanks of other flexible word order languages like Czech and Danish have non-projectivity in 23% (out of 73088 sentences) and 15% (out of 4393 sentences) respectively (Kuhlmann and Nivre, 2006; Nivre et al., 2007).

3 Non projectivity and graph properties

In this section, we define dependency graph formally and discuss standard propertiess uch as single headedness, acyclicity and projectivity. We then look at complex graph constraints like gap degree, edge degree, planarity and well-nestedness which can be used to restrict non-projectivity in graphs.

In what follows, a dependency graph for an input sequence of words $x_1 \cdots x_n$ is an unlabeled directed graph $D = (X, Y)$ where X is a set of nodes and Y is a set of directed edges on these nodes. $x_i \rightarrow x_j$ denotes an edge from x_i to x_j, $(x_i, x_j) \in Y$. $\rightarrow^*$ is used to denote the reflexive and transitive closure of the relation. $x_i \rightarrow^* x_j$ means that the node x_i *dominates* the node x_j, i.e., there is a (possibly empty) path from x_i to x_j. $x_i \leftrightarrow x_j$ denotes an edge from x_i to x_j or vice

[1] The entire dependency tagset can be found at http://ltrc.deptagset.googlepages.com/k1.htm

[2] http://sourceforge.net/projects/nlp-sanchay

versa. For a given node x_i, the set of nodes dominated by x_i is the *projection* of x_i. We use $\pi(x_i)$ to refer to the projection of x_i arranged in ascending order.

Every dependency graph satisfies two constraints: acyclicity and single head. *Acyclicity* refers to there being no cycles in the graph. *Single head* refers to each node in the graph D having exactly one incoming edge (except the one which is at the root). While acyclicity and single head constraints are satisfied by dependency graphs in almost all dependency theories. Projectivity is a stricter constraint used and helps in reducing parsing complexities.

Projectivity: If node x_k depends on node x_i, then all nodes between x_i and x_k are also subordinate to x_i (i.e dominated by x_i) (Nivre, 2006).

$$x_i \rightarrow x_k \;\; \Rightarrow \;\; x_i \rightarrow^* x_j$$

$$\forall x_j \in X : (x_i < x_j < x_k \;\; \vee \;\; x_i > x_j > x_k)$$

Any graph which doesn't satisfy this constraint is *non-projective*. Unlike acyclicity and the single head constraints, which impose restrictions on the dependency relation as such, projectivity constrains the interaction between the dependency relations and the order of the nodes in the sentence (Kuhlmann and Nivre, 2006)..

Graph properties like *planarity*, *gap degree*, *edge degree* and *well-nestedness* have been proposed in the literature to constrain grammar formalisms and parsing algorithms from looking at unrestricted non-projectivity. We define these properties formally here.

Planarity: A dependency graph is *planar* if edges do not cross when drawn above the sentence (Sleator and Temperley, 1993). It is similar to projectivity except that the arc from dummy node at the beginning (or the end) to the root node is not considered.

$$\forall (x_i, x_j, x_k, x_l) \in X,$$

$$\neg((x_i \leftrightarrow x_k \wedge x_j \leftrightarrow x_l) \wedge \; (x_i < x_j < x_k < x_l))$$

Gap degree: The gap degree of a node is the number of gaps in the projection of a node. A gap is a pair of nodes ($\pi(x_i)_k$, $\pi(x_i)_{k+1}$) adjacent in $\pi(x_i)$ but not adjacent in sentence. The gap degree of node $Gd(x_i)$ is the number of such gaps in its projection. The gap degree of a sentence is the maximum among gap degrees of nodes in $D(X, Y)$ (Kuhlmann, 2007).

Edge degree: The number of connected components in the span of an edge which are not dominated by the outgoing node in the edge. Span $span(x_i \rightarrow x_j)$ = $(min(i, j), max(i, j))$. $Ed(x_i \rightarrow x_j)$ is the number of connected componenets in the span $span(x_i \rightarrow x_j)$ whose parent is not in the projection of x_i. The edge degree of a sentence is the maximum among edge degrees of edges in $D(X, Y)$. (Nivre, 2006) defines it as degree of non-projectivity. Following (Kuhlmann and Nivre, 2006), we call this edge degree to avoid confusion.

Well-nested: A dependency graph is well-nested if no two disjoint subgraphs *interleave* (Bodirsky et al., 2005). Two subgraphs are disjoint if neither of their roots dominates the other. Two subtrees S_i,S_j interleave if there are nodes $x_l, x_m \in S_i$ and $x_n, x_o \in S_j$ such that $l < m < n < o$ (Kuhlmann and Nivre, 2006).

The gap degree and the edge degree provide a quantitative measure for the non-projectivity of dependency structures. Well-nestedness is a qualitative property: it constrains the relative positions of disjoint subtrees.

4 Experiments on HyDT

Property	*Count*	*Percentage*
All structures	1865	
Gap degree		
Gd(0)	1603	85.9%
Gd(1)	259	13.89%
Gd(2)	0	0%
Gd(3)	3	0.0016%
Edge degree		
Ed(0)	1603	85.9%
Ed(1)	254	13.6%
Ed(2)	6	0.0032%
Ed(3)	1	0.0005%
Ed(4)	1	0.0005%
Projective	1603	85.9%
Planar	1639	87.9%
Non-projective & planar	36	1.93%
Well-nested	1865	100%

Table 1: Results on HyDT

In this section, we present an experimental evaluation of the graph constraints mentioned in the previous section on the dependency structures in

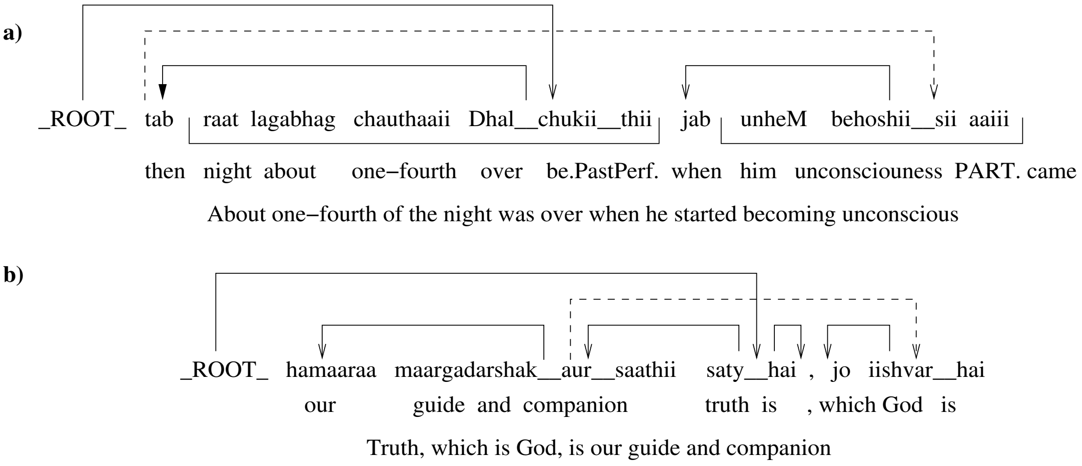

a)
ROOT tab raat lagabhag chauthaaii Dhal__chukii__thii jab unheM behoshii__sii aaiii
then night about one−fourth over be.PastPerf. when him unconsciouness PART. came
About one−fourth of the night was over when he started becoming unconscious
b)
ROOT hamaaraa maargadarshak__aur__saathii saty__hai , jo iishvar__hai
our guide and companion truth is , which God is
Truth, which is God, is our guide and companion

Figure 1: a) Relative co-relative construction, b) Extraposed relative clause construction

HyDT. Since HyDT is a small corpus and is still under construction, these results might not be the exact reflection of naturally occurring sentences in real-world. Nevertheless, we hope these results will give an idea of the kind of structures one can expect in Hindi.

We report the percentage of structures that satisfy various graph properties in table 1. In HyDT, we see that 14% of all structures are non-projective. The highest gap degree for structures in HyDT is 3 and in case of edge degree, it is 4. Only 3 structures (1.5% approx.) have gap degree of more than 1 in a total of 262 non-projective sentences. When it comes to edge degree, only 8 structures (3%) have edge degree more than 1.

The difference in the coverage of gap degree 1 & 2 (and the fact that gap degree 1 accounts for 13.9% of the structures) shows that a parser should handle non-projective constructions at least till gap degree 1 for good coverage. The same can be said about edge degree.

5 Cases of non-projectivity in HyDT

We have carried out a study of the instances of non-projectivity that HyDT brought forth. In this section, we classify these instances based on factors leading to non-projectivity and present our analysis of them. For each of these classes, we look at the *rigidity* of these non-projective constructions and their best projective approximation possible by reordering. Rigidity here is the reorderability of the constructions retaining the gross meaning. *Gross meaning* refers to the meaning of the sentence not taking the discourse and topic-focus into consideration, which is how parsing is typically done.

e.g., the non-projective construction in figure 1b, `yadi rupayoM kii zaruurat thii to mujh ko bataanaa chaahiye thaa`[3] can be reordered to form a projective construction `mujh ko bataanaa chaahiye thaa yadi rupayoM kii zaruurat thii to`. Therefore, this sentence is not rigid.

Study of rigidity is important from natural language generation perspective. Sentence generation from projective structures is easier and more efficient than from non-projective ones. Non-projectivity in constructions that are non-rigid can be effectively dealt with through projectivisation.

Further, we see if these approximations are more *natural* compared to the non-projective ones as this impacts sentence generation quality. A natural construction is the one most preferred by native speakers of that language. Also, it more or less abides by the well established rules and patterns of the language.

We observed that non-projectivity is caused in Hindi, due to various linguistic phenomena manifested in the language, such as relative co-relative constructions, paired connectives, complex coordinating structures, interventions in verbal arguments by non-verbal modifiers, shared arguments in non-finite clauses, movement of modifiers, ellipsis etc. Also, non-projectivity in Hindi can occur within a clause (*intra-clausal*) as well as between elements across clauses (*inter-clausal*).

We now discuss some of these linguistic phenomena causing non-projectivity.

[3] The glosses for the sentences in this section are listed in the corresponding figures and are not repeated to save space.

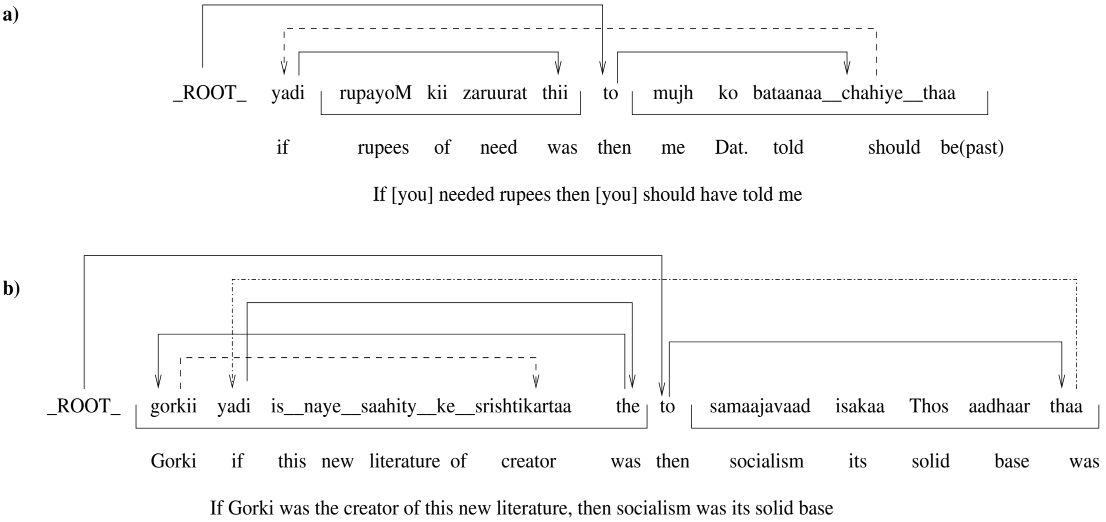

a)
ROOT yadi rupayoM kii zaruurat thii to mujh ko bataanaa__chahiye__thaa
if rupees of need was then me Dat. told should be(past)
If [you] needed rupees then [you] should have told me
b)
ROOT gorkii yadi is__naye__saahity__ke__srishtikartaa the to samaajavaad isakaa Thos aadhaar thaa
Gorki if this new literature of creator was then socialism its solid base was
If Gorki was the creator of this new literature, then socialism was its solid base

Figure 2: a) Paired connectives construction, b) Construction with non-projectivity within a clause

5.1 Relative co-relative constructions

The pattern in co-relatives is that a demonstrative pronoun, which also functions as determiner in Hindi, such as vo (*that*), always occurs in correlation with a relative pronoun, jo (*which*). In fact, the language employs a series of such pronouns : e.g., jis-us '*which-that*', jahaaM-vahaaM '*where-there*', jidhar-udhar '*where-there*', jab-tab '*when-then*', aise-jaise (Butt et al., 2007).

Non-projectivity is seen to occur in relative co-relative constructions with pairs such as jab-tab, if the clause beginning with the tab precedes the jab clause as seen in figure 1a. If the clause with the relative pronoun comes before the clause with the demonstrative pronoun, non-projectivity can be ruled out. So, this class of non-projective constructions is not rigid since projective structures can be obtained by reordering without any loss of meaning. The projective case is relatively more natural than the non-projective one. This is reaffirmed in the corpus where the projective relative co-relative structures are more frequent than the non-projective sentences.

In the example in figure 1a, the sentence can be reordered by moving the tab clause to the right of the jab clause, to remove non-projectivity.

jab unheM behoshii sii aaii tab raat lagabhag chauthaaii Dhal chukii thii – *when he started becoming unconscious, about one-fourth of the night was over*

5.2 Extraposed relative clause constructions

If the relative clause modifying a noun phrase (NP) occurs after the verb group (VP), it leads to non-projectivity.

In the sentence in figure 1b, non-projectivity occurs because jo iishvar hai, the relative clause modifying the NP hamaaraa maargadarshak aur saathii is extraposed after the VP saty hai.

This class of constructions is not rigid as the extraposed relative clause can be moved next to the noun phrase, making it projective. However, the resulting projective construction is less natural than the original non-projective one.

The reordered projective construction for the example sentence is hamaaraa maargadarshak aur saathii, jo iishvar hai, saty hai – *Our guide and companion which is God is truth*

This class of non-projective constructions accounts for approximately half of the total non-projective sentences in the treebank.

5.3 Intra-clausal non-projectivity

In this case, the modifier of the NP is a non-relative clause and is different from the class 5.2.

In the example in figure 2b, the NP gorkii and the phrase modifying it is naye saahity ke srishtikartaa are separated by yadi, a modifier of to clause. Intra-clausal non-projectivity here is within the clause gorkii yadi is naye saahity ke srishtikartaa the.

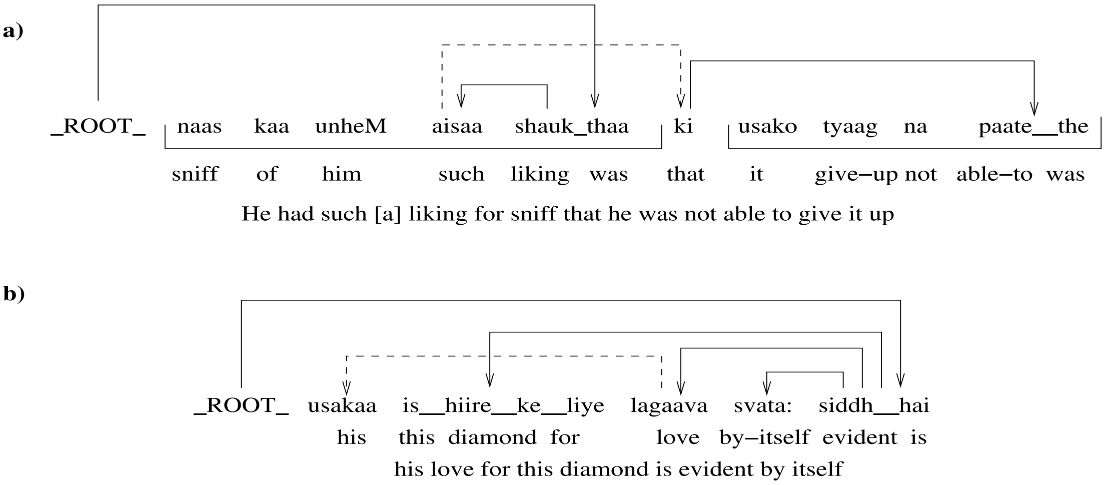

a)
ROOT
naas kaa unheM aisaa shauk_thaa ki usako tyaag na paate__the
sniff of him such liking was that it give−up not able−to was
He had such [a] liking for sniff that he was not able to give it up
b)
ROOT usakaa is__hiire__ke__liye lagaava svata: siddh__hai
his this diamond for love by−itself evident is
his love for this diamond is evident by itself

Figure 3: a) ki complement clause, b) Genetive relation split by a verb modifier

To remove non-projectivity, reordering of such sentences is possible by moving the non-modifier, so that it no more separates them. Here, moving yadi to the left of gorkii takes care of non-projectivity thus making this class not rigid. The reordered projective construction is more natural.

yadi gorkii is naye saahity ke srishtikartaa the to samaajavaad isakaa Thos aadhaar thaa

5.4 Paired connectives

Paired connectives (such as agar-to *'if-then'*, yadi-to *'if-then'*) give rise to non-projectivity in HyDT on account of the annotation scheme used.

As shown in figure 2a, the to clause is modified by the yadi clause in such constructions. Most of these sentences can be reordered while still retaining the meaning of the sentence: the phrase that comes after to, followed by yadi clause, and then to. Here mentioning to is optional.

This sentence can be reordered and is not rigid. However, the resulting projective construction is not a natural one. mujh ko bataanaa chaahiye thaa yadi rupayoM kii zaruurat thii [to] – *(you) should have told me if (you) needed rupees*

Connectives like yadi can also give rise to intra-clausal non-projectivity apart from inter-clausal non-projectivity as discussed. This happens when the connective moves away from the beginning of the sentence (see figure 2b).

5.5 ki complement clause

A phrase (including a VP in it) appears between the ki (*that*) clause and the word it modifies (such as yaha (*this*), asiaa (*such*), is tarah (*such*), itana (*this much*)), resulting in non-projectivity in the ki complement constructions. The verb in this verb group is generally copular. Since Hindi is a verb final language, the complementiser clause (ki clause) occurs after the verb of the main clause, while its referent lies before the verb in the main clause. This leads to non-projectivity in such constructions. The yaha-ki constructions follow the pattern: yaha-*its property*-VP-ki clause.

E.g. yaha-rahasya-hai-ki shukl jii pratham shreNii ke kavi kyoM the.

This class of constructions are rigid and non-projectivity can't be removed from such sentences. In cases where the VP has a transitive verb, the ki clause and its referent, both modify the verb, making the construction projective. For ex. In usane yaha kahaa ki vaha nahin aayegaa, yaha and the ki clause both modify the verb kahaa.

In figure 3a, the phrase shauk thaa separates aisaa and the ki clause, resulting in non-projectivity.

5.6 A genetive relation split by a verb modifier

This is also a case of intra-clausal non-projectivity. In such constructions, the verb has its modifier embedded within the genetive construction.

In the example in figure 3b, the components of the genetive relation, usakaa and lagaav are separated by the phrase is hiire ke liye.

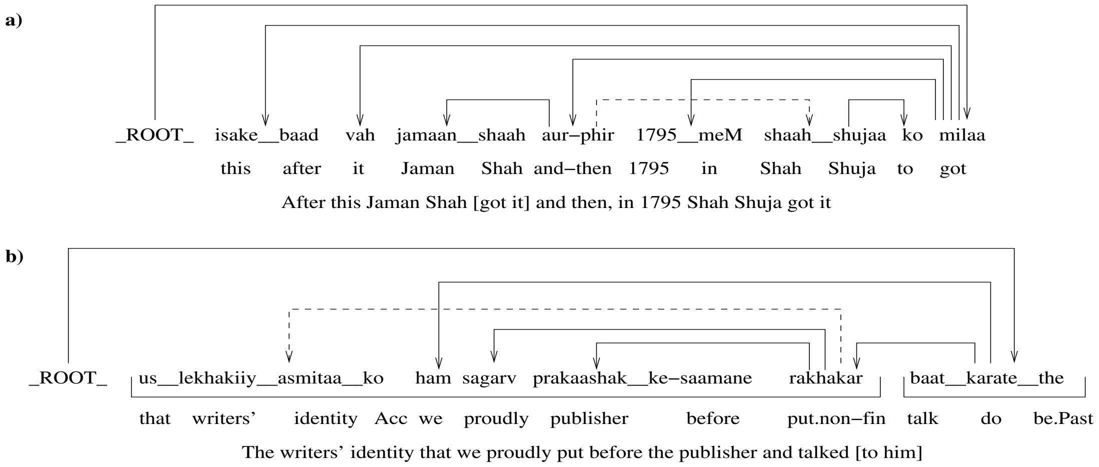


Figure 4: a) A phrase splitting a co-ordinating structure, b) Shared argument splitting the non finite clause

The sentence is not rigid and can be reordered to a projective construction by moving the phrase is hiire ke liye to the left of usakaa. It retains the meaning of the original construction and is also, a more natural one.

is hiire ke liye usakaa lagaav svata: siddh hai – *his love for this diamond is evident by itself*

5.7 A phrase splitting a co-ordinating structure

As seen in figure 4a, non-projectivity is caused in the sentence because, embedding of the phrase 1795 meM splits the co-ordinating structure jamaan shaah aur-phir shaah shujaa. These kinds of constructions can be reordered. So, they are not rigid. The projective constructions are more natural.

isake baad vah jamaan shaah ko aur-phir shaah shujaa ko 1795 meM milaa

Non-projective Class	*Count*	%
Relative co-relatives constructions	18	6.8 %
Extraposed realtive clause constructions	101	38.0 %
Intra-clausal non-projectivity	12	4.5 %
Paired connectives	33	12.4 %
ki complement clauses	52	19.5 %
Genetive relation split by a verb modifier	10	3.8 %
Phrase splitting a co-ordinating structure	4	1.5 %
Shared argument splits the non-finite clause	10	3.8 %
Others	26	9.8 %

Table 2: Non-projectivity class distribution in HyDT

5.8 Shared argument splits the non finite clause

In the example in 4b, hama is annotated as the argument of the main verb baawa karate the. It also is the shared argument of the non finite verb rakhakara (but isn't marked explicitly in the treebank). It splits the non finite clause us lekhakiiya asmitaa ko **ham** sagarv prakaashak ke saamane rakhakara

Through reordering, this sentence can easily be made into a projective construction, which is also the more natural construction for it.

ham us lekhakiiy asmitaa ko sagarv prakaashak ke-saamane rakhakar baat karate the

5.9 Others

There are a few non-projective constructions in HyDT which haven't been classified and discussed in the eight categories above. This is because they are single occurences in HyDT and seem to be rare phenomenon. There are also a few instances of inconsistent *NULL* placement and errors in chunk boundary marking or annotation.

6 Conclusion

Our study of HyDT shows that non-projectivity in Hindi is more or less confined to the classes discussed in this paper. There might be more types of non-projective structures in Hindi which may not have occurred in the treebank.

Recent experiments on Hindi dependency parsing have shown that non-projective structures form a major chunk of parsing errors (Bharati et al., 2008a). In spite of using state-of-art parsers which handle non-projectivity, experiments show that the types of non-projectivity discussed in this paper are not handled effectively.

The knowledge of such non-projective classes could possibly be used to enhance the performance of a parser. This work further corroborates Kuhlmann's work on Czech (PDT) for Hindi (Kuhlmann and Nivre, 2006). Specifically, as discussed in section 4, the non-projective structures in HyDT satisfy the constraints (gap degree ≤ 2 and well-nestedness) to be called as mildly non-projective.

7 Future Work

We propose to use the analysis in this paper to come up with non-projective parsers for Hindi. This can be done in more than one ways, such as:

The constraint based dependency parser for Hindi proposed in (Bharati et al., 2008b) can be extended to incorporate graph properties discussed in section 3 as constraints.

Further, linguistic insights into non-projectivity can be used in parsing to identify when to generate the non-projective arcs. The parser can have specialised machinery to handle non-projectivity only when linguistic cues belonging to these classes are active. The advantage of this is that one need not come up with formal complex parsing algorithms which give unrestricted non-projective structures.

As the HyDT grows, we are bound to come across more instances as well as more types of non-projective constructions that could bring forth interesting phenomenon. We propose to look into these for further insights.

References

R. Begum, S. Husain, A. Dhwaj, D. Sharma, L. Bai, and R. Sangal. 2008. Dependency annotation scheme for indian languages. In *In Proceedings of The Third International Joint Conference on Natural Language Processing (IJCNLP)*, Hyderabad, India.

Akshar Bharati, Vineet Chaitanya, and Rajeev Sangal. 1995. *Natural Language Processing: A Paninian Perspective*. Prentice-Hall of India.

Akshar Bharati, Rajeev Sangal, and Dipti Sharma. 2005. Shakti analyser: Ssf representation. Technical report, International Institute of Information Technology, Hyderabad, India.

Akshar Bharati, Samar Husain, Bharat Ambati, Sambhav Jain, Dipti Sharma, and Rajeev Sangal. 2008a. Two semantic features make all the difference in parsing accuracy. In *Proceedings of the 6th International Conference on Natural Language Processing (ICON-08)*, Pune, India.

Akshar Bharati, Samar Husain, Dipti Sharma, and Rajeev Sangal. 2008b. A two-stage constraint based dependency parser for free word order languages. In *Proceedings of the COLIPS International Conference on Asian Language Processing 2008 (IALP)*, Chiang Mai, Thailand.

Manuel Bodirsky, Marco Kuhlmann, and Mathias Mhl. 2005. Well-nested drawings as models of syntactic structure. In *In Tenth Conference on Formal Grammar and Ninth Meeting on Mathematics of Language*, pages 88–1. University Press.

M. Butt, T. H. King, and S. Roth. 2007. Urdu correlatives: Theoretical and implementational issues. In *Online Proceedings of the LFG07 Conference*, pages 87–106. CSLI Publications.

Marco Kuhlmann and Mathias Möhl. 2007. Mildly context-sensitive dependency languages. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 160–167, Prague, Czech Republic, June. Association for Computational Linguistics.

Marco Kuhlmann and Joakim Nivre. 2006. Mildly non-projective dependency structures. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 507–514, Sydney, Australia, July. Association for Computational Linguistics.

Marco Kuhlmann. 2007. *Dependency Structures and Lexicalized Grammars*. Ph.D. thesis, Saarland University.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic, June. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June. Association for Computational Linguistics.

Joakim Nivre. 2006. Constraints on non-projective dependency parsing. In *In Proceedings of European Association of Computational Linguistics (EACL)*, pages 73–80.

Libin Shen and Aravind Joshi. 2008. LTAG dependency parsing with bidirectional incremental construction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 495–504, Honolulu, Hawaii, October. Association for Computational Linguistics.

Daniel Sleator and Davy Temperley. 1993. Parsing english with a link grammar. In *In Third International Workshop on Parsing Technologies*.

L. Tesnire. 1959. *lments de Syntaxe Structurale*. Libraire C. Klincksieck, Paris.

Annotating and Recognising Named Entities in Clinical Notes

Yefeng Wang
School of Information Technology
The University of Sydney
Australia 2006
ywang1@it.usyd.edu.au

Abstract

This paper presents ongoing research in clinical information extraction. This work introduces a new genre of text which are not well-written, noise prone, ungrammatical and with much cryptic content. A corpus of clinical progress notes drawn form an Intensive Care Service has been manually annotated with more than 15000 clinical named entities in 11 entity types. This paper reports on the challenges involved in creating the annotation schema, and recognising and annotating clinical named entities. The information extraction task has initially used two approaches: a rule based system and a machine learning system using Conditional Random Fields (CRF). Different features are investigated to assess the interaction of feature sets and the supervised learning approaches to establish the combination best suited to this data set. The rule based and CRF systems achieved an F-score of 64.12% and 81.48% respectively.

1 Introduction

A substantial amount of clinical data is locked away in a non-standardised form of clinical language, which if standardised could be usefully mined to improve processes in the work of clinical wards, and to gain greater understanding of patient care as well as the progression of diseases. However in some clinical contexts these clinical notes, as written by a clinicians, are in a less structured and often minimal grammatical form with idiosyncratic and cryptic shorthand. Whilst there is increasing interest in the automatic extraction of the contents of clinical text, this particular type of notes cause significant difficulties for automatic extraction processes not present for well-written prose notes.

The first step to the extraction of structured information from these clinical notes is to achieve accurate identification of clinical concepts or named entities. An entity may refer to a concrete object mentioned in the notes. For example, there are 3 named entities - *CT*, *pituitary macroadenoma* and *suprasellar cisterns* in the sentence: CT revealed pituitary macroadenoma in suprasellar cisterns.

In recent years, the recognition of named entities from biomedical scientific literature has become the focus of much research, a large number of systems have been built to recognise, classify and map biomedical terms to ontologies. However, clinical terms such as findings, procedures and drugs have received less attention. Although different approaches have been proposed to identify clinical concepts and map them to terminologies (Aronson, 2001; Hazlehurst et al., 2005; Friedman et al., 2004; Jimeno et al., 2008), most of the approaches are language pattern based, which suffer from low recall. The low recall rate is mainly due to the incompleteness of medical lexicon and expressive use of alternative lexicogrammatical structures by the writers. However, only little work has used machine learning approaches, because no training data has been available, or the data are not available for clinical named entity identification.

There are semantically annotated corpora that have been developed in biomedical domain in the past few years, for example, the GENIA corpus of 2000 Medline abstracts has been annotated with biological entities (Kim et al., 2003); The PennBioIE corpus of 2300 Medline abstracts annotated with biomedical entities, part-of-speech tag and some Penn Treebank style syntactic structures (Mandel, 2006) and LLL05 challenge task corpus (Nédellec, 2005). However only a few corpora are available in the clinical domain. Many corpora are ad hoc annotations for evaluation, and

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 18–26,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

the size of the corpora are small which is not optimal for machine learning strategies. The lack of data is due to the difficulty of getting access to clinical text for research purposes and clinical information extraction is still a new area to explore. Many of the existing works focused only on clinical conditions or disease (Ogren et al., 2006; Pestian et al., 2007). The only corpus that is annotated with a variety of clinical named entities is the CLEF project (Roberts et al., 2007) .

Most of the works mentioned above are annotated on formal clinical reports and scientific literature abstracts, which generally conform to grammatical conventions of structure and readability. The CLEF data, annotated on clinical narrative reports, still uses formal clinical reports. The clinical notes presented in this work, is another genre of text, that is different from clinical reports, because they are not well-written. Notes written by clinicians and nurses are highly ungrammatical and noise prone, which creates issues in the quality of any text processing. Examples of problems arising from such texts are: firstly, variance in the representation of core medical concepts, whether unconsciously, such as typographical errors, or consciously, such as abbreviations and personal shorthand; secondly, the occurrences of different notations to signify the same concept. The clinical notes contain a great deal of formal terminology but used in an informal and unorderly manner, for example, a study of 5000 instances of Glasgow Coma Score (GCS) readings drawn from the corpus showed 321 patterns are used to denote the same concept and over 60% of them are only used once.

The clinical information extraction problem is addressed in this work by applying machine learning methods to a corpus annotated for clinical named entities. The data selection and annotation process is described in Section 3. The initial approaches to clinical concept identification using both a rule-based approach and machine learning approach are described in Section 4 and Section 5 respectively. A Conditional Random Fields based system was used to study and analyse the contribution of various feature types. The results and discussion are presented in Section 6.

2 Related Work

There is a great deal of research addressing concept identification and concept mapping issues. The Unified Medical Language System Metathesaurus (UMLS) (Lindberg et al., 1993) is the world's largest medical knowledge source and it has been the focus of much research. The simplest approaches to identifying medical concepts in text is to maintain a lexicon of all the entities of interest and to systematically search through that lexicon for all phrases of any length. This can be done efficiently by using an appropriate data structure such as a hash table. Systems that use string matching techniques include SAPHIRE (Hersh and Hickam, 1995), IndexFinder (Zou et al., 2003), NIP (Huang et al., 2005) and MaxMatcher (Zhou et al., 2006). With a large lexicon, high precision and acceptable recall were achieved by this approach in their experiments. However, using these approaches out of box for our task is not feasible, due to the high level of noise in the clinical notes, and the ad hoc variation of the terminology, will result in low precision and recall.

A more sophisticated and promising approach is to make use of shallow parsing to identify all noun phrases in a given text. The advantage of this approach is that the concepts that do not exist in the lexicon can be found. MedLEE (Friedman, 2000) is a system for information extraction in medical discharge summaries. This system uses a lexicon for recognising concept semantic classes, word qualifiers, phrases, and parses the text using its own grammar, and maps phrases to standard medical vocabularies for clinical findings and disease. The MetaMap (Aronson, 2001) program uses a three step process started by parsing freetext into simple noun phrases using the Specialist minimal commitment parser. Then the phrase variants are generated and mapping candidates are generated by looking at the UMLS source vocabulary. Then a scoring mechanism is used to evaluate the fit of each term from the source vocabulary, to reduce the potential matches (Brennan and Aronson, 2003). Unfortunately, the accurate identification of noun phrases is itself a difficult problem, especially for the clinical notes. The ICU clinical notes are highly ungrammatical and contain large number of sentence fragments and ad hoc terminology. Furthermore, highly stylised tokens of combinations of letters, digits and punctuation forming complex morphological tokens about clinical measurements in non-regular patterns add an extra load on morphological analysis, e.g. "4-6ml+/hr" means 4-6 millilitres or more secreted by the patient per hour. Parsers trained on generic text and MEDLINE abstracts have vocabularies and language models that are inappropriate for such ungrammatical texts.

Among the state-of-art systems for concept identification and named entity recognition are those that utilize machine learning or statistical techniques. Machine learners are widely used in biomedical named entity recognition and have outperformed the rule based systems (Zhou et al., 2004; Tsai et al., 2006; Yoshida and Tsujii, 2007). These systems typically involve using many features, such as word morphology or surrounding context and also extensive post-processing. A state-of-the-art biomedical named entity recognizer uses lexical features, orthographic features, semantic features and syntactic features, such as part-of-speech and shallow parsing.

Many sequential labeling machine learners have been used for experimentation, for example, Hidden Markov Model(HMM) (Rabiner, 1989), Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000) and Conditional Random Fields (CRF) (Lafferty et al., 2001). Conditional Random Fields have proven to be the best performing learner for this task. The benefit of using a machine learner is that it can utilise both the information form of the concepts themselves and the contextual information, and it is able to perform prediction without seeing the entire length of the concepts. The machine learning based systems are also good at concept disambiguation, in which a string of text may map to multiple concepts, and this is a difficult task for rule based approaches.

3 Annotation of Corpus

3.1 The Data

Data were selected form a 60 million token corpus of Royal Prince Alfred Hospital (RPAH)'s Intensive Care Service (ICS). The collection consists of clinical notes of over 12000 patients in a 6 year time span. It is composed of a variety of different types of notes, for example, patient admission notes, clinician notes, physiotherapy notes, echocardiogram reports, nursing notes, dietitian and operating theatre reports. The corpus for this study consists of 311 clinical notes drawn from patients who have stayed in ICS for more than 3 days, with most frequent causes of admission. The patients were identified in the patient records using keywords such as cardiac disease, liver disease, respiratory disease, cancer patient, patient underwent surgery etc. Notes vary in size, from 100 words to 500 words. Most of the notes consist of content such as chief complaint, patient background, current condition, history of present illness, laboratory test reports, medications, social history, impression and further plans. The variety of content in the notes ensures completely different classes of concepts are covered by the corpus. The notes were anonymised, patient-specific identifiers such as names, phone numbers, dates were replaced by a like value. All sensitive information was removed before annotation.

Category	**Example**
FINDING	*lung cancer*; *SOB*; *fever*
PROCEDURE	*chest X Ray*;*laparotomy*
SUBSTANCE	*Ceftriaxone*; *CO_2*; *platelet*
QUALIFIER	*left*; *right*;*elective*; *mild*
BODY	*renal artery*; *LAD*; *diaphragm*
BEHAVIOR	*smoker*; *heavy drinker*
ABNORMALITY	*tumor*; *lesion*; *granuloma*
ORGANISM	*HCV*; *proteus*; *B streptococcus*
OBJECT	*epidural pump; larnygoscope*
OCCUPATION	*cardiologist*; *psychiatrist*
OBSERVABLE	*GCS*; *blood pressure*

Table 1: Concept categories and examples.

3.2 Concept Category

Based on the advice of one doctor and one clinician/terminologist, eleven concept categories were defined in order to code the most frequently used clinical concepts in ICS. The eleven categories were derived from the SNOMED CT concept hierarchy. The categories and examples are listed in Table 1. Detailed explanation of these categories can be found in SNOMED CT Reference Guide[1]

3.3 Nested Concept

Nested concepts are concepts containing other concepts and are annotated in the corpus. They are of particular interest due to their compositional nature. For example, the term *left cavernous carotid aneurysm embolisation* is the outermost concept, which belongs to PROCEDURE. It contains several inner concepts: the QUALIFIER *left* and the term *cavernous carotid aneurysm* as a FINDING,

[1] SNOMED CT ® Technical Reference Guide - July 2008 International Release. http://www.ihtsdo.org/

which also contains *cavernous carotid* as BODY and *aneurysm* as ABNORMALITY.

The recognition of nested concepts is crucial for other tasks that depend on it, such as coreference resolution, relation extraction, and ontology construction, since nested structures implicitly contain relations that may help improve their correct recognition. The above outermost concept may be represented by embedded concepts and relationships as: *left cavernous carotid aneurysm embolisation* IS A *embolisation* which has LATERALITY *left*, has ASSOCIATED MORPHOLOGY *aneurysm* and has PROCEDURE SITE *cavernous carotid*.

3.4 Concept Frequency

The frequency of annotation for each concept category are detailed in Table 2. There are in total 15704 annotated concepts in the corpus, 12688 are outermost concepts and 3016 are inner concepts. The nested concepts account for 19.21% of all concepts in the corpus. The corpus has 46992 tokens, with 18907 tokens annotated as concepts, hence concept density is 40.23% of the tokens. This is higher than the density of the GENIA and MUC corpora. The 12688 annotated outermost concepts, results in an average length of 1.49 tokens per concept which is less than those of the GENIA and MUC corpora. These statistics suggest that ICU staff tend to use shorter terms but more extensively in their clinical notes which is in keeping with their principle of brevity.

The highest frequency concepts are FINDING, SUBSTANCE, PROCEDURE, QUALIFIER and BODY, which account 86.35% of data. The remaining 13.65% concepts are distributed into 6 rare categories. The inner concepts are mainly from QUALIFIER, BODY and ABNORMALITY, because most of the long and complex FINDING and PROCEDURE concepts contain BODY, ABNORMALITY and QUALIFIER, such as the example in Section 3.3.

3.5 Annotation Agreement

The corpus had been tokenised using a white-space tokeniser. Each note was annotated by two annotators: the current author and a computational linguist experienced with medical texts. Annotation guidelines were developed jointly by the annotators and the clinicians. The guidelines were refined and the annotators were trained using an iterative process. At the end of each iteration, annotation agreement was calculated and the annotations were reviewed. The guidelines were modified if necessary. This process was stopped until the agreement reached a threshold. In total 30 clinical notes were used in the development of guidelines. Inter-Annotator Agreement (IAA) is reported as the F-score by holding one annotation as the standard. F-score is commonly used in information retrieval and information extraction evaluations, which calculates the harmonic mean of recall and precision as follows:

Category	Outer	Inner	All
ABNORMALITY	0	926	926
BODY	735	1331	2066
FINDING	4741	71	4812
HEALTHPROFILE	399	0	399
OBJECT	179	23	202
OBSERVABLE	198	227	425
OCCUPATION	139	0	139
ORGANISM	36	17	53
PROCEDURE	2353	39	2392
QUALIFIER	1659	21	1680
SUBSTANCE	2249	361	2610
TOTAL	12688	3016	15704

Table 2: Frequencies for nested and outermost concept.

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

The IAA rate in the development cycle finally reached 89.83. The agreement rate between the two annotators for the whole corpus by exact matching was 88.12, including the 30 development notes. An exact match means both the boundaries and classes are exactly the same. The instances where the annotators did not agree were reviewed and relabeled by a third annotator to generate a single annotated gold standard corpus. The third annotator is used to ensure every concept is agreed on by at least two annotators.

Disagreements frequently occur at the boundaries of a term. Sometimes it is difficult to determine whether a modifier should be included in the concept: *massive medial defect* or *medial defect*, in which the latter one is a correct annotation and *massive* is a severity modifier. Mistakes in annotation also came from over annotation of a general term: *anterior approach*, which should not be annotated. Small disagreements were caused by ambiguities in the clinical notes: some medical devices (OBJECT) are often annotated as PROCEDURE, because the noun is used as a verb in the context. Another source of disagreement is due to the ambiguity in clinical knowledge: it was difficult to annotate the man-made tissues as BODY or SUBSTANCE, such as *bone graft* or *flap*.

4 Rule Based Concept Matcher

4.1 Proofreading the Corpus

Before any other processing, the first step was to resolve unknown tokens in the corpus. The unknown tokens are special orthographies or alphabetic words that do not exist in any dictionary, terminologies or gazetteers. Medical words were extracted from the UMLS lexicon and SNOMED CT (SNOMED International, 2009), and the MOBY (Ward, 1996) dictionary was used as the standard English word list. A list of abbreviations were compiled from various resources. The abbreviations in the terminology were extracted using pattern matching. Lists of abbreviations and shorthand were obtained from the hospital, and were manually compiled to resolve the meaning. Every alphabetic token was verified against the dictionary list, and classified into *Ordinary English Words*, *Medical Words*, *Abbreviations*, and *Unknown Words*.

An analysis of the corpus showed 31.8% of the total tokens are non-dictionary words, which contains 5% unknown alphabetic words. Most of these unknown alphabetic words are obvious spelling mistakes. The spelling errors were corrected using a spelling corrector trained on the 60 million token corpus, Abbreviations and shorthand were expanded, for example *defib* expands to *defibrillator*. Table 3 shows some unknown tokens and their resolutions. The proofreading require considerable amount of human effort to build the dictionaries.

4.2 Lexicon look-up Token Matcher

The lexicon look-up performed exact matching between the concepts in the SNOMED CT terminology and the concepts in the notes. A hash table data structure was implemented to index lexical items in the terminology. This is an extension to the algorithm described in (Patrick et al., 2006). A token matching matrix run through the sentence to find all candidate matches in the sentence to the lexicon, including exact longest matches, partial matches, and overlapping between matches. Then a Viterbi algorithm was used to find the best sequence of non-overlapping concepts in a sentence that maximise the total similarity score. This method matches the term as it appears in the terminology so is not robust against term variations that have not been seen in the terminology, which results in an extremely low recall. In addition, the precision may be affected by ambiguous terms or nested terms.

unknown word	examples	resolution
CORRECT WORD	bibasally	bibasally
MISSING SPACE	oliclinomel	Oli Clinomel
SPELLING ERROR	dolaseteron	dolasetron
ACRONYM	BP	blood pressure
ABBREVIATION	N+V	Nausea and vomiting
SHORTHAND	h'serous	haemoserous
MEASUREMENT	e4v1m6	GCS measurement
SLASHWORDS	abg/ck/tropt	ABG CK Tropt
READINGS	7mg/hr	

Table 3: Unknown tokens and their resolutions.

The exact lexicon look-up is likely to fail on matching long and complex terms, as clinicians do not necessarily write the modifier of a concept in a strict order, and some descriptors are omitted. for example *white blood cell count normal* can be written as *normal white cell count*. In order to increase recall, partial matching is implemented. The partial matching tries to match the best sequence, but penalise non-matching gaps between two terms. The above example will be found using partial matching.

5 CRF based Clinical Named Entity Recogniser

5.1 Conditional Random Fields

The concept identification task has been formulated as a named entity recognition task, which can be thought of as a sequential labeling problem: each word is a token in a sequence to be assigned a label, for example, B-FINDING, I-FINDING, B-PROCEDURE, I-PROCEDURE, B-SUBSTANCE, I-SUBSTANCE and so on. Conditional Random Fields (CRF) are undirected statistical graphical models, which is a linear chain of Maximum Entropy Models that evaluate the conditional probability on a sequence of states give a sequence of observations. Such models are suitable for sequence analysis. CRFs has been applied to the task of recognition of biomedical named entities and have outperformed other machine learning models. CRF++[2] is used for conditional random fields learning.

5.2 Features for the Learner

This section describes the various features used in the CRF model. Annotated concepts were converted into BIO notation, and feature vectors were generated for each token.

Orthographic Features: Word formation was genaralised into orthographic classes. The present model uses 7 orthographic features to indicate whether the words are captialised or upper case, whether they are alphanumeric or contains any slashes, as many findings consist of captialised words; substances are followed by dosage, which can be captured by the orthography. Word prefixes and suffixes of character length 4 were also used as features, because some procedures, substances and findings have special affixes, which are very distinguishable from ordinary words.

Lexical Features: Every token in the training data was used as a feature. Alphabetic words in the training data were converted to lowercase, spelling errors detected in proofreading stage were replaced by the correct resolution. Shorthand and abbreviations were expanded into bag of words (*bow*) features. The left and right lexical bigrams were also used as a feature, however it only yielded a slight improvement in performance. To utilise the context information, neighboring words in the window $[-2, +2]$ are also added as features. Context window size of 2 is chosen because it yields the best performance. The target and previous labels are also used as features, and had been shown to be very effective.

Semantic Features: The output from the lexical-lookup system was used as features in the CRF model. The identified concepts were added to the feature set as semantic features, because the terminology can provide semantic knowledge to the learner such as the category information of the term. Moreover, many partially matched concepts from lexicon-lookup were counted as incorrectly matching, however they are single term head nouns which are effective features in NER.

Syntactic features were not used in this experiment as the texts have only a little grammatical structure. Most of the texts appeared in fragmentary sentences or single word or phrase bullet point format, which is difficult for generic parsers to work with correctly.

Experiment	**P**	**R**	**F-score**
no pruning	58.76	26.63	36.35
exact matching	69.48	37.70	48.88
+*proofreading*	74.81	52.42	61.65
+*partial matching*	69.39	59.60	64.12

Table 4: Lexical lookup Performance.

6 Evaluation

This section presents experiment results for both the rule-based system and machine learning based system. Only the 12688 outermost concepts are used in the experiments, because nested terms result in multi-label for a single token. Since there is no outermost concepts in ABNORMALITY, the classification was done on the remaining 10 categories. The performances were evaluated in terms of recall, precision and F-score.

6.1 Token Matcher Performance

The lexical lookup performance is evaluated on the whole corpus. The first system uses only exact matching without any pre-processing of the lexicon. The second experiment uses a pruned terminology with ambiguous categories and unnecessary categories removed, but without proofreading of the corpus. The concept will be removed if it belongs to a category that is not used in the annotation. The third experiment used the proofreaded corpus with all abbreviations annotated. The fourth experiment was conducted on the proofread corpus allowing both exact matching and partial matching. The results are outlined in Table 4.

The lexicon lookup without pruning the terminologies achieved low precision and extremely low recall. This is mainly due to the ambiguous terms in the lexicon. By removing unrelated terms and categories in the lexicon, both precision and recall improved dramatically. Proofreading, correcting a large number of unknown tokens such as spelling errors or irregular conventions further increased both precision and recall. The 14.72 gain in recall mainly came from resolution and expansion of shorthand, abbreviations, and acronyms in the notes. This also suggest that this kind of clinical notes are very noisy, and require a consider-

[2] http://crfpp.sourceforge.net/

able amount of effort in pre-processing. Allowing partial matching increased recall by 7.18, but decreased precision by 5.52, and gave the overall increase of 2.47 F-score. Partial matching discovered a larger number of matching candidates using a looser matching criteria, therefore decreased in precision with compensation of an increase in recall.

The highest precision achieved by exact matching is 74.81, confirming that the lexical lookup method is an effective means of identifying clinical concepts. However, it requires extensive effort on pre-processing both corpus and the terminology and is not easily adapted to other corpora. The lexical matching fails to identify long terms and has difficult in term disambiguation. The low recall is caused by incompleteness of the terminology. However, the benefit of using lexicon lookup is that the system is able to assign a concept identifier to the identified concept if available.

6.2 CRF Feature Performance

The CRF system has been evaluated using 10-fold cross validation on the data set. The evaluation was performed using the CoNLL shared task evaluation script [3].

The CRF classifier experiment results are shown in Table 5. A baseline system was built using only *bag-of-word* features from the training corpus. A context-window size of 2 and tag prediction of previous token were used in all experiments. Without using any contextual features the performance was 48.04% F-score. The baseline performance of 71.16% F-score outperformed the lexical-look up performance. Clearly the contextual information surrounding the concepts gives a strong contribution in identification of concepts, while lexical-lookup hardly uses any contextual information.

The full system is built using all features described in Section 5.2, and achieved the best result of 81.48% F-score. This is a significant improvement of 10.32% F-score over the baseline system. Further experimental analysis of the contribution of feature types was conducted by removing each feature type from the full system. −*bow* means bag-of-word features are removed from the full system. The results show only *bow* and *lexical-lookup* features make significant contribution to the system, which are 5.49% and 4.40% separately. *Bigram*, *orthographic*, *affixes* and *abbreviation* features each makes around ∼ 1% contribution to the F-score, which is individually insignificant, however the combination of them makes a significant contribution, which is 4.83% F-score.

Experiment	P	R	F-score
baseline	76.86	66.26	71.16
+*lexical-lookup*	82.61	74.88	78.55
full	**84.22**	**78.90**	**81.48**
−*bow*	81.26	73.32	77.08
−*bigram*	83.17	78.74	80.89
−*abbreviation*	83.20	77.26	80.12
−*orthographic*	83.67	78.24	80.87
−*affixes*	83.16	77.01	79.97
−*lexical-lookup*	79.06	73.15	75.99

Table 5: Experiment on Feature Contribution for the ICU corpus.

The most effective feature in the system is the output from the lexical lookup system. Another experiment using only *bow* and *lexical-lookup* features showed a boost of 7.39% F-score. This is proof of the hypothesis that using terminology information in the machine learner would increase recall. In this corpus, about one third of the concepts has a frequency of only 1, from which the learner as unable to learn anything from the training data. The gain in performance is due to the ingestion of semantic domain knowledge which is provided by the terminology. This knowledge is useful for determining the correct boundary of a concept as well as the classification of the concept.

6.3 Detailed CRF Performance

The detailed results of the CRF system are shown in Table 6. Precision, Recall and F-score for each class are reported. There is a consistent gap between Recall and Precision across all categories. The best performing classes are among the most frequent categories. This is an indication that sufficient training data is a crucial factor in achieving high performance. SUBSTANCE, PROCEDURE and FINDING are the best three categories due to their high frequency in the corpus. However, QUALIFIER achieved a lower F-score because qualifiers usually appear at the boundaries of two concepts, which is a source of error in boundary recognition.

Low frequency categories generally achieved high precision and low recall. The recall decreases as the number of training instances decreases, be-

[3] http://www.cnts.ua.ac.be/conll2002/ner/bin/

Class	P	R	F-score
BODY	72.00	64.29	67.92
FINDING	83.17	78.74	80.89
BEHAVIOR	83.87	72.22	77.61
OBJECT	75.00	27.27	40.00
OBSERVABLE	89.47	56.67	69.39
ORGANISM	0.00	0.00	0.00
PROCEDURE	87.63	81.09	84.24
QUALIFIER	75.80	75.32	75.56
OCCUPATION	87.50	41.18	56.00
SUBSTANCE	91.90	88.53	90.19

Table 6: Detailed Performance of the CRF system.

cause there is not enough information in the training data to learn the class profiles. It is a challenge to boost the recall of rare categories due to the variability of the terms in the notes. It is not likely that the term would match to the terminology, and hence there would be no utilisation of the semantic information.

Another factor that causes recognition errors is the nested concepts. BODY achieved the least precision because of the high frequency of nested concepts in its category. The nested construction also causes boundary detection problems, for example *C5/6 cervical discectomy* PROCEDURE is annotated as *C5/6* BODY and *cervical discectomy* PROCEDURE.

The results presented here are higher than those reported in biomedical NER system. Although it is difficult to compare with other work because of the different data set, but this task might be easier due to the shorter length of the concepts and fewer long concepts (avg. 1.49 in this corpus vs. avg. 1.70 token per concept in GENIA). Local features would be able to capture most of the useful information while not introducing ambiguity.

7 Future Work and Conclusion

This paper presents a study of identification of concepts in progressive clinical notes, which is another genre of text that hasn't been studied to date. This is the first step towards information extraction of free text clinical notes and knowledge representation of patient cases. Now that the corpus has been annotated with coarse grained concept categories in a reference terminology, a possible improvement of the annotation is to reevaluate the concept categories and create fine grained categories by dividing top categories into smaller classes along the terminology's hierarchy. For example, the FINDING class can be further divided into SYMPTOM/SIGN, DISORDER and EVALUATION RESULTS. The aim would be to achieve better consistency, less ambiguity and greater coverage of the concepts in the corpus.

The nested concepts model the relations between atomic concepts within the outermost concepts. These structures represent important relationships within this type of clinical concept. The next piece of work could be the study of these relationships. They can be extended to represent relationships between clinical concepts and allow for representing new concepts using structured information. The annotation of relations is under development. The future work will move from concept identification to relation identification and automatic ontology extension.

Preliminary experiments in clinical named entity recognition using both rule-based and machine learning approaches were performed on this corpus. These experiments have achieved promising results and show that rule based lexicon lookup, with considerable effort on pre-processing and lexical verification, can significantly improve performance over a simple exact matching process. However, a machine learning system can achieve good results by simply adapting features from biomedical NER systems, and produced a meaningful baseline for future research. A direction to improve the recogniser is to add more syntactic features and semantic features by using dependency parsers and exploiting the unlabeled 60 million token corpus.

In conclusion, this paper described a new annotated corpus in the clinical domain and presented initial approaches to clinical named entity recognition. It has demonstrated that practical acceptable named entity recognizer can be trained on the corpus with an F-score of 81.48%. The challenge in this task is to increase recall and identify rare entity classes as well as resolve ambiguities introduced by nested concepts. The results should be improved by using extensive knowledge resource or by increasing the size and improving the quality of the corpus.

Acknowledgments

The author wish to thank the staff of the Royal Prince Alfred Hospital, Sydney : Dr. Stephen Crawshaw, Dr. Robert Herks and Dr Angela Ryan

for their support in this project.

References

R. Aronson. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *In Proceeding of the AMIA Symposium*,17–21.

F. Brennan and A. Aronson 2003. Towards linking patients and clinical information: detecting UMLS concepts in e-mail. *Journal of Biomedical Informatics*,36(4/5),334–341.

A. Côté and American Veterinary Medical Association and College of American Pathologists. 2009. Snomed International. *College of American Pathologists.*

C. Friedman 2000. A broad coverage natural language processing system. *In Proceedings of the AMIA Symposium*,270–274.

C. Friedman, L. Shagina, Y. Lussier, and G. Hripcsak. 2004. Automated Encoding of Clinical Documents Based on Natural Language Processing. *Journal of the American Medical Informatics Association*,11(5),392–402.

B. Hazlehurst, R. Frost, F. Sittig, and J. Stevens. 2005. MediClass: A System for Detecting and Classifying Encounter-based Clinical Events in Any Electronic Medical Record. *Journal of the American Medical Informatics Association*,12(5),517–529.

R. Hersh, and D. Hickam. 1995. Information retrieval in medicine: The SAPHIRE experience. *Journal of the American Society for Information Science*,46(10),743–747.

Y. Huang, J. Lowe, D. Klein, and J. Cucina. 2005. Improved Identification of Noun Phrases in Clinical Radiology Reports Using a High-Performance Statistical Natural Language Parser Augmented with the UMLS Specialist Lexicon. *Journal of the American Medical Informatics Association*,12(3),275–285.

A. Jimeno, et al. 2008. Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics*,9(3).

D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. 2003. GENIA corpus - a semantically annotated corpus for bio-textmining. *Journal of Bioinformatics*, 19(1),180–182.

J. Lafferty et al. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data *Machine learning-international workshop then conference*, 282–289.

A. Lindberg et al. 1993. The Unified Medical Language System. *Methods Inf Med.*

M. Mandel 2006. Integrated Annotation of Biomedical Text: Creating the PennBioIE corpus. *Text Mining Ontologies and Natural Language Processing in Biomedicine*, Manchester, UK.

A. McCallum, et al. 2000. Maximum entropy Markov models for information extraction and segmentation *Proc. 17th International Conf. on Machine Learning*, 591–598.

C. N'edellec. 2005. Learning Language in Logic - Genic Interaction Extraction Challenge. *Proceedings of the ICML05 Workshop on Learning Language in Logic*, Bonn, 31–37.

V. Ogren, G. Savova, D. Buntrock, and G. Chute. 2006. Building and Evaluating Annotated Corpora for Medical NLP Systems. *AMIA Annu Symp Proceeding.*.

J. Patrick, Y. Wang, and P. Budd. 2006. Automatic Mapping Clinical Notes to Medical Terminologies *In Proceedings of Australasian Language Technology Workshop.*

P. Pestian, C. Brew, P. Matykiewicz, J. Hovermale, N. Johnson, K. Cohen, and W. Duch. 2007. A Shared Task Involving Multi-label Classification of Clinical Free Text *In Proceedings of BioNLP workshop.*

R. Rabiner 1989. A tutorial on hidden Markov models and selected applications inspeech recognition *Proceedings of the IEEE*,77(2), 257–286.

A. Roberts, R. Gaizauskas, M. Hepple, G. Demetriou, Y. Guo, A. Setzer, I. and Roberts. 2007. The CLEF Corpus: Semantic Annotation of Clinical Text. *AMIA Annu Symp Proceeding.*, Oct 11:625–629.

R. Tsai, L. Sung, J. Dai, C. Hung, Y. Sung, and L. Hsu. 2006. NERBio: using selected word conjunctions, term normalization, and global patterns to improve biomedical named entity *BMC Bioinformatics.*

G. Ward 1996. Moby thesaurus. http://etext.icewire.com/moby/.

K. Yoshida, and J. Tsujii. 2007. Reranking for Biomedical Named-Entity Recognition *Proceedings of BioNLP 2007: Biological, translational, and clinical language processing*, 209–216.

G. Zhou, J. Zhang, J. Su, D. Shen, and L. Tan. 2004. Recognizing Names in Biomedical Texts: a Machine Learning Approach *BioInformatics*, 20(7) 1178–1190.

X. Zhou, et al. 2006. MaxMatcher: Biological Concept Extraction Using Approximate Dictionary Lookup. *Proc PRICAI*,1145–1149.

Q. Zou. 2003. IndexFinder: A Method of Extracting Key Concepts from Clinical Texts for Indexing. *Proc AMIA Symp*,763–767.

Paraphrase Recognition Using Machine Learning to Combine Similarity Measures

Prodromos Malakasiotis
Department of Informatics
Athens University of Economics and Business
Patission 76, GR-104 34 Athens, Greece

Abstract

This paper presents three methods that can be used to recognize paraphrases. They all employ string similarity measures applied to shallow abstractions of the input sentences, and a Maximum Entropy classifier to learn how to combine the resulting features. Two of the methods also exploit WordNet to detect synonyms and one of them also exploits a dependency parser. We experiment on two datasets, the MSR paraphrasing corpus and a dataset that we automatically created from the MTC corpus. Our system achieves state of the art or better results.

1 Introduction

Recognizing or generating semantically equivalent phrases is of significant importance in many natural language applications. In question answering, for example, a question may be phrased differently than in a document collection (e.g., "Who *is the author of* War and Peace?" vs. "Leo Tolstoy *is the writer of* War and Peace."), and taking such variations into account can improve system performance significantly (Harabagiu et al., 2003; Harabagiu and Hickl, 2006). A paraphrase generator, meaning a module that produces new phrases or patterns that are semantically equivalent (or almost equivalant) to a given input phrase or pattern (e.g., "X is the writer of Y" $\Leftrightarrow$ "X wrote Y" $\Leftrightarrow$ "Y was written by X" $\Leftrightarrow$ "X is the author of Y", or "X produces Y" $\Leftrightarrow$ "X manufactures Y" $\Leftrightarrow$ "X is the manufacturer of Y") can be used to produce alternative phrasings of the question, before matching it against a document collection.

Unlike paraphrase generators, paraphrase recognizers decide whether or not two *given* phrases (or patterns) are paraphrases, possibly by generalizing over many different training pairs of phrases. Paraphrase recognizers can be embedded in paraphrase generators to filter out erroneous generated paraphrases; but they are also useful on their own. In question answering, for example, they can be used to check if a pattern extracted from the question (possibly by replacing named entities by their semantic categories and turning the question into a statement) matches any patterns extracted from candidate answers. As a further example, in text summarization, especially multi-document summarization, a paraphrase recognizer can be used to check if a sentence is a paraphrase of any other sentence already present in a partially constructed summary.

Note that, although "paraphrasing" and "textual entailment" are sometimes used as synonyms, we use the former to refer to methods that generate or recognize semantically equivalent (or almost equivalent) phrases or patterns, whereas in textual entailment (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007) the expressions or patterns are not necessarily semantically equivalent; it suffices if one entails the other, even if the reverse direction does not hold. For example, "Y was written by X" textually entails "Y is the work of X", but the reverse direction does not necessarily hold (e.g., if Y is a statue); hence, the two sentences are not paraphrases.

In this paper, we focus on paraphrase recognition. We propose three methods that employ string similarity measures, which are applied to several abstractions of a pair of input phrases (e.g., the phrases themselves, their stems, POS tags). The scores returned by the similarity measures are used as features in a Maximum Entropy (ME) classifier (Jaynes, 1957; Good, 1963), which learns to separate true paraphrase pairs from false ones. Two of our methods also exploit WordNet to detect synonyms, and one of them uses additional features to measure similarities of grammatical relations

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 27–35,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

obtained by a dependency parser.[1] Our experiments were conducted on two datasets: the publicly available Microsoft Research Paraphrasing corpus (Dolan et al., 2004) and a dataset that we constructed from the MTC corpus.[2] The experimental results show that our methods perform very well. Even the simplest one manages to achieve state of the art results, even though it uses fewer linguistic resources than other reported systems. The other two, more elaborate methods perform even better.

Section 2 presents the three methods, and section 3 our experiments. Section 4 covers related work. Section 5 concludes and proposes further work.

2 The three methods

The main idea underlying our methods is that by capturing similarities at various shallow abstractions of the input (e.g., the original sentences, the stems of their words, their POS tags), we can recognize paraphrases and textual entailment reasonably well, provided that we learn to assign appropriate weights to the resulting features. Further improvements are possible by recognizing synonyms and by employing similarity measures that operate on the output of dependency grammar parsers.

2.1 Method 1 (INIT)

During training, the first method, called INIT, is given a set $\{\langle S_{1,1}, S_{1,2}, y_1\rangle, \ldots, \langle S_{n,1}, S_{n,2}, y_n\rangle\}$, where $S_{i,1}$ and $S_{i,2}$ are sentences (more generally, phrases), $y_i = 1$ (positive class) if the two sentences are paraphrases, and $y_i = -1$ (negative class) otherwise. Each pair of sentences $\langle S_{i,1}, S_{i,2}\rangle$ is converted to a feature vector $\vec{v_i}$, whose values are scores returned by similarity measures that indicate how similar $S_{i,1}$ and $S_{i,2}$ are at various levels of abstraction. The vectors and the corresponding categories $\{\langle \vec{v_1}, y_i\rangle, \ldots, \langle \vec{v_n}, y_n\rangle\}$ are given as input to the ME classifier, which learns how to classify new vectors $\vec{v}$, corresponding to unseen pairs of sentences $\langle S_1, S_2\rangle$.

We use nine string similarity measures: Levenshtein distance (edit distance), Jaro-Winkler distance, Manhattan distance, Euclidean distance, cosine similarity, n-gram distance (with $n = 3$), matching coefficient, Dice coefficient, and Jaccard coefficient. To save space, we do not repeat the definitions of the similarity measures here, since they are readily available in the literature and they are also summarized in our previous work (Malakasiotis and Androutsopoulos, 2007).

For each pair of input strings $\langle S_1, S_2\rangle$, we form ten new pairs of strings $\langle s_1^1, s_2^1\rangle, \ldots, \langle s_1^{10}, s_2^{10}\rangle$ corresponding to ten different levels of abstraction of S_1 and S_2, and we apply the nine similarity measures to the ten new pairs, resulting in a total of 90 measurements. These measurements are then included as features in the vector $\vec{v}$ that corresponds to $\langle S_1, S_2\rangle$. The $\langle s_1^i, s_2^i\rangle$ pairs are:

$\langle s_1^1, s_2^1\rangle$: two strings consisting of the *original tokens* of S_1 and S_2, respectively, with the original order of the tokens maintained;[3]

$\langle s_1^2, s_2^2\rangle$: as in the previous case, but now the tokens are replaced by their *stems*;

$\langle s_1^3, s_2^3\rangle$: as in the previous case, but now the tokens are replaced by their *part-of-speech* (POS) tags;

$\langle s_1^4, s_2^4\rangle$: as in the previous case, but now the tokens are replaced by their *soundex codes*;[4]

$\langle s_1^5, s_2^5\rangle$: two strings consisting of only the *nouns* of S_1 and S_2, as identified by a POS-tagger, with the original order of the nouns maintained;

$\langle s_1^6, s_2^6\rangle$: as in the previous case, but now with *nouns replaced by their stems*;

$\langle s_1^7, s_2^7\rangle$: as in the previous case, but now with *nouns replaced by their soundex codes*;

$\langle s_1^8, s_2^8\rangle$: two strings consisting of only the *verbs* of S_1 and S_2, as identified by a POS-tagger, with the original order of the verbs maintained;

$\langle s_1^9, s_2^9\rangle$: as in the previous case, but now with *verbs replaced by their stems*;

$\langle s_1^{10}, s_2^{10}\rangle$: as in the previous case, but now with *verbs replaced by their soundex codes*.

Note that the similarities are measured in terms of tokens, not characters. For instance, the edit distance of S_1 and S_2 is the minimum number of operations needed to transform S_1 to S_2, where an operation is an insertion, deletion or substitution of a single token. Moreover, we use high-level

[1] We use Stanford University's ME classifier and parser; see http://nlp.stanford.edu/.

[2] The corpus is available by the LDC, Catalogue Number LDC2002T01, ISBN 1-58563-217-1.

[3] We use Stanford University's tokenizer and POS-tagger, and Porter's stemmer.

[4] Soundex is an algorithm intended to map English names to alphanumeric codes, so that names with the same pronunciations receive the same codes, despite spelling differences; see http://en.wikipedia.org/wiki/Soundex.

POS tags only, i.e., we do not consider the number of nouns, the voice of verbs etc.; this increases the similarity of positive $\langle s_1^3, s_2^3 \rangle$ pairs.

A common problem is that the string similarity measures may be misled by differences in the lengths of S_1 and S_2. This is illustrated in the following examples, where the underlined part of S_1 is much more similar to S_2 than the entire S_1.

S_1: While Bolton apparently fell and was immobilized, Selenski used the mattress to scale a 10-foot, razor-wire fence, Fischi said.

S_2: After the other inmate fell, Selenski used the mattress to scale a 10-foot, razor-wire fence, Fischi said.

To address this problem, when we consider a pair of strings $\langle s_1, s_2 \rangle$, if s_1 is longer than s_2, we obtain all of the substrings s_1' of s_1 that have the same length as s_2. Then, for each s_1', we compute the nine values $f_j(s_1', s_2)$, where f_j $(1 \leq j \leq 9)$ are the string similarity measures. Finally, we locate the s_1' with the best average similarity (over all similarity measures) to s_2, namely $s_1'^*$:

$$s_1'^* = \arg\max_{s_1'} \sum_{j=1}^{10} f_j(s_1', s_2)$$

and we keep the nine $f_j(s_1'^*, s_2)$ values and their average as ten additional measurements. Similarly, if s_2 is longer than s_1, we keep the nine $f_j(s_1, s_2'^*)$ values and their average. This process is applied to pairs $\langle s_1^1, s_2^1 \rangle, \ldots, \langle s_1^4, s_2^4 \rangle$, where large length differences are more likely to appear, adding 40 more measurements (features) to the vector $\vec{v}$ of each $\langle S_1, S_2 \rangle$ pair of input strings.

The measurements discussed above provide 130 numeric features.[5] To those, we add two Boolean features indicating the existence or absence of negation in S_1 or S_2, respectively; negation is detected by looking for words like “not”, “won’t” etc. Finally, we add a length ratio feature, defined as $\frac{\min(L_{S_1}, L_{S_2})}{\max(L_{S_1}, L_{S_2})}$, where L_{S_1} and L_{S_2} are the lengths, in tokens, of S_1 and S_2. Hence, there is a total of 133 available features in INIT.

2.2 Method 2 (INIT+WN)

Paraphrasing may involve using synonyms which cannot be detected by the features we have considered so far. In the following pair of sentences, for example, “dispatched” is used as a synonym of “sent”; treating the two verbs as the same token during the calculation of the string similarity measures would yield a higher similarity. The second method, called INIT+WN, treats words from S_1 and S_2 that are synonyms as identical; otherwise the method is the same as INIT.

S_1: Fewer than a dozen FBI agents were dispatched to secure and analyze evidence.

S_2: Fewer than a dozen FBI agents will be sent to Iraq to secure and analyze evidence of the bombing.

2.3 Method 3 (INIT+WN+DEP)

The features of the previous two methods operate at the lexical level. The third method, called INIT+WN+DEP, adds features that operate on the grammatical relations (dependencies) a dependency grammar parser returns for S_1 and S_2. We use three measures to calculate similarity at the level of grammatical relations, namely S_1 dependency recall (R_1), S_2 dependency recall (R_2) and their F-measure (F_{R_1,R_2}), defined below:

$$R_1 = \frac{|common\ dependencies|}{|S_1\ dependencies|}$$

$$R_2 = \frac{|common\ dependencies|}{|S_2\ dependencies|}$$

$$F_{R_1,R_2} = \frac{2 \cdot R_1 \cdot R_2}{R_1 + R_2}$$

The following two examples illustrate the usefulness of dependency similarity measures in detecting paraphrases. In the first example S_1 and S_2 are not paraphrases and the scores are low, while in the second example where S_1 and S_2 have almost identical meanings, the scores are much higher. Figures 1 and 2 lists the grammatical relations (dependencies) of the two sentences with the common ones shown in bold.

Example 1:

S_1: Gyorgy Heizler, head of the local disaster unit, said the coach was carrying 38 passengers.

S_2: The head of the local disaster unit, Gyorgy Heizler, said the coach driver had failed to heed red stop lights.

$R_1 = 0.43$, $R_2 = 0.32$, $F_{R_1,R_2} = 0.36$

Example 2:

S_1: Amrozi accused his brother, whom he called “the witness”, of deliberately distorting his evidence.

S_2: Referring to him as only “the witness”, Amrozi accused his brother of deliberately distorting his evidence.

$R_1 = 0.69$, $R_2 = 0.6$, $F_{R_1,R_2} = 0.64$

[5] All feature values are normalized in $[-1, 1]$. We use our own implementation of the string similarity measures.

Grammatical relations of S_1

mod(Heizler-2, Gyorgy-1)
arg(said-11, Heizler-2)
mod(Heizler-2, head-4)
mod(head-4, of-5)
mod(unit-9, the-6)
mod(unit-9, local-7)
mod(unit-9, disaster-8)
arg(of-5, unit-9)
mod(coach-13, the-12)
arg(carrying-15, coach-13)
aux(carrying-15, was-14)
arg(said-11, carrying-15)
mod(passengers-17, 38-16)
arg(carrying-15, passengers-17)

Grammatical relations of S_2

mod(head-2, The-1)
arg(said-12, head-2)
mod(head-2, of-3)
mod(unit-7, the-4)
mod(unit-7, local-5)
mod(unit-7, disaster-6)
arg(of-3, unit-7)
mod(Heizler-10, Gyorgy-9)
mod(unit-7, Heizler-10)
mod(driver-15, the-13)
mod(driver-15, coach-14)
arg(failed-17, driver-15)
aux(failed-17, had-16)
arg(said-12, failed-17)
aux(heed-19, to-18)
arg(failed-17, heed-19)
mod(lights-22, red-20)
mod(lights-22, stop-21)
arg(heed-19, lights-22)

Figure 1: Grammatical relations of example 1.

Grammatical relations of S_1

arg(accused-2, Amrozi-1)
mod(brother-4, his-3)
arg(accused-2, brother-4)
arg(called-8, whom-6)
arg(called-8, he-7)
mod(brother-4, called-8)
mod(witness-11, the-10)
dep(called-8, witness-11)
mod(brother-4, of-14)
mod(distorting-16, deliberately-15)
arg(of-14, distorting-16)
mod(evidence-18, his-17)
arg(distorting-16, evidence-18)

Grammatical relations of S_2

dep(accused-12, Referring-1)
mod(Referring-1, to-2)
arg(to-2, him-3)
cc(him-3, as-4)
dep(as-4, only-5)
mod(witness-8, the-7)
conj(him-3, witness-8)
arg(accused-12, Amrozi-11)
mod(brother-14, his-13)
arg(accused-12, brother-14)
mod(brother-14, of-15)
mod(distorting-17, deliberately-16)
arg(of-15, distorting-17)
mod(evidence-19, his-18)
arg(distorting-17, evidence-19)

Figure 2: Grammatical relations of example 2.

As with POS-tags, we use only the highest level of the tags of the grammatical relations, which increases the similarity of positive pairs of S_1 and S_2. For the same reason, we ignore the directionality of the dependency arcs which we have found to improve the results. INIT+WN+DEP employs a total of 136 features.

2.4 Feature selection

Larger feature sets do not necessarily lead to improved classification performance. Despite seeming useful, some features may in fact be too noisy or irrelevant, increasing the risk of overfitting the training data. Some features may also be redundant, given other features; thus, feature selection methods that consider the value of each feature on its own (e.g., information gain) may lead to suboptimal feature sets.

Finding the best subset of a set of available features is a search space problem for which several methods have been proposed (Guyon et al., 2006). We have experimented with a wrapper approach, whereby each feature subset is evaluated according to the predictive power of a classifier (treated as a black box) that uses the subset; in our experiments, the predictive power was measured as F-measure (defined below, not to be confused with F_{R_1,R_2}). More precisely, during feature selection, for each feature subset we performed 10-fold cross validation on the training data to evaluate its predictive power. After feature selection, the classifier was trained on all the training data, and it was evaluated on separate test data.

With large feature sets, an exhaustive search over all subsets is intractable. Instead, we experimented with forward hill-climbing and beam search (Guyon et al., 2006). Forward hill-climbing starts with an empty feature set, to which it adds features, one at a time, by preferring to add at each step the feature that leads to the highest predictive power. Forward beam search is similar, except that the search frontier contains the k best examined states (feature subsets) at each time; we used $k = 10$. For $k = 1$, beam search reduces to hill-climbing.

3 Experiments

We now present our experiments, starting from a description of the datasets used.

3.1 Datasets

We mainly used the Microsoft Research (MSR) Paraphrasing Corpus (Dolan et al., 2004), which consists of 5,801 pairs of sentences. Each pair is manually annotated by two human judges as a true or false paraphrase; a third judge resolved disagreements. The data are split into 4,076 training pairs and 1,725 testing pairs.

We have experimented with a dataset we created from the MTC corpus. MTC is a corpus containing news articles in Mandarin Chinese; for each article 11 English translations (by different translators) are also provided. We considered the translations of the same Chinese sentence as paraphrases. We obtained all the possible paraphrase pairs and we added an equal number of randomly selected non paraphrase pairs, which contained sentences that were not translations of the same sentence. In this way, we constructed a dataset containing 82,260 pairs of sentences. The dataset was then split in training (70%) and test (30%) parts, with an equal number of positive and negative pairs in each part.

3.2 Evaluation measures and baseline

We used four evaluation measures, namely accuracy (correctly classified pairs over all pairs), precision (P, pairs correctly classified in the positive class over all pairs classified in the positive class), recall (R, pairs correctly classified in the positive class over all true positive pairs), and F-measure (with equal weight on precision and recall, defined as $\frac{2 \cdot P \cdot R}{P+R}$). These measures are not to be confused with the R_1, R_2, and F_{R_1,R_2} of section 2.3 which are used as features.

A reasonable baseline method (BASE) is to use just the edit distance similarity measure and a threshold in order to decide whether two phrases are paraphrases or not. The threshold is chosen using a grid search utility and 10-fold cross validation on the training data. More precisely, in a first step we search the range [-1, 1] with a step of 0.1.[6] In each step, we perform 10-fold cross validation and the value that achieves the best F-measure is our initial threshold, th, for the second step. In the second step, we perform the same procedure in the range [th - 0.1, th + 0.1] and with a step of 0.001.

[6] Recall that we normalize similarity in [-1, 1].

3.3 Experimental results

With both datasets, we experimented with a Maximum Entropy (ME) classifier. However, preliminary results (see table 1) showed that our MTC dataset is very easy. BASE achieves approximately 95% in accuracy and F-measure, and an approximate performance of 99.5% in all measures (accuracy, precision, recall, F-measure) is achieved by using ME and only some of the features of INIT (we use 36 features corresponding to pairs $\langle s_1^1, s_2^1 \rangle$, $\langle s_1^2, s_2^2 \rangle$, $\langle s_1^3, s_2^3 \rangle$, $\langle s_1^4, s_2^4 \rangle$ plus the two negation features). Therefore, we did not experiment with the MTC dataset any further.

Table 2 (upper part) lists the results of our experiments on the MSR corpus. We optionally performed feature selection with both forward hill-climbing (FHC) and forward beam search (FBS). All of our methods clearly perform better than BASE. As one might expect, there is a lot of redundancy in the complete feature set. Hence, the two feature selection methods (FHC and FBS) lead to competitive results with much fewer features (7 and 10, respectively, instead of 136). However, feature selection deteriorates performance, especially accuracy, i.e., the full feature set is better, despite its redundancy. Table 2 also includes all other reported results for the MSR corpus that we are aware of; we are not aware of the exact number of features used by the other researchers.

It is noteworthy that INIT achieves state of the art performance, even though the other approaches use many more linguistic resources. For example, Wan et al.'s approach (Wan et al., 2006), which achieved the best previously reported results, is similar to ours, in that it also trains a classifier with similarity measures; but some of Wan et al.'s measures require a dependency grammar parser, unlike INIT. More precisely, for each pair of sentences, Wan et al. construct a feature vector with values that measure lexical and dependency similarities. The measures are: word overlap, length difference (in words), BLEU (Papineni et al., 2002), dependency relation overlap (i.e., R_1 and R_2 but not F_{R_1,R_2}), and dependency tree edit distance. The measures are also applied on sequences containing the lemmatized words of the original sentences, similarly to one of our levels of abstraction. Interestingly, INIT achieves the same (and slightly better) accuracy as Wan et al.'s system, without employing any parsing. Our more enhanced methods, INIT+WN and INIT+WN+DEP, achieve even better results.

Zhang and Patrick (2005) use a dependency grammar parser to convert passive voice phrases to active voice ones. They also use a preprocessing stage to generalize the pairs of sentences. The preprocessing replaces dates, times, percentages, etc. with generic tags, something that we have also done in the MSR corpus, but it also replaces words and phrases indicating future actions (e.g., "plans to", "be expected to") with the word "will"; the latter is an example of further preprocessing that could be added to our system. After the preprocessing, Zhang and Patrick create for each sentence pair a feature vector whose values measure the lexical similarity between the two sentences; they appear to be using the maximum number of consecutive common words, the number of common words, edit distance (in words), and modified n-gram precision, a measure similar to BLEU. The produced vectors are then used to train a decision tree classifier. Hence, Zhang and Patrick's approach is similar to ours, but we use more and different similarity measures and several levels of abstraction of the two sentences. We also use ME, along with a wrapper approach to feature selection, rather than decision tree induction and its embedded information gain-based feature selection. Furthermore, all of our methods, even INIT which employs no parsing at all, achieve better results compared to Zhang and Patrick's.

Qiu et al. (2006) first convert the sentences into tuples using parsing and semantic role labeling. They then match similar tuples across the two sentences, and use an SVM (Vapnik, 1998) classifier to decide whether or not the tuples that have not been matched are important or not. If not, the sentences are paraphrases. Despite using a parser and a semantic role identifier, Qiu et al.'s system performs worse than our methods.

Finally, Finch et al.'s system (2005) achieved the second best overall results by employing POS tagging, synonymy resolution, and an SVM. Interestingly, the features of the SVM correspond to machine translation evaluation metrics, rather than string similarity measures, unlike our system. We plan to examine further how the features of Finch et al. and other ideas from machine translation can be embedded in our system, although INIT+WN+DEP outperforms Finch et al.'s system. Interestingly, even when not using more resources than Finch et al. as in methods INIT and INIT+WN

method	*features*	*accuracy*	*precision*	*recall*	*F-measure*
BASE	–	95.30	98.16	92.32	95.15
INIT'	38	99.62	99.50	99.75	99.62

Table 1: Results (%) of our methods on our MTC dataset.

method	*features*	*accuracy*	*precision*	*recall*	*F-measure*
BASE	1	69.04	72.42	86.31	78.76
INIT	133	75.19	78.51	86.31	82.23
INIT+WN	133	75.48	78.91	86.14	82.37
INIT+WN+DEP	136	76.17	79.35	86.75	82.88
INIT+WN+DEP + FHC	7	73.86	75.14	90.67	82.18
INIT+WN+DEP + FBS	10	73.68	73.68	93.98	82.61
Finch et al.	–	74.96	76.58	89.80	82.66
Qiu et al.	–	72.00	72.50	93.40	81.60
Wan et al.	–	75.00	77.00	90.00	83.00
Zhang & Patrick	–	71.90	74.30	88.20	80.70

Table 2: Results (%) of our methods (upper part) and other methods (lower part) on the MSR corpus.

we achieve similar or better accuracy results.

4 Related work

We have already made the distinction between paraphrase (and textual entailment) *generators* vs. *recognizers*, and we have pointed out that recognizers can be embedded in generators as filters. The latter is particularly useful in bootstrapping paraphrase generation approaches (Riloff and Jones, 1999; Barzilay and McKeown, 2001; Ravichandran and Hovy, 2001; Ravichandran et al., 2003; Duclaye et al., 2003; Szpektor et al., 2004), which are typically given seed pairs of named entities for which a particular relation holds; the system locates in a document collection (or the entire Web) contexts were the seeds cooccur, and uses the contexts as patterns that can express the relation; the patterns are then used to locate new named entities that satisfy the relation, and a new iteration begins. A paraphrase recognizer could be used to filter out erroneous generated paraphrases between iterations.

Another well known paraphrase generator is Lin and Pantel's (2001) DIRT, which produces slotted semantically equivalent patterns (e.g., "X is the writer of Y" $\Leftrightarrow$ "X wrote Y" $\Leftrightarrow$ "Y was written by X" $\Leftrightarrow$ "X is the author of Y"), based on the assumption that different paths of dependency trees (obtained from a corpus) that occur frequently with the same words (slot fillers) at their ends are often paraphrases. An extension of DIRT, named LEDIR, has also been proposed (Bhagat et al., 2007) to recognize directional textual entailment rules (e.g., "Y was written by X" $\Rightarrow$ "Y is the work of X"). Ibrahim et al.'s (2003) method is similar to DIRT, but it uses only dependency grammar paths from aligned sentences (from a parallel corpus) that share compatible anchors (e.g., identical strings, or entity names of the same semantic category). Shinyama and Sekine (2003) adopt a very similar approach.

In another generation approach, Barzilay and Lee (2002; 2003) look for pairs of slotted word lattices that share many common slot fillers; the lattices are generated by applying a multiple-sequence alignment algorithm to a corpus of multiple news articles about the same events. Finally, Pang et al. (2003) create finite state automata by merging parse trees of aligned sentences from a parallel corpus; in each automaton, different paths represent paraphrases. Again, a paraphrase recognizer could be embedded in all of these methods, to filter out erroneous generated patterns.

5 Conclusions and further work

We have presented three methods (INIT, INIT+WN, INIT+WN+DEP) that recognize paraphrases given pairs of sentences. These methods employ nine string similarity measures applied to ten shallow abstractions of the input sentences. Moreover, INIT+WN and INIT+WN+DEP exploit WordNet for synonymy resolution, and INIT+WN+DEP uses additional features that measure grammatical relation similarity. Supervised machine learning is used to learn how to combine the resulting features. We experimented with a Maximum Entropy classifier on two datasets; the publicly available MSR corpus and one that we constructed from the

MTC corpus. However, the latter was found to be very easy, and consequently we mainly focused on the MSR corpus.

On the MSR corpus, all of our methods achieved similar or better performance than the sate of the art, even INIT, despite the fact that it uses fewer linguistic resources. Hence, INIT may have practical advantages in less spoken languages, which have limited resources. The most elaborate of our methods, INIT+WN+DEP, achieved the best results, but it requires WordNet and a reliable dependency grammar parser. Feature selection experiments indicate that there is significant redundancy in our feature set, though the full feature set leads to better performance than the subsets produced by feature selection. Further improvements may be possible by including in our system additional features, such as BLEU scores or features for word alignment.

Our long-term goal is to embed our recognizer in a bootstrapping paraphrase generator, to filter out erroneous paraphrases between bootstrapping iterations. We hope that our recognizer will be adequate for this purpose, possibly in combination with a human in the loop, who will inspect paraphrases the recognizer is uncertain of.

Acknowledgements

This work was funded by the Greek PENED 2003 programme, which is co-funded by the European Union (80%), and the Greek General Secretariat for Research and Technology (20%).

References

R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. 2006. The 2nd PASCAL recognising textual entailment challenge. In *Proceedings of the 2nd* PASCAL *Challenges Workshop on Recognising Textual Entailment*, Venice, Italy.

R. Barzilay and L. Lee. 2002. Bootstrapping lexical choice via multiple-sequence alignment. In *Proceedings of* EMNLP, pages 164–171, Philadelphia, PA.

R. Barzilay and L. Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of* HLT-NAACL, pages 16–23, Edmonton, Canada.

R. Barzilay and K. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of* ACL/EACL, pages 50–57, Toulouse, France.

R. Bhagat, P. Pantel, and E. Hovy. 2007. LEDIR: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of the* EMNLP-CONLL, pages 161–170.

I. Dagan, O. Glickman, and B. Magnini. 2006. The PASCAL recognising textual entailment challenge. In Quiñonero-Candela et al., editor, LNAI, volume 3904, pages 177–190. Springer-Verlag.

B. Dolan, C. Quirk, and C. Brockett. 2004. Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In *Proceedings of* COLING, page 350, Morristown, NJ.

F. Duclaye, F. Yvon, and O. Collin. 2003. Learning paraphrases to improve a question-answering system. In *Proceedings of the* EACL *Workshop on Natural Language Processing for Question Answering Systems*, pages 35–41, Budapest, Hungary.

A. Finch, Y. S. Hwang, and E. Sumita. 2005. Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of the 3rd International Workshop on Paraphrasing*, Jeju Island, Korea.

D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan. 2007. The third Pascal recognizing textual entailment challenge. In *Proceedings of the* ACL-*Pascal Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, Czech Republic.

I. J. Good. 1963. Maximum entropy for hypothesis formulation, especially for multidimentional contigency tables. *Annals of Mathematical Statistics*, 34:911–934.

I.M. Guyon, S.R. Gunn, M. Nikravesh, and L. Zadeh, editors. 2006. *Feature Extraction, Foundations and Applications*. Springer.

S. Harabagiu and A. Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of* COLING-ACL, pages 905–912, Sydney, Australia.

S.M. Harabagiu, S.J. Maiorano, and M.A. Pasca. 2003. Open-domain textual question answering techniques. *Natural Language Engineering*, 9(3):231–267.

A. Ibrahim, B. Katz, and J. Lin. 2003. Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the* ACL *Workshop on Paraphrasing*, pages 57–64, Sapporo, Japan.

E. T. Jaynes. 1957. Information theory and statistical mechanics. *Physical Review*, 106:620–630.

D. Lin and P. Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Engineering*, 7:343–360.

P. Malakasiotis and I. Androutsopoulos. 2007. Learning textual entailment using svms and string similarity measures. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 42–47, Prague, June. Association for Computational Linguistics.

B. Pang, K. Knight, and D. Marcu. 2003. Syntax-based alignment of multiple translations: extracting paraphrases and generating new sentences. In *Proceedings of* HLT-NAACL, pages 102–109, Edmonton, Canada.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318, Philadelphia, Pennsylvania.

L. Qiu, M. Y. Kan, and T.S. Chua. 2006. Paraphrase recognition via dissimilarity significance classification. In *Proceedings of* EMNLP, pages 18–26, Sydney, Australia.

D. Ravichandran and E. Hovy. 2001. Learning surface text patterns for a question answering system. In *Proceedings of* ACL, pages 41–47, Philadelphia, PA.

D. Ravichandran, A. Ittycheriah, and S. Roukos. 2003. Automatic derivation of surface text patterns for a maximum entropy based question answering system. In *Proceedings of* HLT-NAACL, pages 85–87, Edmonton, Canada.

E. Riloff and R. Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of* AAAI, pages 474–479, Orlando, FL.

Y. Shinyama and S. Sekine. 2003. Paraphrase acquisition for information extraction. In *Proceedings of the* ACL *Workshop on Paraphrasing*, Sapporo, Japan.

I. Szpektor, H. Tanev, I. Dagan, and B. Coppola. 2004. Scaling Web-based acquisition of entailment relations. In *Proceedings of* EMNLP, Barcelona, Spain.

V. Vapnik. 1998. *Statistical learning theory*. John Wiley.

S. Wan, M. Dras, R. Dale, and C. Paris. 2006. Using dependency-based features to take the “parafarce” out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop*, pages 131–138, Sydney, Australia.

Y. Zhang and J. Patrick. 2005. Paraphrase identification by text canonicalization. In *Proceedings of the Australasian Language Technology Workshop*, pages 160–166, Sydney, Australia.

A System for Semantic Analysis of Chemical Compound Names

Henriette Engelken
EML Research gGmbH
Schloss-Wolfsbrunnenweg 33
69118 Heidelberg, Germany;
Institute for Natural Language Processing
University of Stuttgart
Azenbergstr. 12
70174 Stuttgart, Germany
engelken@eml-research.de

Abstract

Mapping and classification of chemical compound names are important aspects of the tasks of BioNLP. This paper introduces the architecture of a system for the syntactic and semantic analysis of such names. Our system aims at yielding both the denoted chemical structure and a classification of a given name. We employ a novel approach to the task which promises an elegant and efficient way of solving the problem. The proposed system differs significantly from existing systems, in that it is also able to deal with underspecifying names and class names.

1 Introduction

BioNLP is the branch of computational linguistics developing tools and algorithms tailored to the life sciences domain. Scientific and patent literature in this domain are growing at an enormous pace. This results in a valuable resource for researchers, but at the same time it poses the problem that it can hardly be processed manually by humans. Thus, a major goal of BioNLP is to automatically support humans by means of research in the area of information retrieval, data mining and information extraction. Term identification is of great importance in these tasks. Krauthammer and Nenadic (2004) divide the identification task into the subtasks of term recognition (marking the interesting words in a text), term classification (classifying them according to a taxonomy or an ontology) and term mapping[1] (identifying a term with respect to a referent data source).

Chemical compound names, i. e. names of molecules, are terms which prominently occur in scientific publications, patents and in biochemical databases. Any chemical compound can be unambiguously denoted by its molecular structure, either graphically or by certain representation standards. Established representation formats are SMILES strings (Simplified Molecular Input Line Entry System (Weininger, 1988)) and InChIs [2]. For example, a SMILES string such as *CC(OH)CCC* unambiguously describes a chain of five carbon (C) atoms connected by single bonds having an oxygen (O) and a hydrogen (H) atom connected to the second carbon atom by another single bond (Figure 1).

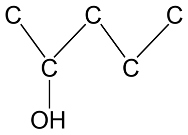


Figure 1: SMILES = *CC(OH)CCC*,
Name = *pentan-2-ol*

However, for communication purposes, e. g. in scientific publications and even in databases, it is common to use names for chemical compounds instead of a structural representation. Contrary to the structural representations, these names are neither always unique nor unambiguous. Biochemical terminology is a subset of natural language which appears to be highly regulated and systematic. The International Union of Pure and Applied Chemistry (IUPAC) (1979; 1993) has developed a nomenclature for chemical compounds. It specifies how to name a molecule systematically, as

[1] Term mapping is also called *term grounding*, amongst others by Kim and Park (2004).

[2] Cf. http://www.iupac.org/inchi/ (accessed May 17, 2009).

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 36–44,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

well as by use of certain trivial names.

The morphemes constituting a name determine the chemical structure it denotes by specifying the type and number of the present atoms and bonds. Morphemes also interact with each other on this structural level. Typically, morphemes describe the atoms and bonds by introducing actions concerning so-called functional groups. About 50 different functional groups can be identified to be the most common ones in organic chemistry.[3] Functional groups are certain groups of atoms which determine the characteristic properties of a molecule, especially its chemical reactions. Hence, the presence or absence of certain functional groups plays a crucial role in classification of chemical compounds. For example, *hydroxy*, used as a prefix of a name, specifies the presence of an OH-group (consisting of an oxygen atom and a hydrogen atom). A molecular structure containing an OH-group can be classified to be an *alcohol*. The morpheme *dehydroxy* in contrast causes deletion of such an OH-group. Thus, it presupposes the existence of some OH-group, which consequently needs to be introduced by another morpheme of the given name. In case there is no additional OH-group left in this molecule after deletion, it does not belong to the class *alcohol*. Apart from addition and deletion, another frequent operation on functional groups, specified by the name's morphemes, is substitution. In this case, a presupposed functional group is replaced by a different functional group. Again, this may change the classes this chemical compound belongs to.

Despite the IUPAC nomenclature, name variations are still in use. On the one hand this is due to competing rules in different editions of the IUPAC nomenclature and on the other hand to the actual usage by chemists who can hardly know every single nomenclature rule. Thus, there can be a number of different names and name types for one chemical compound, namely several systematic, semi-systematic, trivial and trade names. For example, *pentan-2-ol* is the recommended name for the compound in Figure 1, but the same compound can be called *2-pentanol* or *2-hydroxypentane* as well.

Besides synonymy, names allow the omission of specific information about the structure of the compound they denote. This results in not only having a single compound as their reference but a whole set of compounds. Class names like *alcohol* or *alkene* are obvious cases. So-called underspecifying or underspecified[4] names (Reyle, 2006) like *pentanol*, *butene* or *3-chloropropenylidyne* also lack some structural information necessary to fully specify one compound, even though except for this, their names are built according to systematic naming rules. *Pentanol*, for instance, is missing the locant number and could hence stand for *pentan-1-ol*, *pentan-2-ol*, as well as *pentan-3-ol*. We distinguish underspecification from ambiguity, in that underspecifying names do not need to be resolved but denote a set of compounds, analogous to class names.

The particularities of chemical compound names mentioned above, namely synonymy, class names, underspecifying names and interaction between morpheme's meanings, complicate automatic classification and mapping of the names.

To achieve mapping of synonymous chemical compound names, name normalization is a possible approach. Rules can be set up to transform syntactic as well as morphological variations of names into a normalized name form. Basic transformations can be achieved via pattern matching (regular expressions) while for more complex transformations a linguistic parser, yielding a syntactic analysis, would be needed. For example, the names *glyceraldehyde-3-phosphate* and *3-phospho-Glyceraldehyde* could both be normalized to the form *3-phosphoglyceraldehyde* by such rules since the prefix *phospho* is synonymous with the suffix *phosphate*. This way, a synonym relation can be established between any two names which resulted in the same normalized name form. By using this method together with large reference databases[5] providing many synonymous names for their entries, the task of name mapping can be successfully solved in many cases.

However, there are limits to this string based approach. First, it relies on the quality of the referent data source and the quantity of synonyms provided by it. Currently available databases which could be used as a reference lack either quality or quantity. But whether a molecular structure for a term can be determined, or a term classi-

[3] Cf. (Ertl, 2003) and Wikipedia, *Functional group*, http://en.wikipedia.org/wiki/Functional_group (accessed May 17, 2009).

[4] Hereafter we will call these names *underspecifying names* because we consider them to underspecify a chemical structure rather than being underspecified.

[5] E. g. PubChem: http://pubchem.ncbi.nlm.nih.gov/ (accessed May 17, 2009).

fication can be achieved, depends only on this referent data source. Second, it is hardly possible to include every morphosyntactic name variation in the set of transformation rules. *2-hydroxy-3-oxopropyl dihydrogen phosphate*, for example, is the IUPAC name recommended for the chemical compound *glyceraldehyde-3-phosphate*, mentioned above. Obviously, a synonym relation can not be discovered by morphosyntactic name transformations in this case. Finally, this method is not able to deal with class names or underspecifying names.

These observations result in the need to take the meaning of a name's morphemes, i. e. the chemical structure, into account as well. A number of systems for name-to-structure conversion are being developed. The best known commercial systems are Name=Struct[6], ACD/Name[7] and Lexichem[8]. Being commercial, detailed documentation about their methods and evaluation results is not available. Academic approaches are OPSIN (Corbett and Murray-Rust, 2006) and ChemNomParse[9]. The greatest shortcoming of all these approaches is that they are not able to deal with underspecifying names. Instead, they either guess the missing information, in order to determine one specific structure for a given name, or simply fail. But for really underspecifying names and class names, to the best of our knowledge no chemical representation format, like a SMILES string, is provided. In addition, these approaches do not yield any classification of the processed names, regardless of whether these are underspecifying or not.

To overcome these limitations, CHEMorph (Kremer et al., 2006) has been developed. It contains a morphological parser, built according to the IUPAC nomenclature rules. The parser yields a syntactic analysis of a given name and also provides a semantic representation. This semantic representation can be used as a basis for further processing, namely for structure generation or classification. In the CHEMorph project, rules have been set up to achieve these two tasks, but there are limits in the number and correctness of structures and classes retrieved. These limits are partly due to the lack of a comprehensive valence and numbering model for the chemical structures. Also, classification should be based on the structural level rather than on the semantic representation, to ensure that not only the numbering but also default knowledge about chemical structures is included correctly.

The objectives of our own name-to-structure system are the following: Naturally, it should yield a chemical compound structure, in some representation format, as well as a classification for a given name. In case the name does not fully specify one compound, but refers to a set of structures, the system should still allow for structure comparison (mapping) and classification. Several default rules about the names and the chemical structures have to be taken into account. By including default knowledge, a structure can be specified further even if the name itself has left it underspecified. Similarly, a comprehensive way of dealing with valences of atoms has to be included, since the valences restrict the way a chemical structure can be composed.

Our approach to achieve these goals is to use constraint logic programming (CLP). CLP over graph domains is ideal for modeling each name-to-structure task as a so-called constraint satisfaction problem (CSP) and thereby accomplish mapping and classification. We will describe our system, CLP(name2structure), in more detail in the following section.

In this introduction we described the particularities of biochemical terminology. Related work in the area of processing these terms was overviewed and we gave the motivation for our own approach. After presenting our system in Section 2 we will conclude this paper with Section 3, indicating directions for future research.

2 Our Approach

Following Reyle (2006), we observed that any chemical compound name can be seen as a description of a chemical structure – in other words it contains constraints on how the structure is composed. Even if a partial name or a class name does not specify the structure completely but leaves a certain part underspecified, there will at least be some constraints about the structure. On account of this, our proposed system – CLP(name2structure) – employs constraint logic

[6] Cf. http://www.cambridgesoft.com/databases/details/?db=16 (accessed May 17, 2009).

[7] Cf. http://www.acdlabs.com/products/name_lab/rename/batch.html (accessed May 17, 2009).

[8] Cf. http://demo.eyesopen.com/products/toolkits/lexichem-tk_ogham-tk.html (accessed May 17, 2009).

[9] Cf. http://chemnomparse.sourceforge.net/ (accessed May 17, 2009).

programming (CLP) to automatically model so-called constraint satisfaction problems (CSPs) according to given names. Such a CSP captures a name's meaning in that it represents the problem of finding the chemical structure(s) denoted by the name. The solutions to a CSP are determined by a constraint solver. It will find all the structures which satisfy every constraint given by the name. In the case of a fully specified chemical structure, the solution is exactly one structure. This structure is then mapped and classified. For underspecified structures or class names, we distinguish two methods: Either all the structures can be enumerated or the CSP itself can be used for mapping and classification.

Figure 2 shows an overview of the system's architecture. Its component details will be described in the following subsections.

2.1 Parsing and Semantic Representation

We decided to use the CHEMorph parser which is implemented in Prolog. It provides a morpho-semantic grammar which was built according to IUPAC nomenclature rules. The lexicon of this grammar contains the morphemes which can constitute systematic chemical compound names. Also, the lexicon contains a number of trivial and class names. In addition to a syntactic analysis, the CHEMorph parser also yields a semantic representation of the input name. This representation is a term which describes the meaning of the given chemical name in a kind of functor-arguments logic.[10] Example (1), (2) and (3) each show a compound name and its semantic representation generated by CHEMorph:

(1) compound name: *pentan-2,3-diol*
semantic representation: *compd(ane(5*'C'), pref([]), suff([2*[2, 3]-ol]))*

(2) compound name: *2,3-dihydroxy-pentane*
semantic representation: *compd(ane(5*'C'), pref([2*[2, 3]-hydroxy]), suff([]))*

(3) compound name: *propyn-1-imine*
semantic representation: *compd(yne(?? *[??], ane(3*'C')), pref([]), suff([?? *[1]-imine]))*

The general *compd* functor of each semantic representation has three arguments, namely the parent, prefix and suffix representation. The parent argument represents the basic molecular structure, denoted by the parent term of the name. In Example (1) and (2), the parent structure consists of five carbon (C) atoms. This semantic information is encoded with the morpheme *pent* in CHEMorph's lexicon. The parent structure is modified by the functor *ane*, which denotes single bond connections. Prefix and suffix operators, if present, specify further modifications of the basic parent structure. In the case of underspecifying names, as in example (3), the missing pieces of information are represented as *??*.

This way, the semantic representation provides all the information about the chemical structure that is given by the name. Thus, it is an ideal basis for further processing. The next section explains how our system models constraint satisfaction problems on the basis of CHEMorph's semantic representations.

2.2 CSP Modeling

A chemical compound structure can be described as a labeled graph, where the vertices are labeled as atoms and the edges are labeled as bonds. Hence, a chemical compound name can be seen as describing such a graph in that it gives constraints which the graph has to satisfy. In other words, it picks out some specific graph(s) out of the unlimited number of possible graphs in the universe by constraining the possibilities. This observation serves us as a basis for modeling the name-to-structure task as a constraint satisfaction problem (CSP).

A CSP represents a problem as a collection of constraints over a collection of variables. Each of the variables has a domain, which is the set of possible values the variable can take. For the reasons named above, we are working with graph variables and graph domains. The number of chemical compounds, i. e. graphs, could possibly be infinite but we decided it was reasonable and safe to use finite domains. We hence limit the number of possible atoms and bonds for each compound in some way, e. g. on 500 vertices and the corresponding edges or another number estimated according to the semantic representation of the name being processed.

We implement the CSP in ECLiPSe[11], an open-source constraint logic programming (CLP) sys-

[10] Kremer et al. (2006) define the language of the semantic representation in Extended Backus-Naur Form.

[11] Cf. http://eclipse-clp.org/ (accessed May 17, 2009).

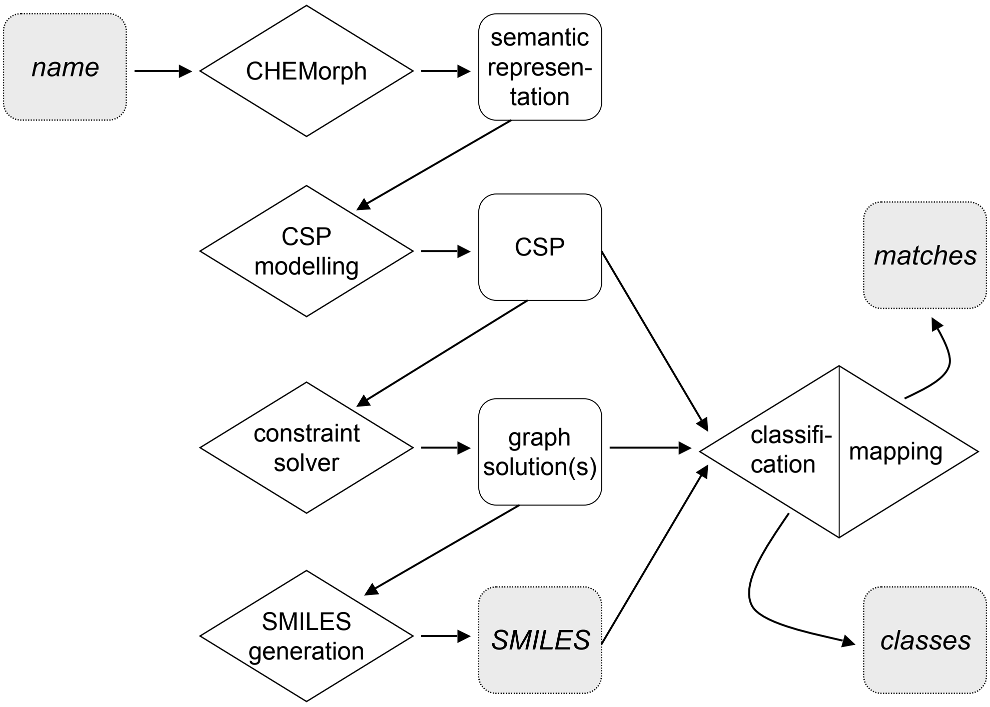


Figure 2: system architecture of CLP(name2structure)

tem, which contains a high-level modeling language, as well as several constraint solver libraries and interfaces for third-party solvers.

To model a CSP for a given input name, several steps have to be taken. First, the semantic representation term provided by CHEMorph has to be parsed. According to its functors and their arguments, the respective constraints have to be called. For this, we are developing a comprehensive set of functions which call the constraints with the correct parameters for the given input name. In these functions, it is determined which constraints over the graph variables a specific functor and argument of the semantic representation is imposing. Thus, in the form of constraints, the functions contain the actions concerning specific functional groups of the denoted molecule, which were described by the name's morphemes. As mentioned in Section 1, these actions include addition, deletion and substitution of certain groups of atoms.

In any case, default rules have to be included while modeling the CSP. Default rules provide constraints about the chemical structures which are not mentioned by any morpheme of the name. For our system they are collected from IUPAC rules as well as from expert knowledge. For example, H-saturation is a default which applies to every chemical compound. This means that every atom of a structure, whose valences are not all occupied by other atoms, has as many H-atoms attached to it as there were free valences. This is one of the reasons why the valences of all the different types of atoms need to be taken into account. We decided to include them as axioms for our models. Knowledge about valences also proves useful for the resolution of underspecification in the case of partial names. Consider a name like *propyn-1-imine* (cf. example (3) in Section 2.1) where it is not specified where the triple bond (denoted by *yn*) is located. However, there are only three C-atoms (introduced by *prop*) to consider, the first of which is connected to an N-atom with a double bond (introduced by *1-imine*). The valence axioms included in our CSPs determine that C-atoms always have a valence of 4, so the first C-atom has only two free valences left until now, since the =N occupies two of them. Consequently, there cannot be a triple bond connected to the same C-atom, as this would use three valences. Hence, the only possibility left is that the triple bond must be located between the second and third C-atom. With the given constraints and axioms, the system is thus able to infer the fully specified compound structure of what would correctly have to be named *prop-2-yn-1-imine* (Figure 3).

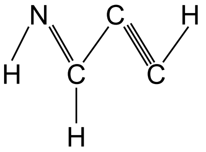


Figure 3: prop-2-yn-1-imine

After modeling a CSP according to the semantic represenation of the input name, the next step in processing is to run a constraint solver. This will be described in the following section.

2.3 Constraint Solver

A constraint solver is a library of tests and operations on constraints. Its purpose is to decide for every conjunction of constraints whether there is a model, i. e. a variable assignment, that satisfies these constraints. This is achieved by consistency checking as well as search techniques, taking the respective variable domains, i. e. the possible values, into account. Besides just deciding whether there is a model for a given CSP, a constraint solver is also able to yield the successful variable assignment(s).

In CLP(name2structure) we use GRASPER[12] (Viegas and Azevedo, 2007), a graph constraint solver based on set constraints. GRASPER enables us to model CSPs using graph variables. In GRASPER, a graph is defined by its set of vertices and its set of edges. Therefore, the domain of a graph consists of a set of possible vertices, in our case for the atoms, and possible edges, in our case for the bonds. The constraints can then narrow these two sets in several ways. For example, certain vertices can be defined to be included as well as the cardinality of a set can be constrained. Also, subgraphs can be defined independently which are then constrained to be part of the final graph solution.

The constraint solver finds one graph solution for graphs which are fully specified by the constraints our system models according to a name. For underspecified graphs, for which the constraints are gathered from underspecifying or class names, the constraint solver could find and enumerate all possible graph solutions if this is desired. This outcome would be the set of all chemical graphs which satisfy the constraints known so far. For example, *chlorohexane* would lead to the set of graphs representing *1-chlorohexane*, *2-chlorohexane* and *3-chlorohexane*.

In general, a chemical name-to-structure system aims at providing the chemical structures in a standard representation format, rather than in a graph notation. In our system, the SMILES generation component carries out this step.

2.4 Generation of a Structural Representation Format

Once a graph is derived from the input name as a solution to its CSP, it specifies the chemical structure completely. It contains the existent vertices and the edges between them, together with labels indicating their respective types and other information like the numbering of atoms. Thus, no additional information has to be considered to generate a chemical representation format from the graph. We focus on generating SMILES strings, rather than some other format, because SMILES themselves use the concept of a graph for representing the molecular structures (Weininger, 1988). For example, the graph solution determined for *pentan-2,3-diol* as well as for *2,3-dihydroxy-pentane* (cf. example (1) and (2) in Section 2.1) can be translated into the SMILES string *CC(OH)C(OH)CC*. In case more than one graph is determined as solution to the CSP (for underspecifying and class names), all the respective SMILES strings could be generated.

Once a SMILES string has successfully been generated, the name-to-structure task is fulfilled and the SMILES string can then be used for tasks such as mapping, classification, picture generation and the like. The next section will describe how classification – one of our main objectives – is accomplished in our approach.

2.5 Classification

Our system offers three different procedures for compound classification. Selection of the appropriate procedure depends on the starting point which could either be a SMILES string, a graph (or a set of graphs) or a CSP.

First, a given SMILES string can be classified based on the functional groups it is comprised of. We use the SMILES classification tool described by Wittig et al. (2004).

[12] GRASPER is distributed with recent builds of the ECLiPSe CLP system.

Second, a graph which is found as solution to a CSP representing an input name can be classified according to a given set of class names. This could for example be some taxonomy which is freely available (like ChEBI (Degtyarenko et al., 2008)). Those class names first have to be transformed into CSPs by use of the parsing and modeling modules of the CLP(name2structure) system. Subsequently, the constraint solver checks whether the graph, or even a set of graphs in the case of an underspecified compound, is a solution to a CSP representing one of the given class names. If the graph or the set of graphs are solutions to one of these CSPs, the compound belongs to the class which provided that CSP. The constraints for the class name *alcohol* for instance, include (amonst others) the presence of an OH-group. Consequently, *pentanol* can be determined to be an alcohol, since its three graph solutions, representing *pentan-1-ol*, *pentan-2-ol* and *pentan-3-ol*, each satisfy the constraints given by *alcohol*.

Third, for some underspecifying names and for class names, it would not be reasonable to generate and classify all the graph solutions or all the SMILES strings – it could simply be too many or even infinitely many. That would slow down performance significantly. Therefore, the system also aims at classifying CSPs themselves, by comparing them directly. If the constraints of CSP-1 are a subset of the constraints of CSP-2, the name which provided CSP-2 is classified to be a hyponym of the more general name which provided CSP-1.

Besides classification, our system aims at mapping chemical compounds. The last module of our system therefore provides algorithms to fulfill this task.

2.6 Mapping

Mapping is needed to fulfill the identification task and to resolve coreference of synonyms. Given a referent data source of chemical compounds, an identity relation should be established if the currently processed compound can successfully be mapped to one of the entries. Again, the procedure depends on whether there is a SMILES string, a set of graph solutions or a CSP to be mapped.

First, matching a SMILES string can be done by simple string comparison. An identity relation between any two compounds holds if their unique SMILES strings (Weininger et al., 1989) match exactly. For example, this is the case for *pentan-2,3-diol* and *2,3-dihydroxy-pentane* since they both yield the same SMILES string (cf. Sections 2.1 and 2.4).

Second, if an underspecifying input name leads to an enumerable number of graph solutions, the set of all the corresponding SMILES strings can be generated. Subsequently, it can be compared to the sets of SMILES strings having been determined for the underspecifying names of the referent data source. If it equals one of the reference SMILES sets, the input name and the respective reference name are successfully identified and thus detected to be synonyms.

Third, mapping of CSPs becomes necessary for class names and underspecifying names with too many graph solutions to enumerate. This works analogously to CSP classification described in Section 2.5 above. The only difference is that a synonym relation between two names, leading to CSP-1 and CSP-2 respectively, is established if the constraints of CSP-1 equal the constraints of CSP-2.

3 Conclusions and Future Work

In this paper we presented the architecture of CLP(name2structure), a system for semantic and syntactic processing of chemical compound names. In the introductory section, we described the characteristic phenomena of biochemical terminology which challenge any such system. Our approach is composed of several modules, carrying out the defined tasks of structure generation, classification and mapping. By employing a morphological parser and constraint logic programming over graph variables, our approach is able to handle the particularities of the chemical compound names.

However, the proposed system CLP(name2structure) still requires work on several of its components. The central task to be completed is to enrich the repository of functions which call the appropriate constraints corresponding to CHEMorph's semantic representation output. This is not a trivial task since it requires to formalize the IUPAC rules of syntax and semantics of the relevant morphemes. This formalization needs to result in an abstract description of the respective constraints over graph variables. Thereby, phenomena like interaction of morphemes' meanings play an important role.

Before we can accomplish the implementation of the complete system according to the proposed architecture, we need to answer a couple of remaining open questions. For example, the exact method on how to compare two CSPs has to be elaborated. Gennari (2002) describes algorithms for normalizing CSPs to enable subsequent equivalence checking. However, these methods can not be applied to our case as they stand but will have to be substantially adapted. Another problem we need to deal with is that labeled graphs, which are required by our system, are not directly supported by the constraint solver GRASPER. Therefore we are currently working on a way to handle the labels indirectly.

Another important task we plan to carry out in the future is the evaluation of CLP(name2structure). Since no gold standard for name-to-structure generation or classification is available yet, such a gold standard or dataset needs to be created first. We propose to use as such a dataset a subset of the entries of an existing curated database, such as ChEBI, which contains names, chemical structures and a classification for currently 17842 compounds. Unless the morphological parser and the repository of constraint functions is further enriched, we suppose our system will yield a high precision rather than a high coverage. To evaluate underspecification handling of our system, underspecifying names from general reaction descriptions[13] could be collected. For this kind of evaluation, determining the correctness of the analysis would require the help of domain experts.

Acknowledgments

The author is funded by the Klaus Tschira Foundation gGmbH, Heidelberg, Germany. Thanks to Uwe Reyle and Fritz Hamm from the University of Stuttgart, Germany, for contributing to the main ideas and for in-depth discussions. Thanks to the Scientific Databases and Visualization group of EML Research, Heidelberg, Germany, for their support. Thanks to Ruben Viegas for comments on graph constraint solving. Thanks to Berenike Litz and the anonymous reviewers for comments on this paper.

[13] As listet by the Enzyme Nomenclature Recommendations: http://www.chem.qmul.ac.uk/iubmb/enzyme/ (accessed May 17, 2009).

References

IUPAC. Commission on the Nomenclature of Organic Chemistry. 1993. *A Guide to IUPAC Nomenclature of Organic Compounds (Recommendations 1993).* Blackwell Scientific Publications, Oxford.

Peter Corbett and Peter Murray-Rust. 2006. High-Throughput Identification of Chemistry in Life Science Texts. *CompLife*, pages 107–118.

Kirill Degtyarenko, Paula de Matos, Marcus Ennis, Janna Hastings, Martin Zbinden, Alan McNaught, Rafael Alcántara, Michael Darsow, Mickaël Guedj, and Michael Ashburner. 2008. ChEBI: a database and ontology for chemical entities of biological interest. *Nucleic Acids Research*, 36(Database-Issue):344–350.

Peter Ertl. 2003. Cheminformatics Analysis of Organic Substituents: Identification of the Most Common Substituents, Calculation of Substituent Properties, and Automatic Identification of Drug-like Bioisosteric Groups. *Journal of Chemical Information and Computer Science*, 43:374–380.

Rosella Gennari. 2002. *Mapping Inferences. Constraint Propagation and Diamond Satisfaction.* Ph.D. thesis, Universiteit van Amsterdam.

Jung-jae Kim and Jong C. Park. 2004. BioAR: Anaphora Resolution for Relating Protein Names to Proteome Database Entries. In *Proceedings of the Reference Resolution and its Applications Workshop in Conjunction with ACL 2004*, pages 79–86.

Michael Krauthammer and Goran Nenadic. 2004. Term Identification in the Biomedical Literature. *Journal of Biomedical Informatics*, 37(6):512–526.

Gerhard Kremer, Stefanie Anstein, and Uwe Reyle. 2006. Analysing and Classifying Names of Chemical Compounds with CHEMorph. In Sophia Ananiadou and Juliane Fluck, editors, *Proceedings of the Second International Symposium on Semantic Mining in Biomedicine,* Friedrich-Schiller-Universität Jena, Germany, 2006, pages 37–43.

IUPAC. Commission on the Nomenclature of Organic Chemistry. 1979. *Nomenclature of Organic Chemistry, Sections A, B, C, D, E, F and H.* Pergamon Press, Oxford.

Uwe Reyle. 2006. Understanding Chemical Terminology. *Terminology*, 12(1):111–136.

Ruben Viegas and Francisco Azevedo. 2007. GRASPER: A Framework for Graph CSPs. In Jimmy Lee and Peter Stuckey, editors, *Proceedings of the Sixth International Workshop on Constraint Modelling and Reformulation (ModRef'07)*, Providence, Rhode Island, USA.

David Weininger, Arthur Weininger, and Joseph L. Weininger. 1989. SMILES 2. Algorithm for

Generation of Unique SMILES Notation. *Journal of Chemical Information and Computer Science*, 29(2):97–101.

David Weininger. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36.

Ulrike Wittig, Andreas Weidemann, Renate Kania, Christian Peiss, and Isabel Rojas. 2004. Classification of chemical compounds to support complex queries in a pathway database. *Comparative and Functional Genomics*, 5:156–162.

Sentence Diagram Generation Using Dependency Parsing

Elijah Mayfield
Division of Science and Mathematics
University of Minnesota, Morris
mayf0016@morris.umn.edu

Abstract

Dependency parsers show syntactic relations between words using a directed graph, but comparing dependency parsers is difficult because of differences in theoretical models. We describe a system to convert dependency models to a structural grammar used in grammar education. Doing so highlights features that are potentially overlooked in the dependency graph, as well as exposing potential weaknesses and limitations in parsing models. Our system performs automated analysis of dependency relations and uses them to populate a data structure we designed to emulate sentence diagrams. This is done by mapping dependency relations between words to the relative positions of those words in a sentence diagram. Using an original metric for judging the accuracy of sentence diagrams, we achieve precision of 85%. Multiple causes for errors are presented as potential areas for improvement in dependency parsers.

1 Dependency parsing

Dependencies are generally considered a strong metric of accuracy in parse trees, as described in (Lin, 1995). In a dependency parse, words are connected to each other through relations, with a head word (the governor) being modified by a dependent word. By converting parse trees to dependency representations before judging accuracy, more detailed syntactic information can be discovered. Recently, however, a number of dependency parsers have been developed that have very different theories of a correct model of dependencies.

Dependency parsers define syntactic relations between words in a sentence. This can be done either through spanning tree search as in (McDonald et al., 2005), which is computationally expensive, or through analysis of another modeling system, such as a phrase structure parse tree, which can introduce errors from the long pipeline. To the best of our knowledge, the first use of dependency relations as an evaluation tool for parse trees was in (Lin, 1995), which described a process for determining heads in phrase structures and assigning modifiers to those heads appropriately. Because of different ways to describe relations between negations, conjunctions, and other grammatical structures, it was immediately clear that comparing different models would be difficult. Research into this area of evaluation produced several new dependency parsers, each using different theories of what constitutes a correct parse. In addition, attempts to model multiple parse trees in a single dependency relation system were often stymied by problems such as differences in tokenization systems. These problems are discussed by (Lin, 1998) in greater detail. An attempt to reconcile differences between parsers was described in (Marneffe et al., 2006). In this paper, a dependency parser (from herein referred to as the Stanford parser) was developed and compared to two other systems: MINIPAR, described in (Lin, 1998), and the Link parser of (Sleator and Temperley, 1993), which uses a radically different approach but produces a similar, if much more fine-grained, result.

Comparing dependency parsers is difficult. The main problem is that there is no clear way to compare models which mark dependencies differently. For instance, when clauses are linked by a conjunction, the Link parser considers the conjunction related to the subject of a clause, while the Stanford parser links the conjunction to the verb of a clause. In (Marneffe et al., 2006), a simple comparison was used to alleviate this problem, which was based only on the presence of dependencies, without semantic information. This solution loses

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 45–53,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

information and is still subject to many problems in representational differences. Another problem with this approach is that they only used ten sentences for comparison, randomly selected from the Brown corpus. This sparse data set is not necessarily congruous with the overall accuracy of these parsers.

In this paper, we propose a novel solution to the difficulty of converting between dependency models. The options that have previously been presented for comparing dependency models are either too specific to be accurate (relying on annotation schemes that are not adequately parallel for comparison) or too coarse to be useful (such as merely checking for the existence of dependencies). By using a model of language which is not as fine-grained as the models used by dependency parsers, but still contains some semantic information beyond unlabelled relations, a compromise can be made. We show that using linear diagramming models can do this with acceptable error rates, and hope that future work can use this to compare multiple dependency models.

Section 2 describes structural grammar, its history, and its usefulness as a representation of syntax. Section 3 describes our algorithm for conversion from dependency graphs to a structural representation. Section 4 describes the process we used for developing and testing the accuracy of this algorithm, and Section 5 discusses our results and a variety of features, as well as limitations and weaknesses, that we have found in the dependency representation of (Marneffe et al., 2006) as a result of this conversion.

2 Introduction to structural grammar

Structural grammar is an approach to natural language based on the understanding that the majority of sentences in the English language can be matched to one of ten patterns. Each of these patterns has a set of slots. Two slots are universal among these patterns: the subject and the predicate. Three additional slots may also occur: the direct object, the subject complement, and the object complement. A head word fills each of these slots. In addition, any word in a sentence may be modified by an additional word. Finally, anywhere that a word could be used, a substitution may be made, allowing the position of a word to be filled by a multiple-word phrase or an entire subclause, with its own pattern and set of slots.

To understand these relationships better, a standardized system of sentence diagramming has been developed. With a relatively small number of rules, a great deal of information about the function of each word in a sentence can be represented in a compact form, using orientation and other spatial clues. This provides a simpler and intuitive means of visualizing relationships between words, especially when compared to the complexity of directed dependency graphs. For the purposes of this paper, we use the system of diagramming formalized in (Kolln and Funk, 2002).

2.1 History

First developed in the early 20th century, structural grammar was a response to the prescriptive grammar approach of the time. Structural grammar describes how language actually is used, rather than prescribing how grammar should be used. This approach allows an emphasis to be placed on the systematic and formulaic nature of language. A key change involved the shift to general role-based description of the usage of a word, whereas the focus before had been on declaring words to fall into strict categories (such as the eight parts of speech found in Latin).

Beginning with the work of Chomsky in the 1950s on transformational grammar, sentence diagrams, used in both structural and prescriptive approaches, slowly lost favor in educational techniques. This is due to the introduction of transformational grammar, based on generative theories and intrinsic rules of natural language structure. This generative approach is almost universally used in natural language processing, as generative rules are well-suited to computational representation. Nevertheless, both structural and transformational grammar are taught at secondary and undergraduate levels.

2.2 Applications of structural grammar

Structural grammar still has a number of advantages over generative transformational grammar. Because it is designed to emulate the natural usage of language, it is more intuitive for non-experts to understand. It also highlights certain features of sentences, such as dependency relationships between words and targets of actions. Many facets of natural language are difficult to describe using a parse tree or other generative data structure. Using structural techniques, many of these aspects are obvious upon basic analysis.

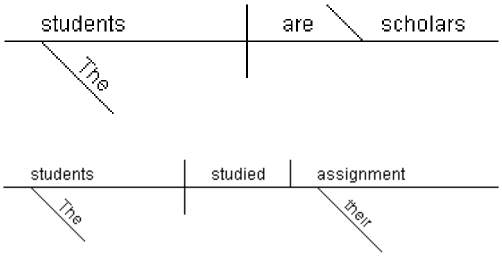


Figure 1: Diagram of "The students are scholars." and "The students studied their assignment."

By developing an algorithm to automatically analyze a sentence using structural grammar, we hope that the advantages of structural analysis can improve the performance of natural language parsers. By assigning roles to words in a sentence, patterns or structures in natural language that cannot be easily gleaned from a data structure are made obvious, highlighting the limitations of that structure. It is also important to note that while sentence diagrams are primarily used for English, they can be adapted to any language which uses subjects, verbs, and objects (word order is not important in sentence diagramming). This research can therefore be expanded into multilingual dependency parser systems in the future.

To test the effectiveness of these approaches, a system must be developed for structural analysis of sentences and subsequent conversion to a sentence diagram.

3 Sentence diagram generation algorithm

In order to generate a sentence diagram, we make use of typed dependency graphs from the Stanford dependency parser. To understand this process requires understanding both the underlying data structure representing a sentence diagram, and the conversion from a directed graph to this data structure.

3.1 Data structure

In order to algorithmically convert dependency parses to a structural grammar, we developed an original model to represent features of sentence diagrams. A sentence is composed of four slots (*Subject*, *Predicate*, *Object*, *Complement*). These slots are represented[1] in two sentences shown in Figure 1 by the words "students," "are," "assignment," and "scholars" respectively. Each slot contains three sets (*Heads*, *Expletives*, *Conjunctions*). With the exception of the *Heads* slot in *Subject* and *Predicate*, all sets may be empty. These sets are populated by words. A word is comprised of three parts: the string it represents, a set of modifying words, and information about its orientation in a diagram. Finally, anywhere that a word may fill a role, it can be replaced by a phrase or subclause. These phrases are represented identically to clauses, but all sets are allowed to be empty. Phrases and subclauses filling the role of a word are connected to the slot they are filling by a pedestal, as in Figure 2.

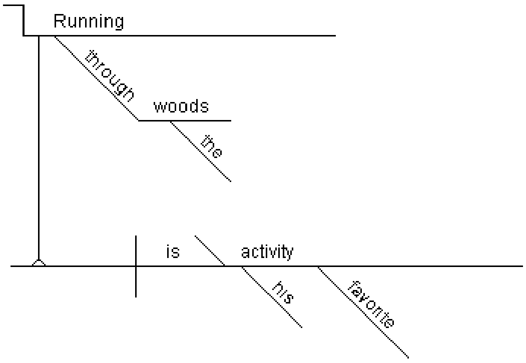


Figure 2: Diagram of "Running through the woods is his favorite activity."

3.2 Conversion from dependency graph

A typed dependency representation of a sentence contains a root – that is, a dependency relation in which neither the governor nor the dependent word in the relation is dependent in any other relation. We use this relation to determine the predicate of a sentence, which is almost always the governor of the root dependency. The dependent is added to the diagram data structure based on its relation to the governor.

Before analysis of dependency graphs begins, our algorithm takes in a set of dependency relations *S* and a set of actions (possible objects and methods to call) *A*. This paper describes an algorithm that takes in the 55 relations from (Marneffe et al., 2006) and the actions in Table 1. The algorithm then takes as input a directed graph *G* representing a sentence, composed of a node rep-

[1] All sentence diagram figures were generated by the algorithm described in this paper. Some diagrams have been edited for spacing and readability concerns. These changes do not affect their accuracy.

resenting each word in the sentence. These nodes are connected by edges in the form `reln(gov, dep)` representing a relation from S between a word `gov` and `dep`. Our algorithm performs the following steps:

1. **Determining root actions:** For each relation type $R \in S$, create an ordered list of actions $Root < R, A >$ from A to perform if that relation is the root relation in the graph.
2. **Determining regular actions:** For each relation type $R \in S$, create an ordered list of actions $Reln < R, A >$ from A to perform if R is found anywhere other than the root in G.
3. **Determining the root:** Using the root-finding process described in (Marneffe et al., 2006), find the root relation $\hat{R}(\hat{G}, \hat{D}) \in G$.
4. **Initialize a sentence diagram:** Find the set of actions$\hat{A}$ from $Root < \hat{R}, A >$and perform those actions.
5. **Finding children:** Create a set *Open* and add to it each relation $\in G$ in which $\hat{G}$ or $\hat{D}$ from step 3 is a governor.
6. **Processing children:** For each relation $\tilde{R}(\tilde{G}, \tilde{D})$ in *Open*,
   (a) **Populate the sentence diagram:** Find the set of actions $\tilde{A}$ from $Reln < \tilde{R}, A >$ and perform those actions.
   (b) **Finding children:** Add to *Open* each relation $R \in G$ in which $\tilde{G}$ or $\tilde{D}$ is a governor.

   This step continues until all relations have been found in a breadth-first order.

Our system of conversion makes the assumption that the governor of a typed dependency will already have been assigned a position in a diagram. This is due to the largely tree-like structure of dependency graphs generated by the dependency parser. Dependencies in most cases "flow" downwards to the root, and in exceptions, such as cycles, the governor will have been discovered by the time it is reached again. As we are searching for words breadth-first, we know that the dependent of any relation will have been discovered already so long as this tree-like structure holds. The number of cases where it does not is small compared to the overall error rate of the dependency parser, and does not have a large impact on the accuracy of the resulting diagram.

3.3 Single-relation analysis

A strength of this system for conversion is that information about the overall structure of a sentence is not necessary for determining the role of each individual word as it is added to the diagram. As each word is traversed, it is assigned a role relative to its parent only. This means that overall structure will be discovered naturally by tracing dependencies throughout a graph.

There is one exception to this rule: when comparing relationships of type *cop* (copula, a linking verb, usually a variant of "to be"), three words are involved: the linking verb, the subject, and the subject complement. However, instead of a transitive relationship from one word to the next, the parser assigns the subject and subject complement as dependent words of the linking verb. An example is the sentence "The students are scholars" as in Figure 1. This sentence contains three relations:

```
det(students, The)
nsubj(scholars, students)
cop(scholars, are)
```

A special case exists in our algorithm to check the governor of a *cop* relation for another relation (usually *nsubj*). This was a necessary exception to make given the frequency of linking verbs in the English language. Dependency graphs from (Marneffe et al., 2006) are defined as a singly rooted directed acyclic graph with no reentrancies; however, they sometimes share nodes in the tree, with one word being a dependent of multiple relations. An example of this exists in the sentence "I saw the man who loves you." The word "who" in this sentence is dependent in two relations:

```
ref(man, who)
rel(loves, who)
```

We here refer to this phenomenon as breaking the tree structure. This is notable because it causes a significant problem for our approach. While the correct relation is identified and assigned in most cases, a duplicated copy of the dependent word will appear in the resulting diagram. This is because the dependent word in each relation is added to the diagram, even if it has already been added. Modifiers of these words are then assigned to each copy, which can result in large areas of duplication. We decided this duplication was acceptable

Term	Definition	Example	
		Input	Output
GOV, DEP, RELN	Elements of a relation	det(``woods", ``the").GOV	``woods"
SBJ, PRD, OBJ, CMP	Slots in a clause	CLAUSE.PRD	HEADS(``is"), EXPL(), CONJ()
HEADS, EXPL, CONJ	Sets of words in a slot	CLAUSE.PRD.HEADS()	``is"
MODS	Set of modifiers of a word	``activity".MODS	(``his", ``favorite")
SEGMENT, CLAUSE	Set or clause of word	``is".SEGMENT()	CLAUSE.PRD
NEW[WORD, Slot]	New clause constructor	NEW(``is", PRD)	CLAUSE(SBJ(), PRD(``is"), OBJ(), CMP())
ADD(WORD[,ORIENT])	Word added to modifiers	``activity".ADD(``his")	
APP(WORD[,RIGHT?])	Word appended to phrasal head	``down".APP(``shut", false)	
SET(ORIENT)	Word orientation set	``his".SET(DIAGONAL)	

Periods represent ownership, parentheses represent parameters passed to a method, separated by commas, and brackets represent optional parameters.

Orientations include HORIZONTAL, DIAGONAL, VERTICAL, GERUND, BENT, DASHED, and CLAUSE as defined in (Kolln and Funk, 2002) .

Table 1: Terms and methods defined in our algorithm.

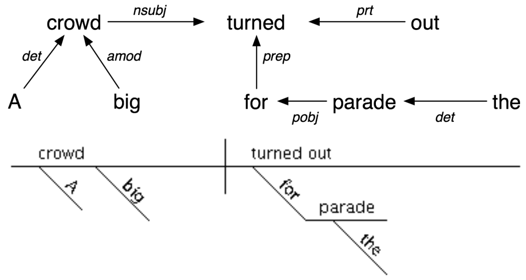


Figure 3: The sentence “A big crowd turned out for the parade.” shown as a dependency graph (top) and a sentence diagram.

to maintain the simplicity of single-relation conversion rules, though remedying this problem is an avenue for further research. For testing purposes, if duplicate copies of a word exist, the correct one is given preference over the incorrect copy, and the diagram is scored as correct if either copy is correctly located.

3.4 An example diagram conversion

To illustrate the conversion process, consider the sentence “A big crowd turned out for the parade.” The dependency graph for this, as generated by the Stanford dependency parser, is shown in Figure 3. The following relations are found, with the actions taken by the conversion algorithm described:

Root: **nsubj(turned, crowd)**

```
NEW(GOV, PRD);
GOV.CLAUSE.SBJ.ADD(DEP);
```

Finding Children: det(crowd, A), amod(crowd, big), prt(turned, out), prep(turned, for) added to *Open*.

Relation: **det(crowd, A)**

```
GOV.ADD(DEP,DIAGONAL);
```

Relation: **amod(crowd, big)**

```
GOV.ADD(DEP,DIAGONAL);
```

Relation: **prt(turned, out)**

```
GOV.APP(DEP,TRUE);
```

Relation: **prep(turned, for)**

Finding Children: pobj(for, parade) added to *Open*.

```
GOV.ADD(DEP,DIAGONAL);
```

Relation: **pobj(for, parade)**

Finding Children: det(parade, the) added to *Open*.

```
GOV.ADD(DEP,HORIZONTAL);
```

Relation: **det(parade, the)**

```
GOV.ADD(DEP,DIAGONAL);
```

4 Experimental setup

In order to test our conversion algorithm, a large number of sentence diagrams were needed in order to ensure a wide range of structures. We decided to use an undergraduate-level English grammar textbook that uses diagramming as a teaching tool for two reasons. The first is a pragmatic matter: the sentences have already been diagrammed accurately for comparison to algorithm output. Second, the breadth of examples necessary to allow students a thorough understanding of the process is beneficial in assuring the completeness of the conversion system. Cases that are especially difficult for students are also likely to be stressed with multiple examples, giving more opportunities to determine the problem if parsers have similar difficulty.

Therefore, (Kolln and Funk, 2002) was selected to be used as the source of this testing data. This textbook contained 292 sentences, 152 from examples and 140 from solutions to problem sets. 50% of the example sentences (76 in total, chosen by selecting every other example) were set aside to use for development. The remaining 216 sentences were used to gauge the accuracy of the conversion algorithm.

Our implementation of this algorithm was developed as an extension of the Stanford dependency parser. We developed two metrics of precision to evaluate the accuracy of a diagram. The first approach, known as the *inheritance* metric, scored the results of the algorithm based on the parent of each word in the output sentence diagram. Head words were judged on their placement in the correct slot, while modifiers were judged on whether they modified the correct parent word. The second approach, known as the *orientation* metric, judged each word based solely on its orientation. This distinction judges whether a word was correctly identified as a primary or modifying element of a sentence.

These scoring systems have various advantages. By only scoring a word based on its immediate parent, a single mistake in the diagram does not severely impact the result of the score, even if it is at a high level in the diagram. Certain mistakes are affected by one scoring system but not the other; for instance, incorrect prepositional phrase attachment will not have an effect on the orientation score, but will reduce the value of the inheritance score. Alternatively, a mistake such as failing to label a modifying word as a participial modifier will reduce the orientation score, but will not reduce the value of the inheritance score. Generally, orientation scoring is more forgiving than inheritance scoring.

5 Results and discussion

The results of testing these accuracy metrics are given in Figure 4 and Table 2. Overall inheritance precision was 85% and overall orientation precision was 92%. Due to the multiple levels of analysis (parsing from tree to phrase structure to dependency graph to diagram), it is sometimes difficult to assign fault to a specific step of the algorithm.

There is clearly some loss of information when converting from a dependency graph to a sentence diagram. For example, fifteen dependency relations are represented as diagonal modifiers in a sentence diagram and have identical conversion rules. Interestingly, these relations are not necessarily grouped together in the hierarchy given in (Marneffe et al., 2006). This suggests that the syntactic information represented by these words may not be as critical as previously thought, given enough semantic information about the words. In total, six sets of multiple dependency relations mapping to the same conversion rule were found, as shown in Table 3.

The vast majority of mistakes that were made came from one of two sources: an incorrect conversion from a correct dependency parse, or a failure of the dependency parser to correctly identify a relation between words in a sentence. Both are examined below.

5.1 Incorrect conversion rules

On occasion, a flaw in a diagram was the result of an incorrect conversion from a correct interpretation in a dependency parse. In some cases, these were because of simple changes due to inaccuracies not exposed from development data. In some cases, this was a result of an overly general relationship, in which one relation correctly describes two or more possible structural patterns in sentences. This can be improved upon by specializing dependency relation descriptions in future versions of the dependency parser.

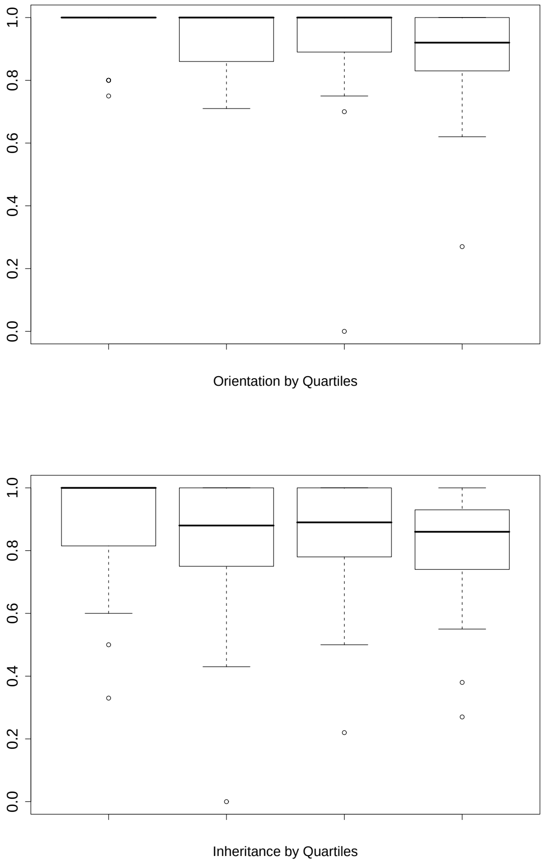
One frequent failure of the conversion rules is due to the overly generalized handling of the root of sentences. It is assumed that the governing word in the root relation of a dependency graph is the main verb of a sentence. Our algorithm has very general rules for root handling. Exceptions to these general cases are possible, especially in

Sentence Length	Ori Mean	Ori Std.Dev.	Inh Mean	Inh Std.Dev.	Count
3-6	96.61	7.42	90.34	15.20	56
7-8	92.37	15.77	86.00	19.34	57
9-10	92.80	8.18	82.73	17.15	45
11-20	89.97	12.54	82.52	15.51	58
3-20	92.91	11.84	85.51	17.05	216

Table 2: Precision of diagramming algorithm on testing data.

Relations	Rule
abbrev, *advmod*, *amod*, *dep*, *det*, *measure*, *neg*, *nn*, *num*, *number*, *poss*, *predet*, *prep*, *quantmod*, *ref*	`GOV.ADD(DEP,DIAGONAL)`
iobj, *parataxis*, *pobj*	`GOV.ADD(DEP,HORIZONTAL)`
appos, *possessive*, *prt*	`GOV.APP(DEP,TRUE)`
aux, *tmod*	`GOV.APP(DEP,FALSE)`
advcl, *csubj*, *pcomp*, *rcmod*	`GOV.ADD(NEW(DEP,PRD))`
complm, *expl*, *mark*	`GOV.SEGMENT.EXPL.ADD(DEP)`

Table 3: Sets of multiple dependency relations which are converted identically.



Orientation by Quartiles

Inheritance by Quartiles

Figure 4: Inheritance (top) and Orientation precision results of diagramming algorithm on testing data. Results are separated by sentence length into quartiles.

interrogative sentences, e.g. the root relation of the sentence “What have you been reading?” is `dobj(reading, What)`. This should be handled by treating “What” as the object of the clause. This problem can be remedied in the future by creating specialized conversion rules for any given relation as a root of a dependency graph.

A final issue is the effect of a non-tree structure on the conversion algorithm. Because relationships are evaluated individually, multiple inheritance for words can sometimes create duplicate copies of a word which are then modified in parallel. An example of this is shown in Figure 5, which is caused due to the dependency graph for this sentence containing the following relations:

```
nsubj(is-4, hope-3)
xsubj(beg-6, hope-3)
xcomp(is-4, beg-6)
```

Because the tree structure is broken, a word (hope) is dependent on two different governing words. While the xsubj relation places the phrase “to beg for mercy” correctly in the diagram, a second copy is created because of the xcomp dependency. A more thorough analysis approach that checks for breaking of the tree structure may be useful in avoiding this problem in the future.

5.2 Exposed weaknesses of dependency parsers

A number of consistent patterns are poorly diagrammed by this system. This is usually due to

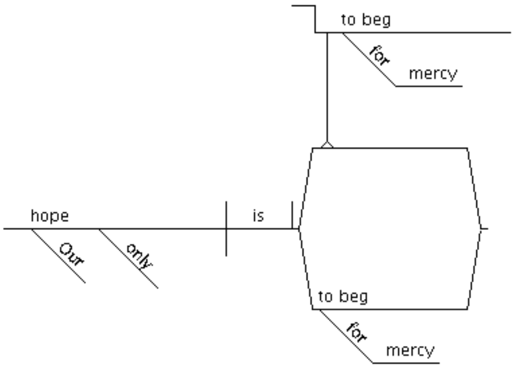


Figure 5: Duplication in the sentence diagram for "Our only hope is to beg for mercy."

limitations in the theoretical model of the dependency parser. These differences between the actual structure of the sentence and the structure the parser assigns can lead to a significant difference in semantic value of phrases. Improving the accuracy of this model to account for these situations (either through more fine-grained separation of relationships or a change in the model) may improve the quality of meaning extraction from sentences.

One major shortcoming of the dependency parser is how it handles prepositional phrases. As described in (Atterer and Schutze, 2007), this problem has traditionally been framed as involving four words (v, n1, p, n2) where v is the head of a verb phrase, n1 is the head of a noun phrase dominated by v, p is the head of a prepositional phrase, and n2 the head of a noun phrase dominated by p. Two options have generally been given for attachment, either to the verb v or the noun n1. This parser struggles to accurately determine which of these two possibilities should be used. However, in the structural model of grammar, there is a third option, treating the prepositional phrase as an object complement of n1. This possibility occurs frequently in English, such as in the sentence "We elected him as our secretary." or with idiomatic expressions such as "out of tune." The current dependency parser cannot represent this at all.

5.3 Ambiguity

A final case is when multiple correct structural analyses exist for a single sentences. In some cases, this causes the parser to produce a gramatically and semantically correct parse which, due to ambiguity, does not match the diagram for comparison. An example of this can be seen in Figure 6, in which the dependency parser assigns the predicate role to "was deserted" when in fact deserted is acting as a subject complement. However, the phrase "was deserted" can accurately act as a predicate in that sentence, and produces a semantically valid interpretation of the phrase.

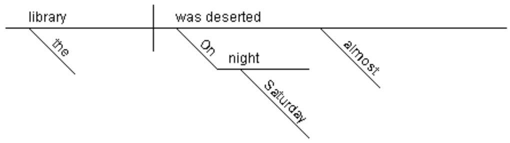


Figure 6: Diagram of "On Saturday night the library was almost deserted."

6 Conclusion

We have demonstrated a promising method for conversion from a dependency graph to a sentence diagram. However, this approach still has the opportunity for a great deal of improvement. There are two main courses of action for future work to reap the benefits of this approach: analyzing current results, and extending this approach to other parsers for comparison. First, a more detailed analysis of current errors should be undertaken to determine areas for improvement. There are two broadly defined categories of error (errors made before a dependency graph is given to the algorithm for conversion, and errors made during conversion to a diagram). However, we do not know what percent of mistakes falls into those two categories. We also do not know what exact grammatical idiosyncracy caused each of those errors. With further examination of current data, this information can be determined.

Second, it must be determined what level of conversion error is acceptable to begin making quantitative comparisons of dependency parsers. Once the level of noise introduced by the conversion process is lowered to the point that the majority of diagram errors are due to mistakes or shortfalls in the dependency graph itself, this tool will be much more useful for evaluation. Finally, this system should be extended to other dependency parsers so that a comparison can be made between multiple systems.

References

Michaela Atterer and Hinrich Schutze. 2007. Prepositional Phrase Attachment without Oracles. In *Com-*

putational Linguistics.

John Carroll, Guido Minnen, and Ted Briscoe. 1999. Corpus annotation for parser evaluation. In *Proceedings of the EACL workshop on Linguistically Interpreted Corpora.*

Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics.*

Martha Kolln and Robert Funk. 2002. Understanding English Grammar, Sixth Edition. *Longman Publishers*.

Dekang Lin. 1995. A Dependency-based Method for Evaluating Broad-Coverage Parsers. In *Natural Language Engineering*.

Dekang Lin. 1998. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*

Marie-Catherine de Marneffe, Bill MacCartney and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *International Conference on Language Resources and Evaluation.*

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing.*

Daniel D. Sleator and Davy Temperley. 1993. Parsing English with a link grammar. In *Third International Conference on Parsing Technologies.*

Accurate Learning for Chinese Function Tags from Minimal Features

Caixia Yuan[1,2], **Fuji Ren**[1,2] **and Xiaojie Wang**[2]
[1]The University of Tokushima, Tokushima, Japan
[2]Beijing University of Posts and Telecommunications, Beijing, China
{yuancai,ren}@is.tokushima-u.ac.jp
xjwang@bupt.edu.cn

Abstract

Data-driven function tag assignment has been studied for English using Penn Treebank data. In this paper, we address the question of whether such method can be applied to other languages and Treebank resources. In addition to simply extend previous method from English to Chinese, we also proposed an effective way to recognize function tags directly from lexical information, which is easily scalable for languages that lack sufficient parsing resources or have inherent linguistic challenges for parsing. We investigated a supervised sequence learning method to automatically recognize function tags, which achieves an F-score of 0.938 on gold-standard POS (Part-of-Speech) tagged Chinese text – a statistically significant improvement over existing Chinese function label assignment systems. Results show that a small number of linguistically motivated lexical features are sufficient to achieve comparable performance to systems using sophisticated parse trees.

1 Introduction

Function tags, such as subject, object, time, location, etc. are conceptually appealing by encoding an event in the format of "who did what to whom, where, when", which provides useful semantic information of the sentences. Lexical semantic resources such as Penn Treebank (Marcus et al., 1994) have been annotated with phrase tree structures and function tags. Figure 1 shows the parse tree with function tags for a sample sentence form the Penn Chinese Treebank 5.0[1] (Xue et al., 2000) (file 0043.fid).

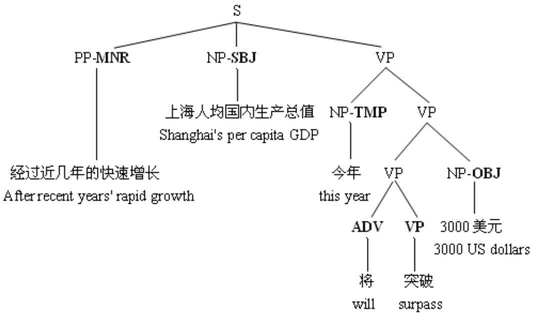


Figure 1: Simplified parse tree with function tags (in black bold) for example sentence.

When dealing with the task of function tag assignment (or function labeling thereafter), one basic question that must be addressed is what features can be extracted in practice for distinguishing different function tag types. In answering this question, several pieces of work (Blaheta and Charniak, 2000; Blaheta, 2004; Merlo and Musillo, 2005; Gildea and Palmer, 2002) have already been proposed. (Blaheta and Charniak, 2000; Blaheta, 2004) described a statistical system trained on the data of Penn Treebank to automatically assign function tags for English text. The system first passed sentences through an automatic parser, then extracted features from the parse trees and predicted the most plausible function label of constituent from these features. Noting that parsing errors are difficult or even impossible to recover at function tag recognition stage, the alternative approaches are obtained by assigning function tags at the same time as producing parse trees (Merlo and Musillo, 2005), through learning deeper syntactic properties such as finer-grained labels, features from the nodes to the left of the current node.

Through all that research, however, successfully addressing function labeling requires accurate parsing model and training data, and the re-

[1] released by Linguistic Data Consortium (LDC) catalog NO. LDC2005T01

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 54–62,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

sults of them show that the performance ceiling of function labeling is limited by the parsers they used. Given the imperfection of existing automatic parsers, which are far from producing gold-standard results, function tags output by such models cannot be satisfactory for practical use. The limitation is even more pertinent for the languages that do not have sophisticated parsing resources, or languages that have inherent linguistic challenges for parsing (like Chinese). It is therefore worthwhile to investigate alternatives to function labeling for languages under the parsing bottleneck, both in terms of features used and effective learning algorithms.

In current study, we focused on the use of parser-independent features for function labeling. Specifically, our proposal is to classify function types directly from lexical features like words and their POS tags and the surface sentence information like the word position. The hypothesis that underlies our proposal is that lexical features are informative for different function types, and capture fundamental properties of the semantics that sometimes can not be concluded from the glance of parse structure. Such cases come when distinguishing phrases of the same structure that differ by just one word – for instance, telling "在上海 (in Shanghai)", which is locative, from "在五月 (in May)", which is temporal.

At a high level, we can say that class-based differences in function labels are reflected in statistics over the lexical features in large-scale annotated corpus, and that such knowledge can be encoded by learning algorithms. By exploiting lexical information collected from Penn Chinese Treebank (CTB) (Xue et al., 2000), we investigate a supervised sequence learning model to test our core hypothesis – that function tags could be guessed precisely through informative lexical features and effective learning methods. At the end of this paper, we extend previous function labeling methods from English to Chinese. The result proves, at least for Chinese language, our proposed method outperforms previous ones that utilize sophisticated parse trees.

In section 2 we will introduce the CTB resources and function tags used in our study. In section 3, we will describe the sequence learning algorithm in the framework of maximum margin learning, showing how to approximate function tagging by simple lexical statistics. Section 4 gives a detailed discussion of our experiment and comparison with pieces of related work. Some final remarks will be given in Section 5.

Table 1: Complete set of function labels in Chinese Treebank and function labels used in our system (selected labels).

type	labels in CTB		selected labels
clause types	IMP	imperative	
	Q	question	
(function/form) discrepancies	ADV	adverbial	√
grammatical roles	EXT	extent	√
	FOC	focus	√
	IO	indirect object	√
	OBJ	direct object	√
	PRD	predicate	√
	SBJ	subject	√
	TPC	topic	√
adverbials	BNF	beneficiary	√
	CND	condition	√
	DIR	direction	√
	IJ	interjective	√
	LGS	logic subject	√
	LOC	locative	√
	MNR	manner	√
	PRP	purpose/reason	√
	TMP	temporal	√
	VOC	vocative	√
miscellaneous	APP	appositive	
	HLN	headline	
	PN	proper names	
	SHORT	short form	
	TTL	title	
	WH	wh-phrase	

2 Chinese Function Tags

The label such as subject, object, time, location, etc. are named as function tags[2] in Penn Chinese Treebank (Xue et al., 2000), a complete list of which is shown in Table 1. Among the 5 categories, grammatical roles such as SBJ, OBJ are useful in recovering predicate-argument structure, while adverbials are actually semantically oriented labels (though not true for all cases, see (Merlo and Palmer, 2006)) that carry semantic role information.

As for the task of function parsing, it is reasonable to ignore the IMP and Q in Table 1 since they do not form natural syntactic or semantic classes. In addition, we regard the miscellaneous labels as an "O" label (out of any function chunks) like labeling constituents that do not bear any function

[2] The annotation guidelines of Penn Chinese Treebank talk of function tags. We will use the term function labels and function tags identically, and hence make no distinction between function labeling and function tagging throughout this paper. Also, the term function chunk signifies a sequence of words that are decorated with the same function label.

tags. Punctuation marks like comma, semi-colon and period that separate sentences are also denoted as “O”. But the punctuation that appear within one sentence like double quotes are denoted with the same function labels with the content they quote.

In the annotation guidelines of CTB (Xue et al., 2000), the function tag “PRD” is assigned to non-verbal predicate. Since VP (verb phrase) is always predicate, “PRD” is assumed and no function tag is attached to it. We make a slight modification to such standard by calling this kind of VP “verbal predicates”, and assigning them with function label “TAR (target verb)”, which is grouped into the same grammar roles type with “PRD”.

To a large extent, PP (preposition phrase) always plays a functional role in sentence, like “PP-MNR” in Figure 1. But there are many such PPs bare of any function type in CTB resources. Like in the sentence “比去年同期增长 25% (increase by 25% over the same period of last year)”, “比去年同期 (over the same period of last year)” is labeled as “PP” in CTB without any function labels attached, thus losing to describe the relationship with the predicate “增长 (increases)”. In order to capture various relationships related to the predicate, we assign function label “ADT (adjunct)” for this scenario, and merge it with other adverbials to form adverbials category. There are 1,415 such cases in CTB resources, which account for a large proportion of adverbials types.

After the modifications discussed above, in our final system we use 20 function labels[3] (18 original CTB labels shown in Table 2 and two newly added labels) that are grouped into two types: grammatical roles and adverbials.

We calculate the frequency (the number of times each tag occurs) and average length (the average number of words each tag covers) of each function category in our selected sentences, which are listed in Table 2. As can be seen, the frequency of adverbials is much smaller than that of grammatical roles. Furthermore, the average length of most adverbials are somewhat larger than 4. Such data distribution is likely to be one cause of the lower identification accuracy of adverbials as we will see in the experiments.

From the layer of function labeling, sentences in CTB are described with the structure of “SV” which indicates a sentence is basically composed of “subject + verb”. But in order to identify objects and complements of predicates, we express sentence by “SVO” framework in our system, which regards sentence as a structure of “subject + verb + object”. The structure transformation is obtained through a preprocessing procedure, by upgrading OBJs and complements (EXT, DIR, etc.) which are under VP in layered brackets.

Table 2: Categories of function tags with their relative frequencies and average length.

Function Labels	Frequency	Average Length
grammatical roles	99507	2.62
FOC	133	1.89
IO	126	1.26
OBJ	25834	4.15
PRD	4428	5.20
SBJ	23809	3.02
TPC	676	3.51
TAR	44501	1.25
adverbials	33287	2.11
ADT	1415	4.51
ADV	21891	1.32
BNF	465	4.66
CND	68	3.15
DIR	1558	4.68
EXT	1048	1.99
IJ	1	1.00
LGS	204	5.42
LOC	2051	4.27
MNR	1053	4.48
PRP	224	4.91
TMP	3309	2.25

3 Learning Function Labels

Function labeling deals with the problem of predicting a sequence of function tags $y = y_1, ..., y_T$, from a given sequence of input words $x = x_1, ..., x_T$, where $y_i \in \Sigma$. Therefore the function labeling task can be formulated as a stream of sequence learning problem. The general approach is to learn a w-parameterized mapping function $F : X \times Y \rightarrow \Re$ based on training sample of input-output pairs and to maximize $F(x, y; w)$ over the response variable to make a prediction.

There has been several algorithms for labeling sequence data including hidden Markov model (Rabiner, 1989), maximum entropy Markov model (Mccallum et al., 2000), conditional random fields (Lafferty et al., 2001) and hidden Markov support vector machine (HM-SVM) (Altun et al., 2003; Tsochantaridis et al., 2004), among which HM-SVM shows notable advantages by its learning

[3] ADV includes ADV and ADVP in CTB recourses, grouped into adverbials. In function labeling level, EXT that signifies degree, amount of the predicates should be grouped into adverbials like in the work of (Blaheta and Charniak, 2000) and (Merlo and Musillo, 2005).

non-linear discriminant functions via kernel function, the properties inherited from support vector machines (SVMs). Furthermore, HM-SVM retains some of the key advantages of Markov model, namely the Markov chain dependency structure between labels and an efficient dynamic programming formulation.

In this paper we investigate the application of the HM-SVM model to Chinese function labeling task. In order to keep the completeness of paper, we here address briefly the HM-SVM algorithm, more details of which could be founded in (Altun et al., 2003; Tsochantaridis et al., 2004), then we will concentrate on the techniques of applying it to our specific task.

3.1 Learning Model

The framework from which HM-SVM are derived is a maximum margin formulation for joint feature functions in kernel learning setting. Given n labeled examples $(x^1, y^1), ..., (x^n, y^n)$, the notion of a separation margin proposed in standard SVMs is generalized by defining the margin of a training example with respect to a discriminant function $F(x, y; w)$, as:

$$\gamma_i = F(x^i, y^i; w) - \max_{y \notin y^i} F(x^i, y; w). \quad (1)$$

Then the maximum margin problem can be defined as finding a weight vector w that maximizes $min_i \gamma_i$. By fixing the functional margin ($max_i \gamma_i \geq 1$) like in the standard setting of SVMs with binary labels, we get the following hard-margin optimization problem with a quadratic objective:

$$\min_w \frac{1}{2} ||w||^2, \quad (2)$$

with constraints,

$$F(x^i, y^i; w) - F(x^i, y; w) \geq 1, \forall_{i=1}^{n}, \forall_{y \neq y^i}.$$

In the particular setting of SVM, F is assumed to be linear in some combined feature representation of inputs and outputs $\Phi(x, y)$, i.e. $F(x, y; w) = \langle w, \Phi(x, y) \rangle$. $\Phi(x, y)$ can be specified by extracting features from an observation/label sequence pair (x, y). Inspired by HMMs, we propose to define two types of features, interactions between neighboring labels along the chain as well as interactions between attributes of the observation vectors and a specific label. For instance, in our function labeling task, we might think of a label-label feature of the form

$$\alpha(y_{t-1}, y_t) = [[y_{t-1} = \text{SBJ} \wedge y_t = \text{TAR}]], \quad (3)$$

that equals 1 if a SBJ is followed by a TAR. Analogously, a label-observation feature may be

$$\beta(x_t, y_t) = [[y_t = \text{SBJ} \wedge x_t \text{ is a noun}]], \quad (4)$$

which equals 1 if x at position t is a noun and labeled as SBJ. The described feature map exhibits a first-order Markov property and as a result, decoding can be performed by a Viterbi algorithm in $O(T|\Sigma|^2)$.

All the features extracted at location t are simply stacked together to form $\Phi(x, y; t)$. Finally, this feature map is extended to sequences (x, y) of length T in an additive manner as

$$\Phi(x, y) = \sum_{t=1}^{T} \Phi(x, y; t). \quad (5)$$

3.2 Features

It deserves to note that features in HM-SVM model can be easily changeable regardless of dependency among them. In this prospect, features are very far from independent can be cooperated in the model.

By observing the particular property of function structure in Chinese sentences, we design several sets of label-observation features which are independent of parse trees, namely:

Words and POS tags: The lexical context is extremely important in function labeling, as indicated by their importance in related task of phrase chunking. Due to long-distance dependency of function structure, intuitively, more wider context window will bring more accurate prediction. However, the wider context window is more likely to bring sparseness problem of features and increase computation cost. So there should be a proper compromise among them. In our experiment, we start from a context of [-2, +2] and then expand it to [-4, 4], that is, four words (and POS tags) around the word in question, which is closest to the average length of most function types shown in Table 2.

Bi-gram of POS tags: Apart from POS tags themselves, we also try on the bi-gram of POS tags. We regard POS tag sequence as an analog to function chains, which reveals somewhat the dependent relations among words.

Verbs: Function labels like subject and object specify the relations between verb and its arguments. As observed in English verbs (Levin, 1993), each class of verb is associated with a set of syntactic frames. Similar criteria can also be found in Chinese. In this sense, we can rely on the surface verb for distinguishing argument roles syntactically. Besides the verbs themselves, we also take into account the special words sharing common property with verbs in Chinese language, which are active voice “把(BA)” and passive voice “被(BEI)”. The verb we refer here is supposed to be the last verb if it happens in a consecutive verb sequence, thus actually not the head verb of sentence.

POS tags of verbs: according to CTB annotation guideline, verbs are labeled with four kinds of POS tags (VA, VC, VE, VV), along with BA (for “把”), LB and SB (for “被”). This feature somewhat notifies the coarse class of verbs talked in (Levin, 1993) and is taken into account as feature candidates.

Position indicators: It is interesting to notice that whether the constituent to be labeled occurs before or after the verb is highly correlated with grammatical function, since subjects will generally appear before a verb, and objects after, at least for Chinese language. This feature may overcome the lack of syntactic structure that could be read from the parse tree.

In our experiment, all feature candidates are introduced to the training instances incrementally by a feature inducing procedure, then we use a gain-driven method to decide whether a feature should be reserved or deleted according to the increase or decrease of the predication accuracy. The procedure are described in Figure 2.

Figure 2: Pseudo-code of feature introducing procedure.

```
initialize feature superset $C$={all feature candidates}, feature set $c$ is empty
**repeat**
  **for** each feature $c_i \in C$ **do**
    construct training instances using $c_i \cup c$
    experiment on k-fold cross-validation data
    **if** accuracy increases **then**
      $c_i \rightarrow c$
    **end if**
  **end for**
**until** all features in $C$ are traversed
```

4 Experiment and Discussion

In this section, we turn to our computational experiments that investigate whether the statistical indicators of lexical properties that we have developed can in fact be used to classify function labels, and demonstrate which kind of feature contributes most in identifying function types, at least for Chinese text.

As in the work of (Ramshaw and Marcus, 1995), each word or punctuation mark within a sentence is labeled with “IOB” tag together with its function type. The three tags are sufficient for encoding all constituents since there are no overlaps among different function chunks. The function tags in this paper are limited to 20 types, resulting in a total of $|\Sigma| = 41$ different outputs.

We use three measures to evaluate the model performance: *precision*, which is the percentage of detected chunks that are correct; *recall*, which is the percentage of chunks in the data that are found by the tagger; and *F-score* which is equal to $2 \times precision \times recall/(precision + recall)$. Under the “IOB” tagging scheme, a function chunk is only counted as correct when its boundaries and its type are both identified correctly. Furthermore, sentence accuracy is used in order to observe the prediction correctness of sentences, which is defined as the percentage of sentences within which all the constituents are assigned with correct tags. As in the work of (Blaheta and Charniak, 2000) and (Merlo and Musillo, 2005), to avoid calculating excessively optimistic values, constituents bearing the “O” label are not counted in for computing overall precision, recall and F-score.

We derived 18,782 sentences from CTB 5.0 with about 497 thousands of words (including punctuation marks). On average, each sentence contains 26.5 words with 2.4 verbs. We followed 5-fold cross-validation method in our experiment. The numbers reported are the averages of the results across the five test sets.

4.1 Evaluation of Different Features and Models

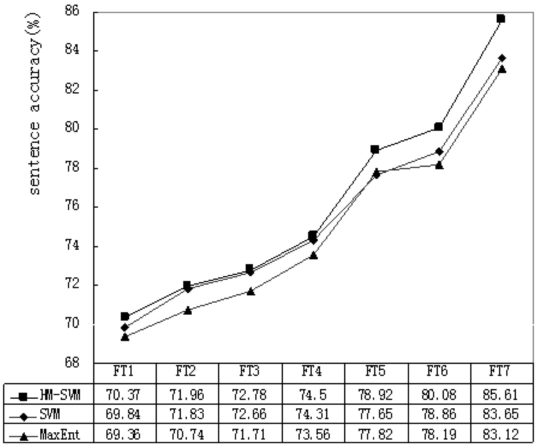
In pilot experiments on a subset of the features, we provide a comparison of HM-SVM with other two learning models, maximum entropy (MaxEnt) model (Berger et al., 1996) and SVM model (Kudo, 2001), to test the effectiveness of HM-SVM on function labeling task, as well as the generality of our hypothesis on different learning

Table 3: Features used in each experiment round.

FT1	word & POS tags within [-2,+2]
FT2	word & POS tags within [-3,+3]
FT3	word & POS tags within [-4,+4]
FT4	FT3 plus POS bigrams within [-4,+4]
FT5	FT4 plus verbs
FT6	FT5 plus POS tags of verbs
FT7	FT6 plus position indicators

models.

In our experiment, SVMs and HM-SVM training are carried out with SVM^{struct} packages[4]. The multi-class SVMs model is realized by extending binary SVMs using *pairwise* strategy. We used a first-order of transition and emission dependency in HM-SVM. Both SVMs and HM-SVM are trained with the linear kernel function and the soft margin parameter c is set to be 1. The MaxEnt model is implemented based on Zhang's MaxEnt toolkit[5] and L-BFGS (Nocedal, 1999) method to perform parameter estimation.



	FT1	FT2	FT3	FT4	FT5	FT6	FT7
■ HM-SVM	70.37	71.96	72.78	74.5	78.92	80.08	85.61
◆ SVM	69.84	71.83	72.66	74.31	77.65	78.86	83.65
▲ MaxEnt	69.36	70.74	71.71	73.56	77.82	78.19	83.12

Figure 3: Sentence accuracy achieved by different models using different feature combinations.

We use sentence accuracy to compare performances of three models with different feature combinations shown in Table 3. The learning curves in Figure 3 illustrate feature combination FT7 gains the best results for all three models we considered. As we have expected, the performance improves as the context window expanded from 2 to 4 (from FT1 to FT3 in Figure 3). The sentence accuracy increases significantly when the features include verbs and position indicators, giving some indication of the complexity of the structure intervening between focus word and the verb. However, at a high level, we can simply say that any further information would help for identifying function types, so we believe that the features we deliberated on currently are by no means the solely optimal feature set.

As observed in Figure 3, the structural sequence model HM-SVM outperforms multi-class SVMs, meanwhile, they both perform slightly better than MaxEnt model, demonstrating the benefit of maximum margin based approach. In the experiment below, we will use feature FT7 and HM-SVM model to illustrate our method.

4.2 Results with Gold-standard POS Tags

By using gold-standard POS tags, this experiment is to view the performance of two types of function labels - grammatical roles and adverbials, and fine-grained function types belonging to them. We cite the average precision, recall and F-score of 5-fold cross validation data output by HM-SVM model to discuss this facet.

Table 4: Average performance for individual categories, using HM-SVM model with feature FT7 and gold-standard POS tags.

	Precision	Recall	F-score
Overall	0.934	0.942	0.938
grammatical roles	0.949	0.960	0.955
FOC	0.385	0.185	0.250
IO	0.857	0.286	0.429
OBJ	0.960	0.980	0.970
PRD	0.985	0.988	0.987
SBJ	0.869	0.912	0.890
TPC	0.292	0.051	0.087
TAR	0.986	0.990	0.990
adverbials	0.887	0.887	0.887
ADT	0.690	0.663	0.676
ADV	0.956	0.955	0.956
BNF	0.729	0.869	0.793
CND	0.000	0.000	0.000
DIR	0.741	0.812	0.775
EXT	0.899	0.820	0.857
LGS	0.563	0.659	0.607
LOC	0.712	0.721	0.716
MNR	0.736	0.783	0.759
PRP	0.656	0.404	0.500
TMP	0.821	0.808	0.814

Table 4 details the results of individual function types. On the whole, grammatical roles outperform adverbials. It seems to reflect the fact that

[4] http://svmlight.joachims.org/s vm_multiclass.html

[5] http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html

syntactic constituents can often be guessed based on POS tags and high-frequency lexical words, largely avoiding sparse-data problems. This is evident particularly for "OBJ" that reaches aggressively 0.970 in F-score. One exception is "TPC", whose precision and recall draws to the lowest among grammatical roles. In CTB resources, "TPC" marks elements that appear before the subject in a declarative sentence, and, it always constitutes a noun phrase together with the subject of the sentence. As an illustrating example, in the sentence "天津与台湾产业结果相似 (The industrial structure of Tianjin and Taiwan is similar)", "天津与台湾 (Tianjin and Taiwan)" is labeled with "TPC", while "产业结构 (The industrial structure)" with "SBJ". In such settings, it is difficult to distinguish between them even for human beings.

Overall, there are three possible explanations for the lower F-score of adverbials. One is that tags characterized by much more semantic information always have flexible syntactic constructions and diverse positions in sentence, which makes it difficult to capture their uniform characteristics. Second one is likely that the long-distance dependency and sparseness problem degrade the performance of adverbials greatly. This can be viewed from the statistics in Table 2, where most of the adverbials are longer than 4, while the frequency of them is significantly lower than that of grammatical roles. The third possible explanation is that there is vagueness among different adverbials. An instance to state such case is the dispute between "ADV" and "MNR" like the phrase "随着改革开放的深入 (with the deepening of reform and opening-up)", which are assigned with "ADV" and "MNR" in two totally the same contexts in our training data. Noting that word sequences for some semantic labels carry several limited formations (e.g., most of "DIR" is preposition phrase beginning with "from, to"), we will try some linguistically informed heuristics to detect such patterns in future work.

4.3 Results with Automatically Assigned POS Tags

Parallel to experiments on text with gold-standard POS tags, we also present results on automatically POS-tagged text to quantify the effect of POS accuracy on the system performance. We adopt automatic POS tagger of (Qin et al., 2008), which got the first place in the forth SIGHAN Chinese POS tagging bakeoff on CTB open test, to assign POS tags for our data. Following the approach of (Qin et al., 2008), we train the automatic POS tagger which gets an average accuracy of 96.18% in our 5-fold cross-validation data. Function tagger takes raw text as input, then completes POS tagging and function labeling in a cascaded way. As shown in Table 5, the F-score of AutoPOS is slightly lower than that of GoldPOS. However, the small gap is still within our first expectation.

Table 5: Performance separated for grammatical roles and adverbials, of our models GoldPOS (using gold-standard POS tags), GoldPARSE (using gold-standard parse trees), AutoPOS (using automatically labeled POS tags).

	grammatical roles			adverbials		
	P	R	F	P	R	F
GoldPOS	0.949	0.960	**0.955**	0.887	0.887	0.887
AutoPOS	0.921	0.948	0.934	0.872	0.867	0.869
GoldPARSE	0.936	0.967	0.951	0.911	0.884	**0.897**

4.4 Results with Gold-standard Parser

A thoroughly different way for function labeling is deriving function labels together with parsing. The work of (Blaheta and Charniak, 2000; Blaheta, 2004; Merlo and Musillo, 2005) has approved its effectiveness in English text. Among them, the work of Merlo and Musillo (Merlo and Musillo, 2005) achieved a state-of-the-art F1 score for English function labeling (0.964 for grammatical roles and 0.863 for adverbials). In order to address the question of whether such method can be successfully applied to Chinese text and whether the simple method we proposed is better than or at least equivalent to it, we used features collected from hand-crafted parse trees in CTB resources, and did a separate experiment on the same text. The features we used are borrowed from feature trees described in (Blaheta and Charniak, 2000). A trivial difference is that in our system the head for prepositional phrases is defined as the prepositions themselves (not the head of object of prepositional phrases (Blaheta and Charniak, 2000)), because we think that the preposition itself is a more distinctive attribute for different semantic meanings.

Results in Table 5 show that the parser tree doesn't help a lot in Chinese function labeling. One reason for this may be sparseness problem of parse tree features – For instance, in one of the 5-

fold data, 34% of syntactic paths in test instances are unseen in training data. For sentences with the average length of more than 40 words, this sparseness becomes even severe. Another possible reason is that some functional chunks are more local and less prone to structured parse trees, as observed in examples listed at the beginning of the paper. In Table 5, although the performance of adverbials grows really huge when using features from the gold-standard parse trees, the performance of grammatical roles drops as introducing such features. As mentioned above, in fact even the simple position feature can give a better explanation to word's grammatical role than complicated syntactic path.

Although the experimental setup is strictly not the same for the present paper and (Blaheta and Charniak, 2000; Blaheta, 2004; Merlo and Musillo, 2005), we observe that the proposed method yields better results with deliberately designed but simple features at lexical level, while attempts in (Blaheta and Charniak, 2000; Blaheta, 2004; Merlo and Musillo, 2005) optimized function labeling together with parsing, which is a more complex task and difficult to realize for languages that lack sufficient parse resources.

The work of (Blaheta and Charniak, 2000; Blaheta, 2004; Merlo and Musillo, 2005) reveal that the performance of parser used sets upper bound on the performance of function labeling. However, the best Chinese parser ever reported (Wang et al., 2006) achieves 0.882 F-score for sentences with less than 40 words, we therefore conclude that the way using auto-parser for Chinese function labeling is not the optimal choice.

4.5 Error Analysis

In the course of our experiment, we wanted to attain some understanding of what sort of errors the system was making. While still working on the gold-standard POS-tagged text, we randomly took one output from the 5-fold cross-validation tests and examined each error. But when observing the 1,550 wrongly labeled function chunks (26,593 in total), we can distinguish three types of errors.

The first and widest category of errors are caused when the lexical construction of the chunk is similar to other chunk types. A typical example is "PRP (purpose)" and "BNF (beneficiary)", both of which are mostly prepositional phrases beginning with "为, 为了(for, in order to)".

The second type of errors are found when the chunk is too long, like more than 8 words. Normally it is not easy to eliminate this kind of errors through local lexical features. In Chinese, the long chunks are mainly composed of "的 (DE)" structure that can be translated into attributive clause in English. The "的 (DE)" structures are usually nested component and used as a modifier of noun phrases, thus this kind of errors can be partly resolved by accurately recognition of such structure.

The third type of errors concern the sentence with some special structure, like intransitive sentence, elliptical sentence (left out of subject or object), and so on. The errors of "IO" with wrong tag "OBJ", and errors of "EXT" with wrong tag "OBJ" fall into the third categories. It is interesting to notice that, when using GoldPARSE (see Table 5), suggesting that features from the trees are helpful when disambiguating function labels that related with sentence structures.

5 Conclusion and Future Work

We have presented the first experimental results on Chinese function labeling using Chinese Treebank resources, and shown that Chinese function labeling can be reached with considerable accuracy given a small number of lexical features. Even though our experiments using hand-crafted parse trees yield promising initial results, this method will be hampered when using fully automatic parser due to the imperfection of Chinese parser, which is our core motivation to assign function labels by exploiting the underlining lexical insights instead of parse trees. Experimental results suggest that our method for Chinese function labeling is comparable with the English state-of-the-art work that utilizes complicated parse trees.

We believe that we have not settled on an "optimal" set of features for Chinese function labeling, hence, more language-specific customization is necessary in the future work. Although there have been speculations and trails on things that function labels might help with, it remains to be important to discover how function labels contribute to other NLP applications, such as the Japanese-Chinese machine translation system we have been working on.

References

Altun, Y., Tsochantaridis, I., Hofmann, T. 2003. Hidden Markov Support Vector Machines. In: *Pro-*

ceedings of ICML 2003, pages 172-188, Washington, DC, USA.

Berger, A., Pietra, D. S., Pietra, D. V. 1996. A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics, 22(1):39-71.

Blaheta, D. 2004. Function Tagging. Ph.D. thesis, Department of Computer Science, Brown University.

Blaheta, D., Charniak, E. 2000. Assigning Function Tags to Parsed Text. In: *Proceedings of the 1st NAACL*, pages 234-240, Seattle, Washington.

Chrupala, G., Stroppa, N., Genabith, J., Dinu, G. 2007. Better Training for Function Labeling. In: *Proceedings of RANLP2007*, Borovets, Bulgaria.

Gildea, D., Palmer, M. 2002. The Necessity of Parsing for Predicate Argument Recognition. In: *Proceedings of the 40th ACL*, pages 239-246, Philadelphia, USA.

Iida, R., Komachi, M., Inui, K., Matsumoto, Y. 2007. Annotating a Japanese Text Corpus with Predicate-argument and Coreference Relations. In: *Proceedings of ACL workshop on the linguistic annotation*, pages 132-139, Prague, Czech Republic.

Jijkoun, V., Rijke D. M. 2004. Enriching the Output of a Parser Using Memory-based Learning. In: *Proceedings of the 42nd ACL*, pages 311-318, Barcelona, Spain.

Kiss, T., Strunk, J. 2006. Unsupervised Multilingual Sentence Boundary Detection. *Computational Linguistics*, 32(4):485-525.

Kudo, T., Matsumoto, Y. 2001. Chunking with Support Vector Machines. In: *Proceedings of the NAACL 2001*, pages 1-8, Pittsburgh, USA.

Nocedal, J., Wright, S. J. 1999. Numerical Optimization. Springer.

Lafferty, J., McCallum, A., Pereira, F. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In: *Proceedings of ICML 2001*, pages 282-289, Williamstown, USA.

Levin, B. 1993. *English Verb Classes and Alternations: A preliminary Investigation.* The University of Chicago Press, USA.

Marcus, M., Kim, G., Marcinkiewicz, A. M., Macintyre, R., Bies, A., Ferguson, M., Katz, K., Schasberger, B. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In: *Proceedings of ARPA Human Language Technology Workshop*, San Francisco, USA.

Mccallum, A., Freitag, D., Pereira, F. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation. In: *Proceedings of ICML 2000*, pages 591-598, Stanford University, USA.

Merlo, P., Ferrer, E. E. 2006. The Notion of Argument in Prepositional Phrase Attachment. *Computational Linguistics*, 32(3):341-378.

Merlo, P., Musillo, G. 2005. Accurate Function Parsing. In: *Proceedings of EMNLP 2005*, pages 620-627, Vancouver, Canada.

Qin, Y., Yuan, C., Sun, J., Wang, X. 2008. BUPT Systems in the SIGHAN Bakeoff 2007. In: *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing*, pages 94-97, Hyderabad, India.

Rabiner, L. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In: *Proceedings of the IEEE*, 77(2):257-286.

Ramshaw, L., Marcus, M. 1995. Text Chunking Using Transformation Based Learning. In: *Proceedings of ACL Third Workshop on Very Large Corpora*, pages 82-94, Cambridge MA, USA.

Swier, R., Stevenson, S. 2004. Unsupervised Semantic Role Labelling. In: *Proceedings of EMNLP-2004*, pages 95-102, Barcelona, Spain.

Tsochantaridis, T., Hofmann, T., Joachims, T., Altun, Y. 2004. Support Vector Machine Learning for Interdependent and Structured Output Spaces. In: *Proceedings of ICML 2004*, pages 823-830, Banff, Canada.

Wang, M., Sagae, K., Mitamura, T. 2006. A Fast, Accurate Deterministic Parser for Chinese. In: *Proceedings of the 44th ACL*, pages 425-432, Sydney, Australia.

Xue, N., Xia, F., Huang, S., Kroch, T. 2000. The Bracketing Guidelines for the Chinese Treebank. *IRCS Tech., rep., University of Pennsylvania.*

Zhao, Y., Zhou, Q. 2006. A SVM-based Model for Chinese Functional Chunk Parsing. In: *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 94-10, Sydney, Australia1.

Zhou, Q., Zhan, W., Ren, H. 2001. Building a Large-scale Chinese Chunkbank (in Chinese). In: *Proceedings of the 6th Joint Conference of Computational Linguistics of China*, Taiyuan, China.

Optimizing Language Model Information Retrieval System with Expectation Maximization Algorithm

Justin Liang-Te Chiu
Department of Computer Science
and Information Engineering,
National Taiwan University
#1 Roosevelt Rd. Sec. 4, Taipei,
Taiwan 106, ROC
b94902009@ntu.edu.tw

Jyun-Wei Huang
Department of Computer Science
and Engineering,
Yuan Ze University
#135 Yuan-Tung Road, Chungli,
Taoyuan,Taiwan,ROC
s976017@mail.yzu.edu.tw

Abstract

Statistical language modeling (SLM) has been used in many different domains for decades and has also been applied to information retrieval (IR) recently. Documents retrieved using this approach are ranked according their probability of generating the given query. In this paper, we present a novel approach that employs the generalized Expectation Maximization (EM) algorithm to improve language models by representing their parameters as observation probabilities of Hidden Markov Models (HMM). In the experiments, we demonstrate that our method outperforms standard SLM-based and tf.idf-based methods on TREC 2005 HARD Track data.

1 Introduction

In 1945, soon after the computer was invented, Vannevar Bush wrote a famous article---"As we may think" (V. Bush, 1996), which formed the basis of research into Information Retrieval (IR). The pioneers in IR developed two models for ranking: the vector space model (G. Salton and M. J. McGill, 1986) and the probabilistic model (S. E. Robertson and S. Jones, 1976). Since then, the research of classical probabilistic models of relevance has been widely studied. For example, Robertson (S. E. Robertson and S. Walker, 1994; S. E. Robertson, 1977) modeled word occurrences into relevant or non-relevant classes, and ranked documents according to the probabilities they belong to the relevant one. In 1998, Ponte and Croft (1998) proposed a language modeling framework which opens a new point of view in IR. In this approach, they gave up the model of relevance; instead, they treated query generation as random sampling from every document model. The retrieval results were based on the probabilities that a document can generate the query string. Several improvements were proposed after their work. Song and Croft (1999), for example, was the first to bring up a model with bi-grams and Good Turing re-estimation to smooth the document models. Latter, Miller et al. (1999) used Hidden Markov Model (HMM) for ranking, which also included the use of bigrams.

HMM, firstly introduced by Rabiner and Juain (1986) in 1986, has been successfully applied into many domains, such as named entity recognition (D. M. Bikel et al., 1997), topic classification (R. Schwartz et al., 1997), or speech recognition (J. Makhoul and R. Schwartz, 1995). In practice, the model requires solving three basic problems. Given the parameters of the model, computing the probability of a particular output sequence is the first problem. This process is often referred to as decoding. Both Forward and Backward procedure are solutions for this problem. The second problem is finding the most possible state sequence with the parameters of the model and a particular output sequence. This is usually completed with Viterbi algorithm. The third problem is the learning problem of HMM models. It is often solved by Baum-Welch algorithm (L. E. Bmjm et al., 1970). Given training data, the algorithm computes the maximum likelihood estimates and posterior mode estimate. It is in essence a generalized Expectation Maximization (EM) algorithm which was first explained and given name by Dempster, Laird and Rubin (1977) in 1977. EM can estimate the maximum likelihood of parameters in probabilistic models which has unseen variables. Nonetheless, in our knowledge, the EM procedure in HMM has never been used in IR domain.

In this paper, we proposed a new language model approach which models the user query and documents as HMM models. We then used EM algorithm to maximize the probability of query words in our model. Our assumption is that if the word's probability in a document is maximized, we can estimate the probability of generating the query word from documents more confidently. Because they not only been calculated by language modeling view features, but also been maximized with statistical methods. Therefore the imprecise cases caused by special distribution in language modeling approach can be further prevented in this way.

The remainders of this paper are organized as follows. We review two related works in Section 2. In Section 3, we introduce our EM IR approach. Section 4 compares our results to two other approaches proposed by Song and Corft (1999) and Robertson (1995) based on the data from TREC HARD track (J. Allan, 2005). Section 5 discusses the effectiveness of our EM training and the EM-based document weighting we proposed. Finally, we conclude our paper in Section 6 and provide some future directions at Section 7.

2 Related Works

Even if we only focus on the probabilistic approach to IR, it is still impossible to discuss all up-to-date research. Instead we focus on two previous works which have inspired the work reported in this paper: the first is a general language model approach proposed by Song and Croft (1999) and the second is a HMM approach by Miller et al. (1999).

2.1 A General Language Model for IR

In 1999, Song and Croft (1999) introduced a language model based on a range of data smoothing technique. The following are some of the features they used:

Good-Turing estimate: Since the effect of Good-Turing estimate was verified as one of the best discount methods (C. D. Manning and H. Schutze, 1999), Song and Croft used Good-Turing estimate for allocating proper probability for the missing terms in the documents. The smoothed probability for term t in document d can be obtained with the following formula:

$$P_{GT}(t|d) = \frac{(tf+1)S(N_{tf+1})}{S(N_{tf})N_d}$$

where N_{tf} is the number of terms with frequency tf in a document. N_d is the total number of terms occurred in document d, and a powerful smoothing function $S(N_{tf})$, which is used for calculating the expected value of N_{tf} regardless of the N_{tf} appears in the corpus or not.

Expanding document model: The document model can be viewed as a smaller part of whole corpus. Due to its limited size, there is a large number of missing terms in documents, and can lead to incorrect distributions of known terms. For dealing with the problem, documents can be expanded with the following weighted sum/product approach:

$$P_{sum}(t|d) = \omega \times P_{doc}(t|d) + (1-\omega) \times P_{corpus}(t)$$
$$P_{product}(t|d) = P_{doc}(t|d)^{\omega} \times P_{corpus}(t)^{(1-\omega)}$$

where ω is a weighting parameter between 0 and 1.

Modeling Query as a Sequence of Terms: Treating a query as a set of terms is commonly seen in IR researches. Song and Croft treated queries as a sequence of terms, and obtained the probability of generating the query by multiplying the individual term probabilities.

$$P_{sequence}(Q|d) = \prod_{i=1}^{m} P(t_i|d)$$

where t_1, t_2 …, t_m is the sequence of terms in a query Q.

Combining the Unigram Model with the Bigram Model: This is commonly implemented with interpolation in statistical language modeling:

$$P(t_{i-1}, t_i|d) = \lambda_1 \times P_1(t_i|d) + \lambda_2 \times P_2(t_{i-1}, t_i|d)$$

where λ_1 and λ_2 are two parameters, and $\lambda_1 + \lambda_2 = 1$. Such interpolation can be modeled by HMM, and can learn the appropriate value from the corpus through EM procedure. A similar procedure is described in Hiemstra and Vries (2000).

2.2 A HMM Information Retrieval System

Miller et al. demonstrated an IR system based on HMM. With a query *Q*, Miller et al. tried to rank the documents according to the probability that *D* is relevant (R) with it, which can be written as *P*(*D* is *R*|*Q*). With Baye's rule, the core formula of their approach is:

$$P(D \text{ is } R|Q) = \frac{P(Q|D \text{ is } R) \cdot P(D \text{ is } R)}{P(Q)}$$

where *P*(*Q*|*D* is *R*) is the probability of query *Q* being posed by a relevant document *D*; *P*(*D* is *R*) is the prior probability that *D* is relevant; *P*(*Q*) is the prior probability of *Q*. Because *P*(*Q*) will be identical, and the *P*(*D* is *R*) is assumed to be constant across all documents, they place their focus on *P*(*Q*|*D* is *R*).

To figure out the value of *P*(*Q*|*D* is *R*), they established a HMM. The union of all words appearing in the corpus is taken as the observation, and each different mechanism of query word generation represent a state. So the observation probability from different states is according to the output distribution of the state.

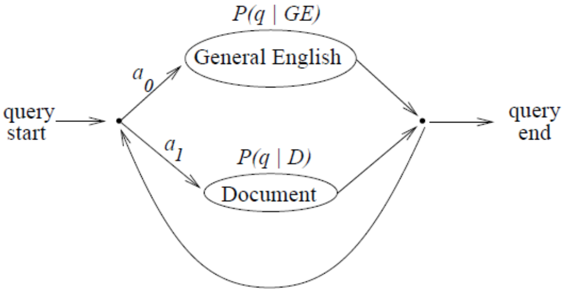


Figure 1. HMM proposed in "A Hidden Markov Model Information Retrieval System"

To estimate the transition and observation probabilities of HMM, EM algorithm is the standard method for parameter estimation. However, due to some difficulty, they make two practical simplifications. First, they assume the transition probabilities are same for all documents, since they establish an individual HMM for each document. Second, they completely abandon the EM algorithm for the estimation of observation probabilities. Instead, they use simple maximum likelihood estimates for each documents. So the probabilities which their HMM generate term q from their HMM states become:

$$\mathrm{P(q|D_k)} = \frac{\text{number of times q appears in } \mathrm{D_k}}{\text{length of } \mathrm{D_k}}$$

$$\mathrm{P(q|GE)} = \frac{\sum_k \text{number of times q appears in } \mathrm{D_k}}{\sum_k \text{length of } \mathrm{D_k}}$$

with these estimated parameters, they state the formula for P(Q|D is R) corresponding to Figure 1 as:

$$\mathrm{P(Q|D_k \text{ is } R)} = \prod_{q \in Q} (a_0 \mathrm{P(q|GE)} + a_1 \mathrm{P(q|D_k)})$$

the probabilities obtained through this formula is then used for calculating the P(D is R|Q). The document is then ranked according to the value of P(D is R|Q).

The HMM model we proposed is far different from Miller et al. (1999). They build HMM for every document, and treat all words in the document as one state's observation, and word that is unrelated to the document, but occurs commonly in natural language queries as another state's observation. Hence, their approach requires information about the words which appears commonly in natural language. The content of the provided information will also affect the IR result, hence it is unstable. We assume that every document is an individual state, and the probabilities of query words generated by this document as the observation probabilities. Our HMM model is built on the corpus we used and does not need further information. This will make our IR result fit on our corpus and not affected by outside information. It will be detailed introduced at Section 3.

3 Our EM IR approach

We formulate the IR problem as follows: given a query string and a set of documents, we rank the documents according to the probability of each document for generating the query terms. Since the EM procedure is very sensitive to the number of states, while a large number of states take much time for one run, we firstly apply a basic language modeling method to reduce our document set. This language modeling method will be detailed at Section 3.1. Based on the reduced document set, we then describe how to build our HMM model, and demonstrate how to obtain the special-designed observance sequence for our HMM training in Section 3.2 and 3.3, respectively. Finally, Section 3.4 introduces the evaluation mechanism to the probability of generating the query for each document.

3.1 The basic language modeling method for document reduction

Suppose we have a huge document set D, and a query Q, we firstly reduce the document set to obtain the document D_r. We require the reducing method can be efficiently computed, therefore two methods proposed by Song and Croft (1999) are consulted with some modifications: Good-Turing estimation and modeling query as a sequence of terms.

In our modified Good-Turing estimation, we gathered the number of terms to calculate the term frequency (*tf*) information in our document set. Table 1 shows the term distribution of the AQUAINT corpus which is used in the TREC 2005 HARD Track (J. Allan, 2005). The detail of the dataset is described in Section 4.1.

tf	N_{tf}	*tf*	N_{tf}
0	1,140,854,966,460	5	3,327,633
1	166,056,563	6	2,163,538
2	29,905,324	7	1,491,244
3	11,191,786	8	1,089,490
4	5,668,929	9	819,517

Table 1. Term distribution in AQUAINT corpus

In this table, N_{tf} is the number of terms with frequency *tf* in a document. The tf = 0 case in the table means the number of words not appear in a document. If the number of all word in our corpus is W, and the number of word in a document d is w_d, then for each document, the tf = 0 will add $W - w_d$. By listing all frequency in our document set, we adapt the formula defined in (Song and Croft, 1999) as follows:

$$P_{mGT}(t|d) = \frac{(tf+1)N_{tf+1}}{N_{tf}N_d}$$

In our formula, the N_d means the number of word tokens in the document d. Moreover, the smoothing function is replaced with accurate frequency information, N_{tf} and N_{tf+1}. Obviously, there could be two problems in our method: First, while in high frequency, there might be some missing N_{tf+1}, because not all frequency is continuously appear. Second, the N_{tf+1} for the highest *tf* is zero, this will lead to its P_{mGT} become zero. Therefore, we make an assumption to solve these problems: If the N_{tf+1} is missing, then its value is the same as N_{tf}. According to Table 1, we can find out that the difference between *tf* and *tf*+1 is decreasing when the *tf* becomes higher. So we assume the difference becomes zero when we faced the missing frequency at a high number. This assumption can help us ensure the completeness of our frequency distribution.

Aside from our Good-Turing estimation design, we also treat query as a sequence of terms. There are two reasons to make us made this decision. By doing so, we will be able to handle the duplicate terms in the query. Furthermore, it will enable us to model query phrase with local contexts. So our document score with this basic method can be calculated by multiplying $P_{mGT}(q|d)$ for every q in Q. We can obtain D_r with the top 50 scores in this scoring method.

3.2 HMM model for EM IR

Once we have the reduced document set D_r, we can start to establish our HMM model for EM IR. This HMM is designed to use the EM procedure to modify its parameters, and its original parameters are given by the basic language modeling approach calculation.

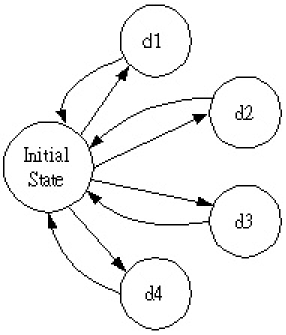


Figure 2. HMM model for EM IR

We define our HMM model as a four-tuple, $\{\mathbf{S},\mathbf{A},\mathbf{B},\boldsymbol{\pi}\}$, where $\mathbf{S}$ is a set of N states, $\mathbf{A}$ is a $N \times N$ matrix of state transition probabilities, $\mathbf{B}$ is a set of N probability functions, each describing the observation probability with respect to a state and $\boldsymbol{\pi}$ is the vector of the initial state probabilities.

In our HMM model, it composes of $|D_r|+1$ states. Every document in the document set is treated as an individual state in our HMM model. Aside from these document states, we add a special state called "Initial State". This state is the only one not associate with any document in our document sets. Figure 2 illustrates the proposed HMM IR model.

The transition probabilities in our HMM can be classified into two types. For the "Initial State", the transition to the other state can be regard as the probability of choosing that document. We assume that every document has the same probability to be chosen at the beginning, so the transition probabilities for "Initial State" are $1/|D_r|$ to every document state. For the document states, their transition probabilities are fixed: 100% to the “Initial State”. Since the transition between documents has no statistical meaning, we make the state transition after the document state back to the Initial State. This design helps us to keep the independency between the query words. We will detail this part at Section 3.3.

The observation probabilities for each state are similar with the concept of language modeling. There are three types of observations in our HMM model.

Firstly, for every document, we can obtain the observation probability for each query term according to our basic language modeling method. Even if the query term is not in the document, it will be assigned a small value according to the method described in Section 3.1.

Secondly, for the terms in a document, which is not part of our query terms, are treating as another observation. Since we mainly focus on the probability of generating the query terms from the documents, the rest terms are treated as the same type which means “not the query term”.

The last type of observation is a special imposed token “$” which has 100% observation probability at the Initial State.

Figure 3 shows a complete built HMM model for EM IR. The transition probability from Initial State is labeled with trans(d_n), and the observation probability in the document state and Initial State is showed with “ob”. The “N” symbol represents the “not the query term”. Summing all the token mentioned above, all possible observations for our HMM model are $|Q|+2$. The possible observation for each state is bolded, so we can see the difference between Initial State and Document State.

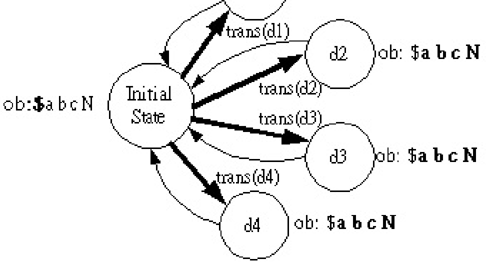


Figure 3. A complete built HMM model for EM IR with parameters

For Initial State, the observations are fixed with 100% for $ token. This special token help we ensure the independency between the query terms. The effect of this token will be discussed in Section 3.3. For the document states, the probabilities for the query terms are calculated with the simple language modeling approach. Even if the query term is not in the document, it will be assigned a small value according to the basic language modeling method. The rest of the terms in a document are treating as another kind of observation, which is the “N” symbol in the Figure 3. Since we mainly focus on the probability of generating the query terms from the documents, the rest of the words are treated as the same kind which means “not the query term”. Additionally, each document state represents a document, so the $ token will never been observed in them.

3.3 The observance sequence and HMM training procedure

After establishing the HMM model, the observation sequence is another necessary part for our HMM training procedure. The observation sequence used in HMM training means the trend for the observation while running HMM. In our approach, since we want to find out the document which is more related with our query, so we use the query terms as our observation sequence. During the state transition with query, we can maximize the probability for each document to generate our query. This will help us figure out which document is more related with our query.

Due to the state transitions in the proposed HMM model are required to go back to the Initial State after transiting to the document state, generating the pure query terms observation sequence is impossible, because the Initial State won’t produce any query term. Therefore, we add the $ token into our observation sequence before each query terms. For instance, if we are running a HMM training with query “a b c“, the exact observation sequence for our HMM training becomes “$ a $ b $ c”. Additionally, each document state represents a document, so the $ token will never been observed in them. By tuning our HMM model with the data from our query instead of other validation data, we can focus on the document we want more precisely.

The reason why we use this special setting for EM training procedure is because we are trying to maintain the independency assumption for query terms in HMM. The HMM observance sequence not only shows the trend of this model’s observation, but also indicate the dependency between these observations. However, the independency between all query terms is a common assumption for IR system (F. Song and W. B. Croft, 1999; V. Lavrenko and W. B. Croft, 2001; A. Berger and J. Lafferty, 1999). To ensure this assumption still works in our HMM system, we use the Initial State to separate each transition to the document state and observe the query terms. No matter the early or late the query term *t* occurs, the training procedure is fixed as "Starting from the Initial state and observed $, transit to a document state, and observe *t*". We've made experiments to verify the independency assumption still work, and the result remains the same no matter how we change the order of our query terms.

After constructing the HMM model and the observance sequence, we can start our EM training procedure. EM algorithm is used for finding maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved latent variables. In our experiment, we use EM algorithm to find the parameters of our HMM model. These parameters will be used for information retrieval. The detail implementation information can be found in (C. D. Manning and H. Schutze, 1999), which introduce HMM and the training procedure very well.

3.4 Scoring the documents with EM-trained HMM model

When the training procedure is completed, each document will have new parameters for the word's observation probability. Moreover, the transition probabilities from Initial State to the document state are no longer uniform due to the EM training. So the probability for a document *d* to generate the query *Q* becomes:

$$P(Q|d) = \mathrm{trans}(d) * \prod_{q \in Q} P(q|d)$$

In this formula, the trans(*d*) means the transition probability from the Initial State to the document state of *d*, which we called "EM-based document weighting". The *P*(*q*|*d*) means the observation probability for query term *q* in document state of *d*, which is also tuned in our EM training procedure. With this formula, we can rank the IR result according to this probability. This performs better than the GLM when the document size is relatively small, since GLM gives those documents as with too high score.

4 Experiment Results

4.1 Data Set

We use the AQUAINT corpus as our training data set. It is used in the TREC 2005 HARD Track (J. Allan, 2005). The AQUAINT corpus is prepared by the LDC for the AQUAINT Project, and is used in official benchmark evaluations conducted by National Institute of Standards and Technology (NIST). It contains news from three sources: the Xinhua News Service (People's Republic of China), the New York Times News Service, and the Associated Press Worldstream News Service.

The topics we used are the same as the TREC Robust track (E. M. Voorhees, 2005), which are the topics from number 303 to number 689 of the TREC topics. Each topic is described in three formats including titles, descriptions and narratives. In our experiment, due to the fact that our observation sequence is very sensitive to the query terms, we only focus on the title part of the topic. In this way, we can avoid some commonly appeared words in narratives or descriptions, which may reduce the precision of our training procedure for finding the real document. Table 2 shows the detail about the corpus.

Datasize	2.96GB
#Documents	1,030,561
#Querys	50
Term Types	2,002,165
Term Tokens	431,823,255

Table 2. Statistics of the AQUAINT corpus

4.2 Experiment Design and Results

By using the AQUAINT corpus, two different traditional IR methods are implemented for comparing. The two IR methods which we use as baselines are the General Language Modeling (GLM) proposed by Song and Croft (1999) and the tf.idf measure proposed by Robertson (1995). The GLM has been introduced in Section 2. The following formulas show the core of tf.idf:

$$\mathrm{tf.idf}(Q, D) = \sum_{q_i \in Q} \mathrm{wtf}(q_i, D) \cdot \mathrm{idf}(q_i)$$

$$\mathrm{wtf}(q, D) = \frac{\mathrm{tf}(q, D)}{\mathrm{tf}(q, D) + 0.5 + 1.5\frac{l(D)}{al}}$$

$$\mathrm{idf}(q) = \frac{\log\frac{N}{n_q}}{N + 1}$$

N is the number of documents in the corpus; n_q is the number of documents in the corpus containing *q*; tf(*q*, *D*) is the number of times *q* appears in *D*; *l*(*D*) is the length of *D* in words and the *al* is the average length in words of a *D* in the corpus.

For the proposed EM IR approach, two configurations are listed to compare. The first (Config.1) is the proposed HMM model without making use of the EM-based document weighting that is don't multiply the transition probability, trans(*d*), in equation (2). The second (Config.2) is the HMM model with EM-based document weighting. The comparison is based on precision. For each problem, we retrieved the documents with the highest 20 scores, and divided the number of correct answer with the number of retrieved document to obtain precision. If there are documents with same score at the rank of 20, all of them will be retrieved.

Methods	Precision	%Change	%Change
tf.idf	29.7%	-	
GLM	30.5%	2.69%	-
Config.1	28.8%	-5.58%	-3.14%
Config.2	32.2%	8.41%	5.57%

Table 3. Experiment Results of three IR methods on the AQUAINT corpus

As shown in Table 3, our EM IR system outperforms tf.idf method 8.41% and GLM method 5.57%.

5 Discussion

In this section, we will discuss the effectiveness of the EM-based document weighting and the EM procedure. Both of them rely on the HMM design we have proposed.

5.1 The effectiveness of EM-based document weighting

When we establish our HMM model, the transition probability from Initial State to the document state is assigned as uniform, since we don't have any information about the importance of every document. These transition probabilities represent the probability of choosing the document with the given observation sequence.

During EM training procedure, the transition probability, exclusive the transition probability from document states which is fixed to 100% to the Initial State, will be re-estimated according to the observation sequence (the query) and the observation probabilities of each state. As shown in Table 3, two configurations (Config.1 and Config.2) are conducted to verify the effectiveness of using the transition probability.

The transition probability works due to the EM training procedure. The training procedure works for maximizing the probability for generating the query words, so the weight for each document will be given according to mathematical formula. The advantage of this mechanism is it will use the same formula regardless of different content of document. Yet other statistical methods will have to fix the content or formula previously to avoid the noise or other disturbance. Some researches employee the number of terms in the document to calculate the document weighting. Since the observation probability already use the number of words in a document N_d as a parameter, using number of words as document weight will make it affect too much in our system.

The experiment results show an improvement of 11.80% by using the transition probability of Initial State. Accordingly, we can understand that the EM procedure helps our HMM model not only on the observation probability of generating query words, but also suggests a useful weight for each document.

5.2 The effectiveness of EM training

In HMM model training, the iteration numbers of EM procedure is always a tricky issue for experiment design. While training with too much iteration will lead to overfitting for the observation sequence, to less iteration will weaken the effect of EM training.

For our EM IR system, we've made a series of experiments with different iterations for examining the effect of EM training. Figure 3 shows the results.

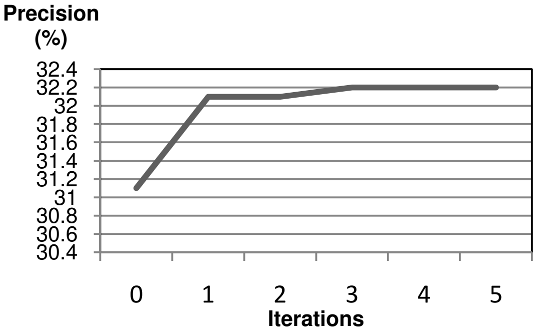


Figure 4. The precision change with the EM training iterations

As you can see in Figure 4, the precision increased with the iteration numbers. Still, the growing rate of precision becomes very slow after 2 iterations. We have analysis this result and find out two possible causes for this evidence. First, the training document sets are limited in a small size due to the computation time complexity for our approach. Therefore we can only retrieve correct document with high score in basic language modeling, which is used for document reduction. So the precision is also limited with the performance of our reducing methods. The number of correct answer is limited by the basic language modeling, so as the highest precision our system can achieve. Second, our observation only composed query terms, which gives a limited improving space.

6 Conclusion

We have proposed a method for using EM algorithm to improve the precision in information retrieval. This method employees the concept of language model approach, and merge it with the HMM. The transition probability in HMM is treated as the probability of choosing the document, and the observation probability in HMM is treated as the probability of generating the terms for the document. We also implement this method, and compare it with two existing IR methods with the dataset from TREC 2005 HARD Track. The experiment results show that the proposed approach outperforms two existing methods by 2.4% and 1.6% in precision, which are 8.08% and 5.24% increasing for the existing method. The effectiveness of using the tuned transition probability and EM training procedure is also discussed, and been proved can work effectively.

7 Future Work

Since we have achieved such improvement with EM algorithm, other kinds of algorithm with similar functions can also be tried in IR system. It might be work in the form of parameter reestimation, tuning or even generating parameters by statistical measure.

For the method we have proposed, we also have some part can be done in the future. Finding a better observance sequence will be an important issue. Since we use the exact query terms as our observance sequence, it's possible to use the method like statistical translation to generate more words which are also related with the documents we want and used as observance sequence.

Another possible issue is to integrate the bigram or trigram information into our training procedure. Corpus information might be used in more delicate way to improve the performance.

References

A. Berger and J. Lafferty, "Information retrieval as statistical translation," 1999, pp. 222-229.

A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society,* vol. 39, pp. 1-38, 1977.

C. D. Manning and H. Schutze, *Foundations of statistical natural language processing*: MIT Press, 1999.

D. Hiemstra and A. P. de Vries, *Relating the new language models of information retrieval to the traditional retrieval models*: University of Twente [Host]; University of Twente, Centre for Telematics and Information Technology, 2000.

D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel, "Nymble: a high-performance learning namefinder," 1997, pp. 194-201.

D. R. H. Miller, T. Leek, and R. M. Schwartz, "A hidden Markov model information retrieval system," 1999, pp. 214-221.

E. M. Voorhees, "The TREC robust retrieval track," 2005, pp. 11-20.

F. Song and W. B. Croft, "A general language model for information retrieval," 1999, pp. 316-321.

G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*: McGraw-Hill, Inc. New York, NY, USA, 1986.

J. Allan, "HARD track overview in TREC 2005: High accuracy retrieval from documents," 2005.

J. Makhoul and R. Schwartz, "State of the Art in Continuous Speech Recognition," *Proceedings of the National Academy of Sciences,* vol. 92, pp. 9956-9963, 1995.

J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," 1998, pp. 275-281.

L. E. Bmjm, T. Petrie, G. Soules, and N. Weiss, "A MAXIMIZATION TECHNIQUE OCCURRING IN THE STATISTICAL ANALYSIS OF PROBABILISTIC FUNCTIONS OF MARKOV CHAINS," *The Annals of Mathematical Statistics,* vol. 41, pp. 164-171, 1970.

L. Rabiner and B. Juang, "An introduction to hidden Markov models," *ASSP Magazine, IEEE [see also IEEE Signal Processing Magazine],* vol. 3, pp. 4-16, 1986.

R. Schwartz, T. Imai, F. Kubala, L. Nguyen, and J. Makhoul, "A Maximum Likelihood Model for Topic Classification of Broadcast News," 1997.

S. E. Robertson, "The probability ranking principle in IR," *Journal of Documentation,* vol. 33, pp. 294-304, 1977.

S. E. Robertson and S. Jones, "Relevance Weighting of Search Terms," *Journal of the American Society for Information Science,* vol. 27, pp. 129-46, 1976.

S. E. Robertson and S. Walker, "Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval," 1994, pp. 232-241.

S. E. Robertson, S. Walker, and S. Jones, "M. Hancock-Beaulieu, M., and Gatford, M.(1995). Okapi at TREC-3," pp. 109–126.

V. Bush, "As we may think," *interactions,* vol. 3, pp. 35-46, 1996.

V. Lavrenko and W. B. Croft, "Relevance based language models," 2001, pp. 120-127.

Data Cleaning for Word Alignment

Tsuyoshi Okita
CNGL / School of Computing
Dublin City University, Glasnevin, Dublin 9
tokita@computing.dcu.ie

Abstract

Parallel corpora are made by human beings. However, as an MT system is an aggregation of state-of-the-art NLP technologies without any intervention of human beings, it is unavoidable that quite a few sentence pairs are beyond its analysis and that will therefore not contribute to the system. Furthermore, they in turn may act against our objectives to make the overall performance worse. Possible unfavorable items are $n : m$ mapping objects, such as paraphrases, non-literal translations, and multiword expressions. This paper presents a pre-processing method which detects such unfavorable items before supplying them to the word aligner under the assumption that their frequency is low, such as below 5 percent. We show an improvement of Bleu score from 28.0 to 31.4 in English-Spanish and from 16.9 to 22.1 in German-English.

1 Introduction

Phrase alignment (Marcu and Wong, 02) has recently attracted researchers in its theory, although it remains in infancy in its practice. However, a phrase extraction heuristic such as grow-diag-final (Koehn et al., 05; Och and Ney, 03), which is a single difference between word-based SMT (Brown et al., 93) and phrase-based SMT (Koehn et al., 03) where we construct word-based SMT by bidirectional word alignment, is nowadays considered to be a key process which leads to an overall improvement of MT systems. However, technically, this phrase extraction process after word alignment is known to have at least two limitations: 1) the objectives of uni-directional word alignment is limited only in $1 : n$ mappings and 2) an atomic unit of phrase pair used by phrase extraction is thus basically restricted in $1 : n$ or $n : 1$ with small exceptions.

Firstly, the posterior-based approach (Liang, 06) looks at the posterior probability and partially delays the alignment decision. However, this approach does not have any extension in its $1 : n$ uni-directional mappings in its word alignment. Secondly, the aforementioned phrase alignment (Marcu and Wong, 02) considers the $n : m$ mapping directly bilingually generated by some concepts without word alignment. However, this approach has severe computational complexity problems. Thirdly, linguistic motivated phrases, such as a tree aligner (Tinsley et al., 06), provides $n : m$ mappings using some information of parsing results. However, as the approach runs somewhat in a reverse direction to ours, we omit it from the discussion. Hence, this paper will seek for the methods that are different from those approaches and whose computational cost is cheap.

$n : m$ mappings in our discussion include paraphrases (Callison-Burch, 07; Lin and Pantel, 01), non-literal translations (Imamura et al., 03), multiword expressions (Lambert and Banchs, 05), and some other noise in one side of a translation pair (from now on, we call these 'outliers', meaning that these are not systematic noise). One common characteristic of these $n : m$ mappings is that they tend to be so flexible that even an exhaustive list by human beings tends to be incomplete (Lin and Pantel, 01). There are two cases which we should like to distinguish: when we use external resources and when we do not. For example, Quirk et al. employ external resources by drawing pairs of English sentences from a comparable corpus (Quirk et al., 04), while Bannard and Callison-Burch (Bannard and Callison-Burch, 05) identified English paraphrases by pivoting through phrases in another language. However, in this paper our interest is rather the case when our resources are limited within our parallel corpus.

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 72–80,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

Imamura et al. (Imamura et al., 03), on the other hand, do not use external resources and present a method based on literalness measure called TCR (Translation Correspondence Rate). Let us define literal translation as a word-to-word translation, and non-literal translation as a non word-to-word translation. Literalness is defined as a degree of literal translation. Literalness measure of Imamura et al. is trained from a parallel corpus using word aligned results, and then sentences are selected which should either be translated by a 'literal translation' decoder or by a 'non-literal translation' decoder based on this literalness measure. Apparently, their definition of literalness measure is designed to be high recall since this measure incorporates all the possible correspondence pairs (via realizability of lexical mappings) rather than all the possible true positives (via realizability of sentences). Adding to this, the notion of literal translation may be broader than this. For example, literal translation of "C'est la vie." in French is "That's life." or "It is the life." in English. If literal translation can not convey the original meaning correctly, non-literal translation can be applied: "This is just the way life is.", "That's how things happen.", "Love story.", and so forth. Non-literal translation preserves the original meaning[1] as much as possible, ignoring the exact word-to-word correspondence. As is indicated by this example, the choice of literal translation or non-literal translation seems rather a matter of translator preference.

This paper presents a pre-processing method using the alternative literalness score aiming for high precision. We assume that the percentages of these $n : m$ mappings are relatively low. Finally, it turned out that if we focus on outlier ratio, this method becomes a well-known sentence cleaning approach. We refer to this in Section 5.

This paper is organized as follows. Section 2 outlines the $1 : n$ characteristics of word alignment by IBM Model 4. Section 3 reviews an atomic unit of phrase extraction. Section 4 explains our Good Points Algorithm. Experimental results are presented in Section 5. Section 6 discusses a sentence cleaning algorithm. Section 7 concludes and provides avenues for further research.

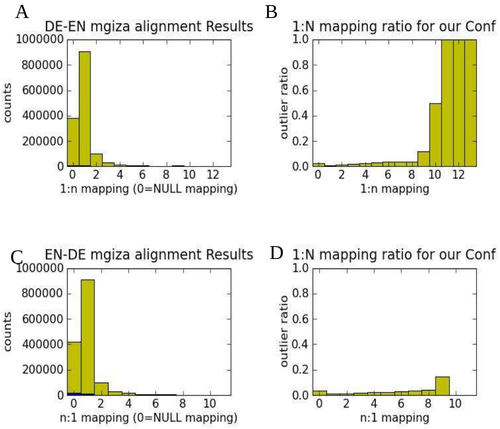


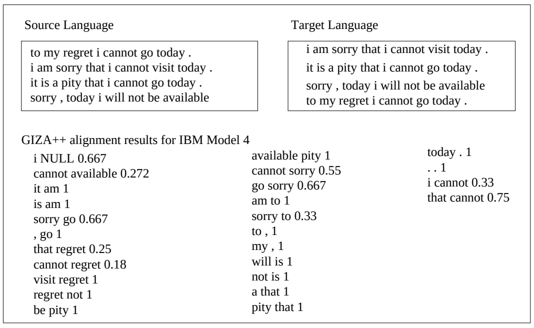
Figure 1: Figures A and C show the results of word alignment for DE-EN where outliers detected by Algorithm 1 are shown in blue at the bottom. We check all the alignment cept pairs in the training corpus inspecting so-called A3 final files by type of alignment from 1:1 to 1:13 (or NULL alignment). It is noted that outliers are miniscule in A and C because each count is only 3 percent. Most of them are NULL alignment or 1:1 alignment, while there are small numbers of alignments with 1:3 and 1:4 (up to 1:13 in the DE-EN direction in Figure A). In Figure C, 1:11 is the greatest. Figure B and D show the ratio of outliers over all the counts. Figure B shows that in the case of 1:10 alignments, 1/2 of the alignments are considered to be outliers by Algorithm 1, while 100 percent of alignment from 1:11 to 1:13 are considered to be outliers (false negative). Figure D shows that in the case of EN-DE, most of the outlier ratios are less than 20 percent.

2 $1 : n$ Word Alignment

Our discussion of uni-directional alignments of word alignment is limited to IBM Model 4.

Definition 1 (Word alignment task) *Let* e_i *be the* i*-th sentence in target language,* $\bar{e}_{i,j}$ *be the* j*-th word in* i*-th sentence, and* $\bar{e}_i$ *be the* i*-th word in parallel corpus (Similarly for* f_i*,* $\bar{f}_{i,j}$*, and* $\bar{f}_i$*). Let* $|e_i|$ *be a sentence length of* e_i*, and similarly for* $|f_i|$*. We are given a pair of sentence aligned bilingual texts* $(f_1, e_1), \ldots, (f_n, e_n) \in \mathcal{X} \times \mathcal{Y}$*, where* $f_i = (\bar{f}_{i,1}, \ldots, \bar{f}_{i,|f_i|})$ *and* $e_i = (\bar{e}_{i,1}, \ldots, \bar{e}_{i,|e_i|})$*. It is noted that* e_i *and* f_i *may include more than one sentence. The task of word alignment is to find a lexical translation probability* $p_{\bar{f}_i} : \bar{e}_i \rightarrow p_{\bar{f}_j}(\bar{e}_i)$ *such that* $\Sigma p_{\bar{f}_j}(\bar{e}_i) = 1$ *and* $\forall \bar{e}_i : 0 \leq p_{\bar{f}_j}(\bar{e}_i) \leq 1$ *(It is noted that some models such*

[1] Dictionary goes as follows: something that you say when something happens that you do not like but which you have to accept because you cannot change it [Cambridge Idioms Dictionary 2nd Edition, 06].



Source Language

to my regret i cannot go today .
i am sorry that i cannot visit today .
it is a pity that i cannot go today .
sorry , today i will not be available

Target Language

i am sorry that i cannot visit today .
it is a pity that i cannot go today .
sorry , today i will not be available
to my regret i cannot go today .

GIZA++ alignment results for IBM Model 4

```
i NULL 0.667
cannot available 0.272
it am 1
is am 1
sorry go 0.667
, go 1
that regret 0.25
cannot regret 0.18
visit regret 1
regret not 1
be pity 1
available pity 1
cannot sorry 0.55
go sorry 0.667
am to 1
sorry to 0.33
to , 1
my , 1
will is 1
not is 1
a that 1
pity that 1
today . 1
. . 1
i cannot 0.33
that cannot 0.75
```

Figure 2: Example shows an example alignment of paraphrases in a monolingual case. Source and target use the same set of sentences. Results show that only the matching between the colon is correct[3].

as IBM Model 3 and 4 have deficiency problems). It is noted that there may be several words in source language and target language which do not map to any words, which are called unaligned (or null aligned) words. Triples $(\bar{f}_i, \bar{e}_i, p_{\bar{f}_i}(\bar{e}_1))$ *(or* $(\bar{f}_i, \bar{e}_i, -\log_{10} p_{\bar{f}_i}(\bar{e}_1))$*) are called T-tables.*

As the above definition shows, the purpose of the word alignment task is to obtain a lexical translation probability $p(\bar{f}_i|\bar{e}_i)$, which is a $1 : n$ uni-directional word alignment. The initial idea underlying the IBM Models, consisting of five distinctive models, is that it introduces an alignment function $a(j|i)$, or alternatively the distortion function $d(j|i)$ or $d(j - \odot_i)$, when the task is viewed as a missing value problem, where i and j denote the position of a cept in a sentence and $\odot_i$ denotes the center of a cept. $d(j|i)$ denotes a distortion of the absolute position, while $d(j-\odot_i)$ denotes the distortion of relative position. Then this missing value problem can be solved by EM algorithms : E-step is to take expectation of all the possible alignments and M-step is to estimate maximum likelihood of parameters by maximizing the expected likelihood obtained in the E-step. The second idea of IBM Models is in the mechanism of fertility and a NULL insertion, which makes the performance of IBM Models competitive. Fertility and a NULL insertion is used to adjust the length n when the length of the source sentence is different from this n. Fertility is a mechanism to augment one source word into several source words or delete a source word, while a NULL insertion is a mechanism of generating several words from blank words. Fertility uses a conditional probability depending only on the lexicon. For example, the length of 'today' can be conditioned only on the lexicon 'today'.

As is already mentioned, the resulting alignments are $1 : n$ (shown in the upper figure in Figure 1). For DE-EN News Commentary corpus, most of the alignments fall in either 1:1 mapping or NULL mappings whereas small numbers are 1:2 mappings and miniscule numbers are from 1:3 to 1:13. However, this $1 : n$ nature of word alignment will cause problems if we encounter $n : m$ mapping objects, such as a paraphrase, nonliteral translation, or multiword expression. Figure 2 shows such difficulties where we show a monolingual paraphrase. Without loss of generality this can be easily extended to bilingual paraphrases. In this case, results of word alignment are completely wrong, with the exception of the example consisting of a colon. Although these paraphrases, nonliteral translations, and multiword expressions do not always become outliers, they may face the potential danger of producing the incorrect word alignments with incorrect probabilities.

3 Phrase Extraction and Atomic Unit of Phrases

The phrase extraction is a process to exploit phrases for a given bi-directional word alignment (Koehn et al., 05; Och and Ney, 03). If we focus on its generative process, this would become as follows: 1) add intersection of two word alignments as an alignment point, 2) add new alignment points that exist in the union with the constraint that a new alignment point connects at least one previously unaligned word, 3) check the unaligned row (or column) as unaligned row (or column, respectively), 4) if n alignment points are contiguous in horizontal (or vertical) direction we consider that this is a contiguous $1 : n$ (or $n : 1$) phrase pair (let us call these type I phrase pairs), 5) if a neighborhood of a contiguous $1 : n$ phrase pair is (an) unaligned row(s) or (an) unaligned column(s) we grow this region (with consistency constraint) (let us call these type II phrase pair), and 6) we consider all the diagonal combinations of type I and

[3] It is noted that there might be a criticism that this is not a fair comparison because we do not have sufficient data. Under a transductive setting (where we can access the test data), we believe that our statement is valid. Considering the nature of the $1 : n$ mapping, it would be quite lucky if we obtain $n : m$ mapping after phrase extraction (Our focus is not on the incorrect probability, but rather on the incorrect matching.)

type II phrase pairs generatively.

The atomic unit of type I phrase pairs is $1 : n$ or $n : 1$, while that of type II phrase pairs is $n : m$ if unaligned row(s) and column(s) exist in neighborhood. So, whether they form a $n : m$ mapping or not depends on the existence of unaligned row(s) and column(s). And at the same time, n or m should be restricted to a small value. There is a chance that a $n : m$ phrase pair can be created in this way. This is because around one third of word alignments, which is quite a large figure, are $1 : 0$ as is shown in Figure 1. Nevertheless, our concern is if the results of word alignment is very low quality, e.g. similar to the situation depicted in Figure 2, this mechanism will not work. Furthermore, this mechanism is only restricted in the unaligned row(s) and column(s).

4 Our Approach: Good Points Approach

Our approach aims at removing *outliers* by the literalness score, which we defined in Section 1, between a pair of sentences. Sentence pairs with low literalness score should be removed. Following two propositions are the theory behind this. Let a word-based MT system be M_{WB} and a phrase-based MT system be M_{PB}. Then,

Proposition 1 *Under an ideal MT system* M_{PB}*, a paraphrase is an inlier (or realizable), and*

Proposition 2 *Under an ideal MT system* M_{WB}*, a paraphrase is an outlier (or not realizable).*

Based on these propositions, we could assume that if we measure the literalness score under a word-based MT M_{WB} we will be able to determine the degree of *outlier*-ness whatever the measure we use for it. Hence, what we should do is, initially, to score it under a word-based MT M_{WB} using Bleu, for example. (Later we replace it with a variant of Bleu, i.e. cumulative n-gram score). However, despite Proposition 1, our MT system at hand is unfortunately not ideal. What we can currently do is the following: if we witness bad sentence-based scores in word-based MT, we can consider our MT system failing to incorporating a $n : m$ mapping object for those sentences. Later in our revised version, we use both of word-based MT and phrase-based MT. The summary of our first approach becomes as follows: 1) employing the mechanism of word-based MT trained on the same parallel corpus, we measure the literalness between a pair of sentences, 2) we use the variants

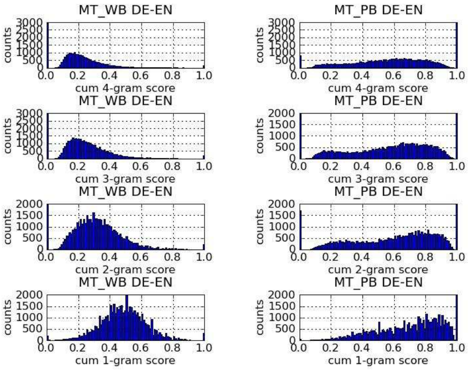


Figure 3: Left figure shows sentence-based Bleu score of word-based SMT and right figure shows that of phrase-based SMT. Each row shows the cumulative n-gram score (n = 1,2,3,4) and we use News Commentary parallel corpus (DE-EN).

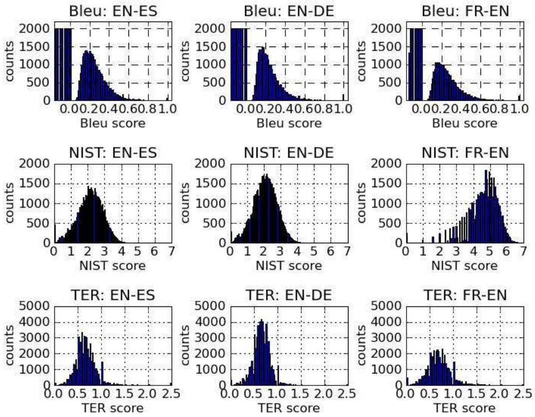


Figure 4: Each row shows Bleu, NIST, and TER, while each column shows different language pairs (EN-ES, EN-DE and FR-DE). These figures show the scores of all the training sentences by the word-based SMT system. In the row for Bleu, note that the area of rectangle shows the number of sentence pairs whose Bleu scores are zero. (There are a lot of sentence pairs whose Bleu score are zero: if we draw without en-folding the coordinate, these heights reach to 25,000 to 30,000.) There is a smooth probability distribution in the middle, while there are two non-smoothed connections at 1.0 and 0.0. Notice there is a small number of sentences whose score is 1.0. In the middle row for NIST score, similarly, there is a smooth probability distribution in the middle and we have a non-smoothed connection at 0.0. In the bottom row for TER score, the 0.0 is the best score unlike Bleu and NIST, and we omit scores more than 2.5 in these figures. (The maximum was 27.0.)

of Bleu score as the measure of literalness, and 3) based on this score, we reduce the sentences in parallel corpus. Our algorithm is as follows:

Algorithm 1 Good Points Algorithm

```
Step 1: Train word-based MT.
Step 2: Translate all training sentences by the above trained word-based MT decoder.
Step 3: Obtain the cumulative $X$-gram score for each pair of sentences where $X$ is 4, 3, 2, and 1.
Step 4: By the threshold described in Table 1, we produce new reduced parallel corpus.
(Step 5: Do the whole procedure of phrase-based SMT using the reduced parallel corpus which we obtain from Step 1 to 4.)
```

conf	A1	A2	A3	A4
Ours	0.05	0.05	0.1	0.2
1	0.1			
2	0.1	0.2		
3	0.1	0.2	0.3	0.5
4	0.05	0.1	0.2	0.4
5	0.22	0.3	0.4	0.6
6	0.25	0.4	0.5	0.7
7	0.2	0.4	0.5	0.8
8				0.6

Table 1: Table shows our threshold where A1, A2, A3, and A4 correspond to the absolute cumulative n-gram precision value (n=1,2,3,4 respectively). In experiments, we compare ours with eight configurations above in Table 6.

but this does not matter . peu importe !
we may find ourselves there once again . va-t-il en être de même cette fois-ci ?
all for the good . et c’ est tant mieux !
but if the ceo is not accountable , who is ? mais s’ il n’ est pas responsable , qui alors ?

Table 2: Sentences judged as outliers by Algorithm 1 (ENFR News Commentary corpus).

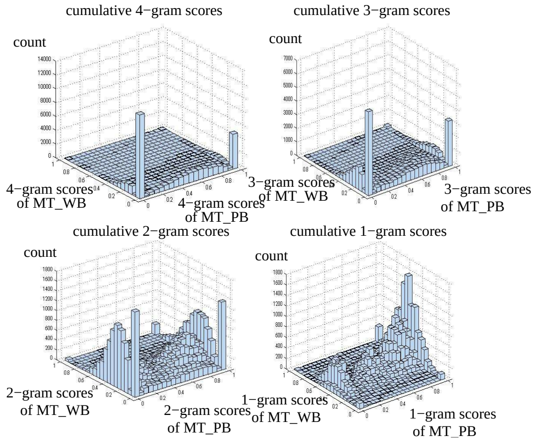


Figure 5: Four figures show the sentence-based cumulative n-gram scores: x-axis is phrase-based SMT and y-axis is word-based SMT. Focus is on the worst point (0,0) where both scores are zero. Many points reside in (0,0) in cumulative 4-gram scores, while only small numbers of point reside in (0,0) in cumulative 1-gram scores.

We would like to mention our motivation for choosing the variant of Bleu. In Step 3 we need to set up a threshold in M_{WB} to determine *outliers*. Natural intuition is that this distribution takes some smooth distribution as Bleu takes weighted geometric mean. However, as is shown in the first row of Figure 4, typical distribution of words in this space M_{WB} is separated in two clusters: one looks like a geometric distribution and the other one contains a lot of points whose value is zero. (Especially in the case of Bleu, if the sentence length is less than 3 the Bleu score is zero.) For this reason, we use the variants of Bleu score: we decompose Bleu score in cumulative n-gram score (n=1,2,3,4), which is shown in Figure 3. It is noted that the following relation holds: $S_4(e,f) \leq S_3(e,f) \leq S_2(e,f) \leq S_1(e,f)$ where e denotes an English sentence, f denotes a foreign sentence, and S_X denotes cumulative X-gram scores. For 3-gram scores, the tendency to separate in two clusters is slightly decreased. Furthermore, for 1-gram scores, the distribution approaches to normal distribution. We model P(outlier) taking care of the quantity of $S_2(e,f)$, where we choose 0.1: other configurations in Table 1 are used in experiments. It is noted that although we choose the variants of Bleu score, it is clear, in this context, that we can replace Bleu with any other measure, such as METEOR (Banerjee and Lavie, 05), NIST (Doddington, 02), GTM (Melamed et al., 03), TER (Snover et al., 06), labeled dependency approach (Owczarzak et al., 07) and so forth (see Figure 4). Table 2 shows outliers detected by Algorithm 1.

Finally, a revised algorithm which incorporates sentence-based X-gram scores of phrase-based MT is shown in Algorithm 2. Figure 5 tells us that there are many sentence pair scores actually improved in phrase-based MT even if word-based score is zero.

Algorithm 2 Revised Good Points Algorithm

```
Step 1: Train word-based MT for full parallel corpus. Translate all training sentences by the above trained word-based MT decoder.
Step 2: Obtain the cumulative $X$-gram score $S_{WB,X}$ for each pair of sentences where $X$ is 4, 3, 2, and 1 for word-based MT decoder.
Step 3: Train phrase-based MT for full parallel corpus. Note that we do not need to run a word aligner again in here, but use the results of Step 1. Translate all training sentences by the above trained phrase-based MT decoder.
Step 4: Obtain the cumulative $X$-gram score $S_{PB,X}$ for each pair of sentences where $X$ is 4, 3, 2, and 1 for phrase-based MT decoder.
Step 5: Remove sentences whose $(S_{WB,2}, S_{PB,2}) = (0, 0)$. We produce new reduced parallel corpus.
(Step 6: Do the whole procedure of phrase-based SMT using the reduced parallel corpus which we obtain from Step 1 to 5.)
```

5 Results

We evaluate our algorithm using the News Commentary parallel corpus used in 2007 Statistical Machine Translation Workshop shared task (corpus size and average sentence length are shown in Table 8). We use the devset and the evaluation set provided by this workshop. We use Moses (Koehn et al., 07) as the baseline system, with mgiza (Gao and Vogel, 08) as its word alignment tool. We do MERT in all the experiments below.

alignment	ENFR	ESEN
grow-diag-final	0.058	0.115
union	0.205	0.116
intersection	0.164	0.116

Table 3: Performance of word-based MT system in different alignment methods. The above is between ENFR and ESEN.

pair	ENFR	FREN
score	0.205	0.176
ENES	ENDE	DEEN
0.276	0.134	0.208

Table 4: Performance of word-based MT system for different language pairs with union alignment method.

Step 1 of Algorithm 1 produces, for a given parallel corpus, a word-based MT. We do this using Moses with option max-phrase-length set to 1, alignment as union as we would like to extract the bi-directional results of word alignment with high recall. Although we have chosen union, other selection options may be possible as Table 3 suggests. Performance of this word-based MT system is as shown in Table 4.

Step 2 is to obtain the cumulative n-gram score for the entire training parallel corpus by using the word-based MT system trained in Step 1. Table 5 shows the first two sentences of News Commentary corpus. We score for all the sentence pairs.

c_score = [0.4213,0.4629,0.5282,0.6275] consider the number of clubs that have qualified for the european champions ' league top eight slots . considérons le nombre de clubs qui se sont qualifiés parmi les huit meilleurs de la ligue des champions europenne .
c_score = [0.0000,0.0000,0.0000,0.3298] estonia did not need to ponder long about the options it faced . l' estonie n' a pas eu besoin de longuement rflchir sur les choix qui s' offraient à elle .

Table 5: Four figures marked as score shows the cumulative n-gram score from left to right. The following EN and FR are the calculated sentences used by word-based MT system trained on Step 1.

In Step 3, we obtain the cumulative n-gram score (shown in Figure 3). As is already mentioned, there are a lot of sentence pairs whose cumulative 4-gram score is zero. In the cumulative 3-gram score, this tendency is slightly decreased. For 1-gram scores, the distribution approaches to normal distribution. In Step 4, other than our configuration we used 8 different configurations in Table 6 to reduce our parallel corpus.

Now we obtain the reduced parallel corpus. In Step 5, using this reduced parallel corpus we carried out training of MT system from the beginning: we again started from the word alignment, followed by phrase extraction, and so forth. The results corresponding to these configurations are shown in Table 6. In Table 6, in the case of

ENES	Bleu	effective sent	UNK
Base	0.280	99.30 %	1.60%
Ours	<u>0.314</u>	96.54%	1.61%
1	0.297	56.21%	2.21%
2	0.294	60.37%	2.09%
3	0.301	66.20%	1.97%
4	0.306	84.60%	1.71%
5	0.299	56.12%	2.20%
6	0.271	25.05%	2.40%
7	0.283	35.28%	2.26%
8	0.264	19.78%	4.22%

	DEEN	%	ENFR	%
Base	0.169	99.10%	0.180	91.81%
Ours	<u>0.221</u>	96.42%	<u>0.192</u>	96.38%
1	0.201	40.49%	0.187	49.37%
2	0.205	48.53%	0.188	55.03%
3	0.208	58.07%	0.187	61.22%
4	0.215	83.10%	0.190	81.57%
5	0.192	29.03%	0.180	31.52%
6	0.174	17.69%	0.162	29.97%
7	0.186	24.60%	0.179	30.52%
8	0.177	18.29%	0.167	17.11%

Table 6: Table shows Bleu score for ENES, DEEN, and ENFR: 0.314, 0.221, and 0.192, respectively. All of these are better than baseline. Effective ratio can be considered to be the inlier ratio, which is equivalent to 1 - (outlier ratio). The details for the baseline system are shown in Table 8.

ENES	Bleu	effective sent
Base	0.280	99.30 %
Ours	<u>0.317</u>	97.80 %
DEEN	Bleu	effective sent
Base	0.169	99.10 %
Ours	<u>0.218</u>	97.14 %

Table 7: This table shows results for the revised Good Points Algorithm.

English-Spanish our configuration discards 3.46 percent of sentences, and the performance reaches 0.314 which is the best among other configurations. Similarly in the case of German-English our configuration attains the best performance among configurations. It is noted that results for the baseline system are shown in Table 8 where we picked up the score where n is 100. It is noted that the baseline system as well as other configurations use MERT. Similarly, results for a revised Good Points Algorithm is shown in Table 7.

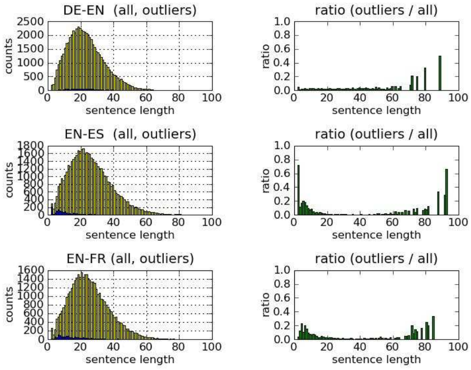


Figure 6: Three figures in the left show the histogram of sentence length (main figures) and histogram of sentence length of outliers (at the bottom). (As the numbers of outliers are less than 5 percent in each case, outliers are miniscule. In the case of EN-ES, we can observe the blue small distributions at the bottom from 2 to 16 sentence length.) Three figures in the right show that if we see this by ratio of outliers over all the counts, all of three figures tend to be more than 20 to 30 percent from 80 to 100 sentence length. The lower two figures show that sentence length 1 to 4 tend to be more than 10 percent.

6 Discussion

In Section 1, we mentioned that if we aim at outlier ratio using the indirect feature *sentence length*, this method reduces to a well-known sentence cleaning approach shown below in Algorithm 3.

Algorithm 3 Sentence Cleaning Algorithm

```
  Remove sentences with lengths greater than $X$ (or remove sentences with lengths smaller than $X$ in the case of short sentences).
```

This approach is popular although the reason behind why this approach works is not well understood. Our explanation is shown in the right-hand side of Figure 6 where outliers are shown at the bottom (almost invisible) which are extracted by Algorithm 1. The region that Algorithm 3 removes via sentence length X is possibly the region where the ratio of outliers is high.

This method is a high recall method. This method does not check whether the removed sentences are really sentences whose behavior is bad or not. For example, look at Figure 6 for sen-

X	ENFR	FREN	ESEN	DEEN	ENDE
10	0.167	0.088	0.143	0.097	0.079
20	0.087	0.195	0.246	0.138	0.127
30	0.145	0.229	0.279	0.157	0.137
40	0.175	0.242	0.295	0.168	0.142
50	0.229	0.250	0.297	0.170	0.145
60	0.178	0.253	0.297	0.171	0.146
70	0.179	0.251	0.298	0.170	0.146
80	0.181	0.252	0.301	0.169	0.147
90	0.180	0.252	0.297	0.171	0.147
100	0.180	0.251	0.302	0.169	0.146
#	51k	51k	51k	60k	60k
ave	21.0/23.8(EN/FR) 20.9/24.5(EN/ES)				
len	20.6/21.6(EN/DE)				

Table 8: Bleu score after cleaning of sentences with length greater than X. The row shows X, while the column shows the language pair. Parallel corpus is News Commentary parallel corpus. It is noted that the default setting of MAX_SENTENCE_LENTH_ALLOWED in GIZA++ is 101.

tence length 10 to 30 where there are considerably many outliers in the region that a lot of inliers reside. However, this method cannot cope with such outliers. Instead, the method cope with the region that the outlier ratio is possibly high at both ends, e.g. sentence length > 60 or sentence length < 5. The advantage is that sentence length information is immediately available from the sentence which is easy to implement. The results of this algorithm is shown in Table 8 where we varies X and language pair. This table also suggests that we should refrain from saying that $X = 60$ is best or $X = 80$ is best.

7 Conclusions and Further Work

This paper shows some preliminary results that data cleaning may be a useful pre-processing technique for word alignment. At this moment, we observe two positive results, improvement of Bleu score from 28.0 to 31.4 in English-Spanish and 16.9 to 22.1 in German-English which are shown in Table 6. Our method checks the realizability of target sentences in training sentences. If we witness bad cumulative X-gram scores we suspect that this is due to some problems caused by the $n : m$ mapping objects during word alignment followed by phrase extraction process.

Firstly, although we removed training sentences whose n-gram scores are low, we can duplicate such training sentences in word alignment. This method is appealing, but unfortunately if we use mgiza or GIZA++, our training process often ceased in the middle by unrecognized errors. However, if we succeed in training, the results often seem comparable to our results. Although we did not supply back removed sentences, it is possible to examine such sentences using the T-tables to extract phrase pairs.

Secondly, it seems that one of the key matters lies in the quantities of $n : m$ mapping objects which are difficult to learn by word-based MT (or by phrase-based MT). It is possible that such quantities are different depending on their language pairs and on their corpora size. A rough estimation is that this quantity may be somewhere less than 10 percent (in FR-EN Hansard corpus, recall and precision reach around 90 percent (Moore, 05)), or less than 5 percent (in News Commentary corpus, the best Bleu scores by Algorithm 1 are when this percentage is less than 5 percent). As further study, we intend to examine this issue further.

Thirdly, this method has other aspects that it removes discontinuous points: such discontinuous points may relate to the smoothness of optimization surface. One of the assumptions of the method such as Wang et al. (Wang et al., 07) relates to smoothness. Then, our method may improve their results, which is our further study.

In addition, although our algorithm runs a word aligner more than once, this process can be reduced since removed sentences are less than 5 percent or so.

Finally, we did not compare our method with TCR of Imamura. In our case, the focus was 2-gram scores rather than other n-gram scores. We intend to investigate this further.

8 Acknowledgements

This work is supported by Science Foundation Ireland (Grant No. 07/CE/I1142). Thanks to Yvette Graham and Sudip Naskar for proof reading, Andy Way, Khalil Sima'an, Yanjun Ma, and annonymous reviewers for comments, and Machine Translation Marathon.

References

Colin Bannard and Chris Callison-Burch. 2005. *Paraphrasing with bilingual parallel corpora*. ACL.

Satanjeev Banerjee and Alon Lavie. 2005. *METEOR: An Automatic Metric for MT Evaluation With Improved Correlation With Human Judgments*. Workshop On Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization.

Peter F. Brown, Vincent J.D. Pietra, Stephen A.D. Pietra, and Robert L. Mercer. 1993. *The Mathematics of Statistical Machine Translation: Parameter Estimation," Computational Linguistics, Vol.19, Issue 2*.

Chris Callison-Burch. 2007. *Paraphrasing and Translation*. PhD Thesis, University of Edinburgh.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. *Improved Statistical Machine Translation Using Paraphrases.* NAACL.

Chris Callison-Burch, Trevor Cohn, and Mirella Lapala. 2008. *ParaMetric: An Automatic Evaluation Metric for Paraphrasing.* COLING.

A.P. Dempster, N.M. Laird, and D.B. Rubin. 1977. *Maximum likelihood from Incomplete Data via the EM algorithm*. Journal of the Royal Statistical Society.

Yonggang Deng and William Byrne. 2005. *HMM Word and Phrase Alignment for Statistical Machine Translation*. Proc. Human Language Technology Conference and Empirical Methods in Natural Language Processing.

George Doddington. 2002. *Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics*. HLT.

David A. Forsyth and Jean Ponce. 2003. *Computer Vision*. Pearson Education.

Qin Gao and Stephan Vogel. 2008. *Parallel Implementations of Word Alignment Tool*. Software Engineering, Testing, and Quality Assurance for Natural Language Processing.

Kenji Imamura, Eiichiro Sumita, and Yuji Matsumoto. 2003. *Automatic Construction of Machine Translation Knowledge Using Translation Literalness*. EACL.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. *Statistical Phrase-Based Translation*. HLT/NAACL.

Philipp Koehn, Amittai Axelrod, Alexandra Birch, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. *Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation.* International Workshop on Spoken Language Translation.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. *Moses: Open Source Toolkit for Statistical Machine Translation*. ACL.

Patrik Lambert and Rafael E. Banchs. 2005. *Data Inferred Multiword Expressions for Statistical Machine Translation.* Machine Translation Summit X.

Percy Liang, Ben Taskar, and Dan Klein. 2006. *Alignment by agreement.* HLT/NAACL.

Dekang Lin and Patrick Pantel. 1999. *Induction of Semantic Classes from Natural Language Text*. In Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD-01).

Daniel Marcu and William Wong. 2002. *A Phrase-based, Joint Probability Model for Statistical Machine Translation*. In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP).

I. Dan Melamed, Ryan Green, and Joseph Turian. 2003. *Precision and Recall of Machine Translation.* NAACL/HLT 2003.

Robert C. Moore. 2005. *A Discriminative Framework for Bilingual Word Alignment.* HLT/EMNLP.

Franz Josef Och and Hermann Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models.* Computational Linguistics, volume 20,number 1.

Karolina Owczarzak, Josef van Genabith, and Andy Way. 2007. *Evaluating Machine Translation with LFG Dependencies*. Machine Translation, Springer, Volume 21, Number 2.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. *BLEU: A Method For Automatic Evaluation of Machine Translation* ACL.

Chris Quirk, Chris Brockett, and William Dolan. 2004. *Monolingual machine translation for paraphrase generation*. EMNLP-2004.

Matthew Snover. Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. *A Study of Translation Edit Rate with Targeted Human Annotation.* Association for Machine Translation in the Americas.

John Tinsley, Ventsisiav Zhechev, Mary Hearne, and Andy Way. 2006. *Robust Language Pair-Independent Sub-Tree Alignment*. Translation Summit XI.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. *HMM-based Word Alignment in Statistical Translation*. COLING 96.

Zhuoran Wang, John Shawe-Taylor, and Sandor Szedmak. 2007. *Kernel Regression Based Machine Translation*. Proceedings of NAACL-HLT 2007.

The Modulation of Cooperation and Emotion in Dialogue: The REC Corpus

Federica Cavicchio

Mind and Brain Center/ Corso Bettini 31,
38068 Rovereto (Tn) Italy
federica.cavicchio@unitn.it

Abstract

In this paper we describe the Rovereto Emotive Corpus (REC) which we collected to investigate the relationship between emotion and cooperation in dialogue tasks. It is an area where still many unsolved questions are present. One of the main open issues is the annotation of the so-called "blended" emotions and their recognition. Usually, there is a low agreement among raters in annotating emotions and, surprisingly, emotion recognition is higher in a condition of modality deprivation (i. e. only acoustic or only visual modality vs. bimodal display of emotion). Because of these previous results, we collected a corpus in which "emotive" tokens are pointed out during the recordings by psychophysiological indexes (ElectroCardioGram, and Galvanic Skin Conductance). From the output values of these indexes a general recognition of each emotion arousal is allowed. After this selection we will annotate emotive interactions with our multimodal annotation scheme, performing a kappa statistic on annotation results to validate our coding scheme. In the near future, a logistic regression on annotated data will be performed to find out correlations between cooperation and negative emotions. A final step will be an fMRI experiment on emotion recognition of blended emotions from face displays.

1 Introduction

In the last years many multimodal corpora have been collected. These corpora have been recorded in several languages and have being elicited with different methodologies: acted (such as for emotion corpora, see for example Goeleven, 2008), task oriented corpora, multiparty dialogs, corpora elicited with scripts or storytelling and ecological corpora. Among the goals of collection and analysis of corpora there is shading light on crucial aspects of speech production. Some of the main research questions are how language and gesture correlate with each other (Kipp et al., 2006) and how emotion expression modifies speech (Magno Caldognetto et al., 2004) and gesture (Poggi, 2007). Moreover, great efforts have been done to analyze multimodal aspects of irony, persuasion or motivation.

Multimodal coding schemes are mainly focused on dialogue acts, topic segmentation and the so called "emotional area". The collection of multimodal data has raised the question of coding scheme reliability. The aim of testing coding scheme reliability is to assess whether a scheme is able to capture observable reality and allows some generalizations. From mid Nineties, the kappa statistic has begun to be applied to validate coding scheme reliability. Basically, the kappa statistic is a statistical method to assess agreement among a group of observers. Kappa has been used to validate some multimodal coding schemes too. However, up to now many multimodal coding schemes have a very low kappa score (Carletta, 2007, Douglas-Cowie et al., 2005; Pianesi et al., 2005, Reidsma et al., 2008). This could be due to the nature of multimodal data. In fact, annotation of mental and emotional states of mind is a very demanding task. The low annotation agreement which affects multimodal corpora validation could also be due to the nature of the kappa statistics. In fact, the assumption underlining the use of kappa as reliability measure is that coding scheme categories are mutually exclusive and equally distinct one another. This is clearly difficult to be obtained in multimodal corpora annotation, as communication channels (i.e. voice, face movements, gestures and posture) are deeply interconnected one another.

To overcome these limits we are collecting a new corpus, Rovereto Emotive Corpus (REC), a task oriented corpus with psychophysiological data registered and aligned with audiovisual data. In our opinion this corpus will allow to clearly identify emotions and, as a result, having a clearer idea of facial expression of emotions in dialogue. In fact, REC is created to shade light on the relationship between cooperation and emotions in dialogues. This resource is the first

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 81–87,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

up to now with audiovisual and psychophysiological data recorded together.

2 The REC Corpus

REC (Rovereto Emotive Corpus) is an audiovisual and psychophysiological corpus of dialogues elicited with a modified Map Task. The Map Task is a cooperative task involving two participants. It was used for the first time by the HCRC group at Edinburg University (Anderson et al., 1991). In this task two speakers sit opposite one another and each of them has a map. They cannot see each other's map because the they are separated by a short barrier. One speaker, designated the Instruction Giver, has a route marked on her map; the other speaker, the Instruction Follower, has no route. The speakers are told that their goal is to reproduce the Instruction Giver's route on the Instruction Follower's map. To the speakers are told explicitly that the maps are not identical at the beginning of the dialogue session. However, it is up to them to discover how the two maps differ.

Our map task is modified with respect to the original one. In our Map Task the two participants are sitting one in front of the other and are separated by a short barrier or a full screen. They both have a map with some objects. Some of them are in the same position and with the same name, but most of them are in different positions or have names that sound similar to each other (e. g. Maso Michelini vs. Maso Nichelini, see Fig. 1). One participant (the giver) must drive the other participant (the follower) from a starting point (the bus station) to the finish (the Castle).

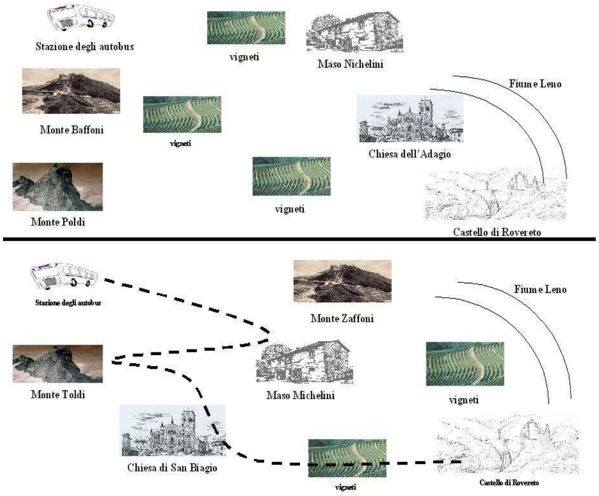


Figure 1: Maps used in the recording of REC corpus

Giver and follower are both native Italian speakers. In the instructions it was told them that they will have no more than 20 minutes to accomplish the task. The interaction has two conditions: screen and no screen. In screen condition a barrier was present between the two speakers. In no screen condition a short barrier, as in the original map task, was placed allowing giver and follower to see each other's face. With these two conditions we want to test whether seeing the speakers face during interactions influences facial emotion display and cooperation (see Kendon, 1967; Argyle and Cook 1976; for the relationship between gaze/no gaze and facial displays; for the influence of gaze on cooperation and coordination see Brennan et al., 2008). A further condition, emotion elicitation, was added. In "emotion" condition the follower or the giver can alternatively be a confederate, with the aim of getting the other participant angry. In this condition the psychophysiological state of the confederate is not recorded. In fact, as it is an acted behavior, it is not interesting for research purpose. All the participants had given informed consent and the experimental protocol has been approved by the Human Research Ethics Committee of Trento University.

REC is by now made up of 17 dyadic interactions, 9 with confederate, for a total of 204 minutes of audiovisual and psychophysiological recordings (electrocardiogram and derived heart rate value, and skin conductance). Our goal is reaching 12 recordings in the confederate condition. During each dialogue, the psychophysiological state of non-confederate giver or follower is recorded and synchronized with video and audio recordings. So far, REC corpus is the only multimodal corpus which has psychophysiological data to assess emotive states.

The psychophysiological state of each participant has been recorded with a BIOPAC MP150 system. In particular, Electrocardiogram (ECG) was recorded by Ag AgC1 surface electrodes fixed on participant's wrists, low pass filter 100 Hz, at a 200 samples/second rate. Heart Rate (HR) has been automatic calculated as number of heart beats per minute. Galvanic Skin Conductance (SK) was recorded with Ag AgC1 electrodes attached to the palmar surface of the second and third fingers of the non dominant hand, and recorded at a rate of 200samples/second. Artefacts due to hand movements have been removed with proper algorithms. Audiovisual interactions are recorded with 2 Canon Digital Cameras and 2 free field Sennheiser half-cardioid microphones with permanently polarized condenser, placed in front of each speaker

The recording procedure of REC is the following. Before starting the task, we record baseline condition that is to say we record participants' psychophysiological outputs for 5 minutes without challenging them. Then the task started and we recorded the psychophysiological outputs during the interaction which we called task condition. Then the confederate started challenging the speaker with the aim of getting him/her angry. To do so, the confederate at minutes 4, 9 and 13 of the interaction plays a script (negative emotion elicitation in giver; Anderson et al., 2005):

•*You driving me in the wrong direction, try to be more accurate!";*

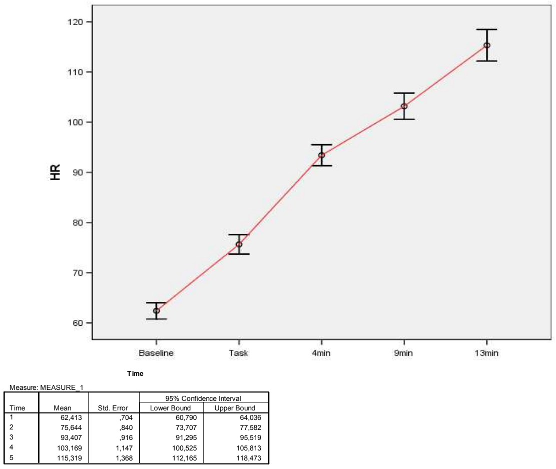
•*"It's still wrong, this can't be your best, try harder! So, again, from where you stop";*

•*"You're obviously not good enough in giving instruction".*

In Fig. 2 we show the results of a 1x5 ANOVA executed in confederate condition. Heart rate (HR) is confronted over the five times of interest (baseline, task, after 4 minutes, after 9 minutes, after 13 minutes). The times of interest are baseline, task, and after 4, 9 and 13 minutes, that is to say just after emotion elicitation with the script.

We find that HR is significantly different in the five conditions, which means that the procedure to elicit emotions is incremental and allows recognition of different psychophysiological states, which in turns are linked to emotive states. Mean HR values are in line with the ones showed by Anderson et al. (2005). Moreover, from the inspection of skin conductance values (Fig. 3) there is a linear increase of the number of peaks of conductance over time. This can be due to two factors: emotion elicitation but also an increasing of task difficulty leading to higher stress and therefore to an increasing number of skin conductance peaks.

As Cacioppo et al. (2000) pointed out, it is not possible to assess the emotion typology from psychophysiological data alone. In fact, HR and skin conductance are signals of arousal which in turns can be due both to high arousal emotions such as happiness or anger. Therefore, we asked participants after the conclusion of the task to report on a 8 points rank scale the valence of the emotions felt towards the interlocutor during the task (from extremely positive to extremely negative). On 10 participants, 50% of them rated the experience as quite negative, 30% rated the experience as almost negative, 10% of participants rated it as negative and 10% as neutral.



Measure: MEASURE_1

Time	Mean	Std. Error	95% Confidence Interval	
			Lower Bound	Upper Bound
1	62,413	,704	60,790	64,036
2	75,644	,840	73,707	77,582
3	93,407	,916	91,295	95,519
4	103,169	1,147	100,525	105,813
5	115,319	1,368	112,165	118,473

Figure 2: 1x5 ANOVA on heart rate (HR) over time in emotion elicitation condition in 9 partecipants

Participants who have reported a neutral or positive experience were discarded from the corpus.

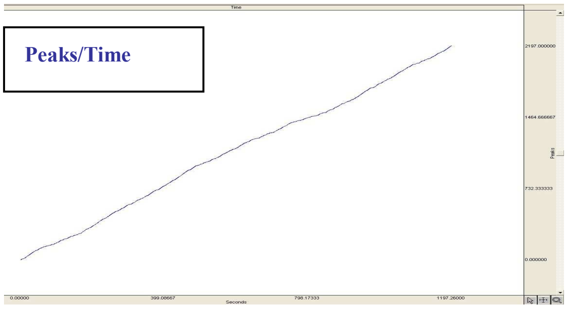


Figure 3: Number of skin conductance positive peaks over time in emotion elicitation condition in 9 partecipants

3 Annotation Method and Coding Scheme

The emotion annotation coding scheme used to analyze our map task is quite far from the emotion annotation schemes proposed in Computational Linguistic literature. Craggs and Woods (2005) proposed to annotate emotions with a scheme where emotions are expressed at different blending levels (i. e. blending of different emotion and emotive levels). In Craggs and Woods opinions' annotators must label the given emotion with a main emotive term (e. g. anger, sadness, joy etc.) correcting the emotional state with a score ranging from 1 (low) to 5 (very high). Martin et al. (2006) used a three steps rank scale of emotion valence (positive, neutral and negative) to annotate their corpus recorded from TV interviews.

But both these methods had quite poor results in terms of annotation agreement among coders.

Several studies on emotions have shown how emotional words and their connected concepts influence emotion judgments and their labeling (for a review, see Feldman Barrett et al., 2007). Thus, labeling an emotive display (e. g. a voice or a face) with a single emotive term could be not the best solution to recognize an emotion. Moreover researchers on emotion recognition from face displays find that some emotions as anger or fear are discriminated only by mouth or eyes configurations. Face seems to be evolved to transmit orthogonal signals, with a lower correlation each other. Then, these signals are deconstructed by the "human filtering functions", i. e. the brain, as optimized inputs (Smith et al., 2005). The Facial Action Units (FACS, Ekman and Friesen, 1978) is a good scheme to annotate face expressions starting from movement of muscular units, called action units. Even if accurate, it is a little problematic to annotate facial expression, especially the mouth ones, when the subject to be annotated is speaking, as the muscular movements for speech production overlaps with the emotional configuration.

On the basis of such findings, an ongoing debate is whether the perception of a face and, specifically, of a face displaying emotions, is based on holistic perception or perception of parts. Although many efforts are ongoing in neuroscience to determine the basis of emotion perception and decoding, little is still known on how brains and computer might learn part of an object such as a face. Most of the research in this field is based on PCA-alike algorithms which learn holistic representations. On the contrary other methods such as non Negative Matrix Factorization are based on only positive constrains leading to part based additive representations. Keeping this in mind, we decide not to label emotions directly but to attribute valence and activation to nonverbal signals, "deconstructing" them in simpler elements. These elements have implicit emotive dimensions, as for example mouth shape. Thus, in our coding scheme a smile would be annotate as ")" and a large smile as "+)". The latter means a higher valence and arousal than the previous signal, as when the speaker is laughing.

In the following, we describe the modalities and the annotation features of our multimodal annotation scheme. As an example, the analysis of emotive labial movements implemented in our annotation scheme is based on a little amount of signs similar to emoticons. We sign two levels of activation using the plus and minus signs. So, annotation values for mouth shape are:

- **o** open lips when the mouth is open;
- **-** closed lips when the mouth is closed;
- **)** corners up e.g. when smiling**; +)** open smile;
- **(** corners down**; +(** corners very down
- **1cornerup** for asymmetric smile;
- **O** protruded, when the lips are rounded.

Similar signals are used to annotate eyebrows shape.

3.1 Cooperation Analysis

The approach we have used to analyze cooperation in dialogue task is mainly based on Bethan Davies model (Bethan Davies, 2006). The basic coded unit is the "move", which means individual linguistic choices to successfully fulfill Map Task. The idea of evaluating utterance choices in relation to task success can be traced back to Anderson and Boyle (1994) who linked utterance choices to the accuracy of the route performed on the map. Bethan Davies extended the meaning of "move" to the goal evaluation, from a narrow set of indicators to a sort of data-driven set. In particular, Bethan Davies stressed some useful points for the computation of collaboration between two communicative partners:

- *social needs of dialogue:* there is a minimum "effort" needed to keep the conversation going. It includes minimal answers like "yes" or "no" and feedbacks. These brief utterances are classified by Bethan Davies (following Traum, 1994) as low effort, as they do not require much planning to the overall dialogue and to the joint task;
- *responsibility of supplying the needs of the communication partner:* to keep an utterance going, one of the speakers can provide follow-ups which take more consideration of the partner's intentions and goals in the task performance. This involves longer utterances, and of course a larger effort;
- *responsibility of maintaining a known track of communication or starting a new one:* there is an effort in considering the actions of a speaker within the context of a particular goal: that is, they mainly deal with situations where a speaker is reacting to the instruction or question offered by the other participant, rather than moving the discourse on another goal. In fact the latter

is perceived as a great effort as it involves reasoning about the task as a whole, beside planning and producing a particular utterance.

Following Traum (1994), speakers tend to engage in lower effort behaviors than higher ones. Thus, if you do not answer to a question, the conversation will end, but you can choose whether or not to query an instruction or offer a suggestion about what to do next. This is reflected in a weighting system where behaviors account for the effort invested and provides a basis for the empirical testing of dialogue principles. The use of this system provides a positive and negative score for each dialogue move. We slightly simplified the Bethan Davies' weighting system and propose a system giving positive and negative weights in an ordinal scale from +2 to -2. We also attribute a weight of 0 for actions which are in the area of "minimum social needs" of dialogue. In Table 1 we report some of the dialogue moves, called cooperation type, and the corresponding cooperation weighting level. There is also a description of different type of moves in terms of Grice's conversational rules breaking or following. Due to the nature of the map task, where giver and a follower have different dialogue roles, we have two slightly different versions of the cooperation annotation scheme. For example "giving instruction" is present only when annotating the giver cooperation. On the other hand "feedback" is present in both annotation schemes. Other communicative collaboration indexes we codify in our coding scheme are the presence or absence of eye contact through gaze direction (to the interlocutor, to the map, unfocused), even in full screen condition, where the two speakers can't see each other. Dialogue turns management (turn giving, turn offering, turn taking, turn yielding, turn concluding, and feedback) has been annotated as well. Video clips have been orthographically transcribed. To do so, we adopted a subset of the conventions applied to the transcription of the speech corpus of the LUNA project corpus annotation (see Rodriguez et al., 2007).

3.2 Coding Procedure and Kappa Scores

Up to now we have annotated 9 emotive tokens of an average length of 100 seconds each. They have been annotated with the coding scheme previously described by 6 annotators. Our coding scheme has been implemented into ANVIL software (Kipp, 2001). A Fleiss' kappa statistic (Fleiss, 1971) has been performed on the annotations. We choose Fleiss' kappa as it is the suitable statistics when chance agreement is calculated on more than two coders. In this case the agreement is expected on the basis of a single distribution reflecting the combined judgments of all coders.

Cooperation level	Cooperation type
-2	**No response to answer:** breaks the maxims of quality, quantity and relevance
-2	**No information add when required:** breaks the maxims of quality, quantity and manner
-2	**No turn giving, no check**: breaks the maxims of quality, quantity and relevance
-1	**Inappropriate reply (no giving info)**: breaks the maxims of quantity and relevance
0	**Giving instruction**: cooperation baseline, task demands
1	**Question answering y/n**: applies the maxims of quality and relevance
1	**Repeating instruction**: applies the maxims of quantity and manner
2	**Question answering y/n + adding info**: applies the maxims of quantity, quality and relevance
2	**Checking the other understands (*ci sei? Capito?*):** applies the maxims of quantity, quality and manner
2	**Spontaneous info/description adding**: applies the maxims of quantity, quality and manner

Table 1: Computing cooperation in our coding scheme (from Bethan Davies, 2006 adapted)

Thus, expected agreement is measured as the overall proportion of items assigned to a category k by all coders n.

Cooperation annotation for giver has a Fleiss' kappa score of 0.835 (p<0.001), while for follower cooperation annotation is 0.829 (p<0.001). Turn management has a Fleiss kappa score of 0.784 (p<0.001). As regard gaze, Fleiss kappa score is 0.788 (p<0.001). Mouth shape annotation has a Fleiss kappa score of 0.816 (p<0.001) and eyebrows shape annotation has a Fleiss kappa of 0.855 (p<0.001). In the last years a large debate on the interpretation of kappa scores has widespread. There is a general lack of consensus on how to interpret those values. Some authors (Allwood et al., 2006) consider as reliable for multimodal annotation kappa values between 0.67 and 0.8. Other authors accept as reliable only scoring rates over 0.8 (Krippendorff, 2004) to allow some generalizations. What is clear is that it seems inappropriate to propose a general cut off point, especially for multimodal annotation where very little literature on kappa agreement has been reported. In this field it seems more necessary that researches report clearly the method they apply (e. g. the number of coders, if they code independently or not, if their coding relies only manually).

Our kappa scores are very high if compared with other multimodal annotation results. This is because we analyze cooperation and emotion with an unambiguous coding scheme. In particular, we do not refer to emotive terms directly. In fact every annotator has his/her own representation of a particular emotion, which could be pretty different from the one of another coder. This representation will represent a problem especially for annotation of blended emotions, which are ambiguous and mixed by nature. As some authors have argued (Colletta et al., 2008) annotation of mental and emotional states is a very demanding task. The analysis of non verbal features requires a different approach if compared with other linguistics tasks as multimodal communication is multichannel (e.g. audiovisual) and has multiple semantic levels (e.g. a facial expression can deeply modify the sense of a sentence, such as in humor or irony).

The final goal of this research is performing a logistic regression on cooperation and emotion display. We will also investigate speakers' role (giver or follower) and screen/no screen conditions role with respect to cooperation. Our predictions are that in case of full screen condition (i. e. the two speakers can't see each other) the cooperation will be lower with respect to short screen condition (i. e. the two speakers can see each other's face) while emotion display will be wider and more intense for full screen condition with respect to short barrier condition. No predictions are made on the speaker role.

4 Conclusions and Future Directions

Cooperative behavior and its relationship with emotions is a topic of great interest in the field of dialogue annotation. Usually emotions achieve a low agreement among raters (see Douglas-Cowie et al., 2005) and surprisingly emotion recognition is higher in a condition of modality deprivation (only acoustic or only visual vs. bimodal).

Neuroscience research on emotion shows that emotion recognition is a process performed firstly by sight, but the awareness of the emotion expressed is mediated by the prefrontal cortex. Moreover a predefined set of emotion labels can influence the perception of facial expression. Therefore we decide to deconstruct each signal without attributing directly an emotive label. We consider promising the implementation in computational coding schemes of neuroscience evidences on transmitting and decoding of emotions. Further researches will implement an experiment on coders' brain activation of to understand if emotion recognition from face is a whole or a part based process.

References

Allwood J., Cerrato L., Jokinen K., Navarretta C., and Paggio P. 2006. A Coding Scheme for the Annotation of Feedback, Turn Management and Sequencing Phenomena. In Martin, J.-C., Kühnlein, P., Paggio, P., Stiefelhagen, R., Pianesi, F. (Eds.) *Multimodal Corpora: From Multimodal Behavior Theories to Usable Models*: 38-42.

Anderson A., Bader M., Bard E., Boyle E., Doherty G. M., Garrod S., Isard S., Kowtko J., McAllister J., Miller J., Sotillo C., Thompson H. S. and Weinert R. 1991. The HCRC Map Task Corpus. *Language and Speech,* 34:351-366

Anderson A. H., and Boyle E. A. 1994. Forms of introduction in dialogues: Their discourse contexts and communicative_consequences. *Language and Cognitive Process* , 9(1):101 - 122

Anderson J. C., Linden W., and Habra M. E. 2005. The importance of examining blood pressure reactivity and recovery in anger provocation research. *International Journal of Psychophysiology* 57(3): 159-163

Argyle M. and Cook M. 1976 *Gaze and mutual gaze*, Cambridge: Cambridge University Press

Bethan Davies L. 2006. Testing Dialogue Principles in Task-Oriented Dialogues: An Exploration of Cooperation, Collaboration, Effort and Risk. In *University of Leeds papers*

Brennan S. E., Chen X., Dickinson C. A., Neider M. A. and Zelinsky J. C. 2008. Coordinating cognition: The costs and benefits of shared gaze during collaborative search. *Cognition* 106(3): 1465-1477

Ekman P. and Friesen WV. 1978. *FACS Facial Action Codind Scheme. A technique for the measurement of facial action*, Palo Alto, CA: Consulting Press

Carletta, J. 2007. Unleashing the killer corpus: experiences in creating the multi-everything AMI Meeting Corpus, *Language Resources and Evaluation,* 41: 181-190

Colletta, J.-M., Kunene, R., Venouil, and A. Tcherkassof, A. 2008. Double Level Analysis of the Multimodal Expressions of Emotions in Human-machine Interaction. In Martin, J.-C., Patrizia, P., Kipp, M., Heylen, D., (Eds.) *Multimodal Corpora: From Models of Natural Interaction to Systems and Applications*, 5-11

Craggs R., and Wood M. 2004. A Categorical Annotation Scheme for Emotion in the Linguistic Content of Dialogue. In *Affective Dialogue Systems*, Elsevier, 89-100

Douglas-Cowie E., Devillers L., Martin J.-C., Cowi R., Savvidou S., Abrilian S., and Cox C. 2005. Multimodal Databases of Everyday Emotion: Facing up to Complexity. In *9th European Conference on Speech Communication and Technology (Interspeech'2005)* Lisbon, Portugal, September 4-8, 813-816

Feldman Barrett L., Lindquist K. A., and Gendron M. 2007. Language as Context for the Perception of Emotion. *Trends in Cognitive Sciences*, 11(8): 327-332.

Fleiss J. L. 1971. Measuring Nominal Scale Agreement among Multiple Coders *Psychological Bulletin* 11(4): 23-34.

Goeleven E., De Raedt R., Leyman L., and Verschuere, B. 2008. The Karolinska Directed Emotional Faces: A validation study, *Cognition and Emotion*, 22:1094 -1118

Kendon A. 1967. Some Functions of Gaze Directions in Social Interaction, *Acta Psychologica* 26(1):1-47

Kipp M., Neff M., and Albrecht I. 2006. An Annotation Scheme for Conversational Gestures: How to economically capture timing and form. In Martin, J.-C., Kühnlein, P., Paggio, P., Stiefelhagen, R., Pianesi, F. (Eds.) *Multimodal Corpora: From Multimodal Behavior Theories to Usable Models*, 24-28

Kipp M. 2001. ANVIL - A Generic Annotation Tool for Multimodal Dialogue. In *Eurospeech 2001* Scandinavia 7th European Conference on Speech Communication and Technology

Krippendorff K. 2004. Reliability in content analysis: Some common misconceptions and recommendations. *Human Communication Research*, 30:411-433

Magno Caldognetto E., Poggi I., Cosi P., Cavicchio F. and Merola G. 2004. Multimodal Score: an Anvil Based Annotation Scheme for Multimodal Audio-Video Analysis. In Martin, J.-C., Os, E.D., Kühnlein, P., Boves, L., Paggio, P., Catizone, R. (eds.) Proceedings of Workshop *Multimodal Corpora: Models Of Human Behavior For The Specification And Evaluation Of Multimodal Input And Output Interfaces.* 29-33

Martin J.-C., Caridakis G., Devillers L., Karpouzis K. and Abrilian S. 2006. Manual Annotation and Automatic Image Processing of Multimodal Emotional Behaviors: Validating the Annotation of TV Interviews. In *Fifth international conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy

Pianesi F., Leonardi C., and Zancanaro M. 2006. Multimodal Annotated Corpora of Consensus Decision Making Meetings. In Martin, J.-C., Kühnlein, P., Paggio, P., Stiefelhagen, R., Pianesi, F. (Eds.) *Multimodal Corpora: From Multimodal Behavior Theories to Usable Models*, 6--9

Poggi I., 2007. *Mind, hands, face and body. A goal and belief view of multimodal communication,* Berlin: Weidler Buchverlag

Reidsma D. Heylen D., and Op den Akker R. 2008. On the Contextual Analysis of Agreement Scores. In Martin, J.-C., Patrizia, P., Kipp, M., Heylen, D., (Eds.) *Multimodal Corpora: From Models of Natural Interaction to Systems and Applications*, 52--55

Rodríguez K., Stefan K. J., Dipper S., Götze M., Poesio M., Riccardi G., and Raymond C., and Wisniewska J., 2007. Standoff Coordination for Multi-Tool Annotation in a Dialogue Corpus. In *Proceedings of the Linguistic Annotation Workshop at the ACL'07* (LAW-07), Prague, Czech Republic.

Smith M. L., Cottrell G. W., Gosselin F., and Schyns P. G. 2005. Transmitting and Decoding Facial Expressions. *Psychological Science* 16(3):184-189

Tassinary L. G. and Cacioppo J. T. 2000. The skeletomotor system: Surface electromyography. In LG Tassinary, GG Berntson, JT Cacioppo (eds) *Handbook of psychophysiology*, New York: Cambridge University Press, 263-299

Traum D. R. 1994. A Computational Theory of Grounding in Natural Language Conversation, PhD Dissertation. urresearch.rochester.edu

Clustering Technique in Multi-Document Personal Name Disambiguation

Chen Chen
Key Laboratory of Computational Linguistics (Peking University),
Ministry of Education, China
chenchen@pku.edu.cn

Hu Junfeng
Key Laboratory of Computational Linguistics (Peking University),
Ministry of Education, China
hujf@pku.edu.cn

Wang Houfeng
Key Laboratory of Computational Linguistics (Peking University),
Ministry of Education, China
wanghf@pku.edu.cn

Abstract

Focusing on multi-document personal name disambiguation, this paper develops an agglomerative clustering approach to resolving this problem. We start from an analysis of pointwise mutual information between feature and the ambiguous name, which brings about a novel weight computing method for feature in clustering. Then a trade-off measure between within-cluster compactness and among-cluster separation is proposed for stopping clustering. After that, we apply a labeling method to find representative feature for each cluster. Finally, experiments are conducted on word-based clustering in Chinese dataset and the result shows a good effect.

1 Introduction

Multi-document named entity co-reference resolution is the process of determining whether an identical name occurring in different texts refers to the same entity in the real world. With the rapid development of multi-document applications like multi-document summarization and information fusion, there is an increasing need for multi-document named entity co-reference resolution. This paper focuses on multi-document personal name disambiguation, which seeks to determine if the same name from different documents refers to the same person.

This paper develops an agglomerative clustering approach to resolving multi-document personal name disambiguation. In order to represent texts better, a novel weight computing method for clustering features is presented. It is based on the pointwise mutual information between the ambiguous name and features. This paper also develops a trade-off point based cluster-stopping measure and a labeling algorithm for each clusters. Finally, experiments are conducted on word-based clustering in Chinese dataset. The dataset contains eleven different personal names with varying-sized datasets, and has 1669 texts in all.

The rest of this paper is organized as follows: in Section 2 we review the related work; Section 3 describes the framework; section 4 introduces our methodologies including feature weight computing with pointwise mutual information, cluster-stopping measure based on trade-off point, and cluster labeling algorithm. These are the main contribution of this paper; Section 5 discusses our experimental result. Finally, the conclusion and suggestions for further extension of the work are given in Section 6.

2 Related Work

Due to the varying ambiguity of personal names in a corpus, existing approaches typically cast it as an unsupervised clustering problem based on vector space model. The main difference among these approaches lies in the features, which are used to create a similarity space. Bagga & Baldwin (1998) first performed within-document co-reference resolution, and then explored features in local context. Mann & Yarowsky (2003) extracted local biographical information as features. Al-Kamha and Embley (2004) clustered search results with feature set including attributes, links and page similarities. Chen and Martin (2007) explored the use of a range of syntactic and semantic features in unsupervised clustering of documents. Song (2007) learned the PLSA and LDA model as feature sets. Ono *et al*. (2008) used mixture features including co-occurrences

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 88–95,
Suntec, Singapore, 4 August 2009.

of named entities, key compound words, and topic information. Previous works usually focus on feature identification and feature selection. The method to assign appropriate weight to each feature has not been discussed widely.

A major challenge in clustering analysis is determining the number of ‘clusters’. Therefore, clustering based approaches to this problem still require estimating the number of clusters. In Hierarchy clustering, it equates to determine the stopping step of clustering. The measure to find the “knee” in the criterion function curve is a well known cluster-stopping measure. Pedersen and Kulkarni had studied this problem (Pedersen and Kulkarni, 2006). They developed cluster-stopping measures named PK1, PK2, PK3, and presented the Adapted Gap Statistics.

After estimating the number of ‘clusters’, we obtain the clustering result. In order to label the ‘clusters’, the method that finding representative features for each ‘cluster’ is needed. For example, the captain John Smith can be labeled as captain. Pedersen and Kulkarni (2006) selected the top N non-stopping word features from texts grouped in a cluster as label.

3 Framework

On the assumption of “one person per document” (i.e. all mentions of an ambiguous personal name in one document refer to the same personal entity), the task of disambiguating personal name in text set intends to partition the set into subsets, where each subset refer to one particular entity.

Suppose the set of texts containing the ambiguous name is denoted by $D= \{d_1,d_2,\ldots,d_n\}$, and d_i ($0<i<n+1$) stands for one text. The entities with the ambiguous name are denoted by a set $E= \{e_1,e_2,\ldots,e_m\}$, where the number of entities ‘*m*’ is unknown. The ambiguous name in each text d_i indicates only one entity e_k. The aim of the work is to map an ambiguous name appearing in each text to an entity. Therefore, those texts indicating the same entity need to be clustered together.

In determining whether a personal name refers to a specific entity, the personal information, social network information and related topics play important roles, all of which are expressed by words in texts,. Extracting words as features, this paper applies an agglomerative clustering approach to resolving name co-reference. The framework of our approach consists of the following seven main steps:

Step 1: Pre-process each text with Chinese word segmentation tool;

Step 2: Extract words as features from the set of texts D;.

Step 3: Represent texts $d_1,\ldots,d_n$ by features vectors;

Step 4: Calculate similarity between texts;

Step 5: Cluster the set D step by step until only one cluster exists;

Step 6: Estimate the number of entities in accordance with cluster-stopping measure;

Step 7: Assign each cluster a discriminating label.

This paper focuses on the *Step 4*, *Step 6* and *Step 7*, i.e., feature weight computing method, clustering stopping measure and cluster labeling method. They will be described in the next section in detail.

Step1 and *Step3* are simple, and there is no further description here. In *Step 2*, we use co-occurrence words of the ambiguous name in texts as features. In the process of agglomerative clustering (see *Step 5*), each text is viewed as one cluster at first, and the most similar two clusters are merged together as a new cluster at each round. After replacing the former two clusters with the new one, we use average linked method to update similarity between clusters.

4 Methodology

4.1 Feature weight

Each text is represented as a feature vector, and each item of the vector represents the weight value for corresponding feature in the text. Since our approach is completely unsupervised we cannot use supervised methods to select significant features. Since the weight of feature will be adjusted well instead of feature selection, all words in set *D* are used as feature in our approach.

The problem of computing feature weight is involved in both text clustering and text classification. By comparing the supervised text classification and unsupervised text clustering, we find that the former one has a better performance owing to the selection of features and the computing method of feature weight. Firstly, in the application of supervised text classification, features can be selected by many methods, such as, Mutual Information (MI) and Expected Cross Entropy (ECE) feature selection methods. Secondly, model training methods, such as SVM model, are generally adopted by programs when to find the optimal feature weight. There is no training data for unsupervised tasks, so above-mentioned methods are unsuitable for text clustering.

In addition, we find that the text clustering for personal name disambiguation is different from common text clustering. System can easily judge whether a text contains the ambiguous personal name or not. Thus the whole collection of texts can be easily divided into two classes: texts with or without the name. As a result, we can easily calculate the pointwise mutual information between feature words and the personal name. To a certain extent, it represents the correlative degree between feature words and the underlying entity corresponding to the personal name.

For these reasons, our feature weight computing method calculates the pointwise mutual information between personal name and feature word. And the value of pointwise mutual information will be used to expresse feature word's weight by combining the feature's *tf (*the abbreviation for term-frequency*)* in text and *idf (*the abbreviation for inverse document frequency*)* in dataset. The formula of feature weight computing proposed in this paper is as below, and it is need both texts containing and not containing the ambiguous personal name to form dataset *D*. For each t_k in d_i that contains *name,* its *mi_weight* is computed as follow:

$$\begin{aligned}&\text{mi\_weight}(t_k, name, d_i) = (1+\log(tf(t_k, d_i)))\\&\times\log(1+\text{MI}(t_k, name))\times\log(|D|/df(t_k))\end{aligned} \tag{1}$$

And

$$\begin{aligned}\text{MI}(t_k, name) &= \frac{p(name, t_k)}{p(name)\times p(t_k)}\\&= \frac{df(name, t_k)/|D|}{df(name)\times df(t_k)/|D|^2}\\&= \frac{df(name, t_k)\times|D|}{df(name)\times df(t_k)}\end{aligned} \tag{2}$$

Where t_k is a feature; *name* is the ambiguous name; d_i is the i^th^ text in dataset; $tf(t_k,d_i)$ represents term frequency of feature t_k in text d_i; $df(t_k)$, $df(name)$ is the number of the texts containing t_k or *name* in dataset *D* respectively; $df(t_k,name)$ is the number of texts containing both t_k and *name*; |*D*| is the number of all the texts.

Formula (2) can be comprehended as: if word t_k occurs much more times in texts containing the ambiguous name than in texts not containing the name, it must have some information about the name.

A widely used approach for computing feature weight is *tf*idf* scheme as formula (3) (Salton and Buckley. 1998), which only uses the texts containing the ambiguous name. We denote it by *old_weight* . For each t_k in d_i containing *name,* the *old_weight* is computed as follow:

$$\begin{aligned}&\text{old\_weight}(t_k, name, d_i)\\&= (1+\log(tf(t_k, d_i)))\\&\times\log(df(name)/df(t_k, name))\end{aligned} \tag{3}$$

The first term on the right side is *tf*, and the second term is *idf*. If the *idf* scheme is computed in the whole dataset *D* for reducing noise, the weight computing formula can be expressed as follow, and is denoted by *imp_weight*:

$$\begin{aligned}&\text{imp\_weight}(t_k, d_i)\\&= (1+\log(tf(t_k, d_i)))\times\log(|\text{D}|/df(t_k))\end{aligned} \tag{4}$$

Before clustering, the similarity between texts is computed by cosine value of the angle between vectors (such as $\mathbf{d_x}$, $\mathbf{d_y}$ in formula (5)):

$$\cos(\mathbf{d}_x, \mathbf{d}_y) = \frac{\mathbf{d}_x\cdot\mathbf{d}_y}{\|\mathbf{d}_x\|\cdot\|\mathbf{d}_y\|} \tag{5}$$

Each item of the vector (i.e. $\mathbf{d_x}$, $\mathbf{d_y}$) represents the weight value for corresponding feature in the text.

4.2 Cluster-stopping measure

The process of clustering will produce *n* cluster results, one for each step. Independent of clustering algorithm, the cluster stopping measure should choose the cluster results which can represent the structure of data.

A fundamental and difficult problem in cluster analysis is to measure the structure of clustering result. The geometric structure is a representative method. It defines that a "good" clustering results should make data points from one cluster "compact", while data points from different cluster are "separate" as far as possible. The indicators should quantify the "compactness" and "separation" for clusters, and combine both. In the study of cluster stopping measures by Pedersen and Kulkarni (2006), the criterion functions defines text similarity based on cosine value of the angle between vectors. Their cluster-stopping measures focused on finding the 'knee' of criterion function.

Our cluster-stopping measure is also based on the geometric structure of dataset. The measure aims to find the trade-off point between within-cluster compactness and among-cluster separation. Both the within-cluster compactness (Internal critical function) and among-cluster separation (External critical function) are defined by Euclidean distance. The hybrid critical function (Hybrid critical function) combines internal and external criterion functions.

Suppose that the given dataset contains *N* references, which are denoted as: $d_1,d_2,\ldots,d_N$; the data have been repeatedly clustered into *k* clusters, where *k=N,…,1*; and clusters are denoted as C_r, *r=1,…k*; and the number of references in each cluster is $\mathrm{n_r}$, so $n_r=|C_r|$. We introduce *Incrf* (Internal critical function), *Excrf* (External critical function) and *Hycrf* (Hybrid critical function) to measure it as follows.

$$\mathrm{Incrf}(k)=\sum_{i=1}^{k}\sum\nolimits_{\mathbf{d_x},\mathbf{d_y}\in\mathbf{C_i}}\left\|\mathbf{d_x}-\mathbf{d_y}\right\|^{\mathbf{2}} \quad (6)$$

$$\mathrm{Excrf}(k)=\sum_{i=1}^{k}\sum_{j=1,j\neq i}^{k}\frac{1}{n_i n_j}\sum\nolimits_{\mathbf{d_x}\in\mathbf{C_i},\mathbf{d_y}\in\mathbf{C_j}}\left\|\mathbf{d_x}-\mathbf{d_y}\right\|^{\mathbf{2}}$$

(7)

$$\mathrm{Hycrf}(k)=\frac{1}{\mathrm{M}}\times(\mathrm{Incrf}(k)+\mathrm{Excrf}(k)) \quad (8)$$

Where M=Incrf(1)=Excrf(N)

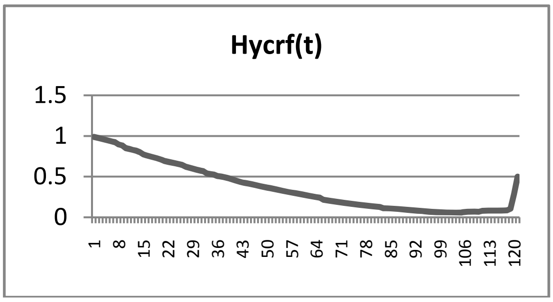


Figure 1 *Hycrf* vs. *t* (N-k)

Chen proved the existence of the minimum value between (0,1) in Hycrf(k) (see Chen *et al.* 2008). The *Hycrf* value in a typical Hycrf(t) curve is shown as Figure 1, where t=N-k.

Function *Hycrf* based on *Incrf* and *Excrf* is used as the Hybrid criterion function. The *Hycrf* curve will rise sharply after the minimum, indicating that the cluster of several optimal partitions' subsets will lead to drastic drop in cluster quality. Thus cluster partition can be determined. Using the attributes of the *Hycrf(k)* curve, we put forward a new cluster-stopping measure named trade-off point based cluster-stopping measure (TO_CSM).

$$\mathrm{TO\_CSM}(k)=\frac{1}{\mathrm{Hycrf}(k+1)}\times\frac{\mathrm{Hycrf}(k)}{\mathrm{Hycrf}(k+1)}$$

(9)

Trade-off point based cluster-stopping measure (*TO_CSM*) selects the *k* value which maximizes *TO_CSM(k)*, and indicates the number of cluster. The first term on the right side of formula (9) is used to minimize the value of *Hycrf(k)*, and the second one is used to find the 'knee' rising sharply.

4.3 Labeling

Once the clusters are created, we label each entity to represent the underlying entity with some important information. A label is represented as a list of feature words, which summarize the information about cluster's underlying entity.

The algorithm is outlined as follows: after clustering *N* references into *m* clusters, for each cluster $\mathrm{C_k}$ in $\{\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_m}\}$, we calculate the score of each feature for $\mathrm{C_k}$ and choose features as the label of $\mathrm{C_k}$ whose scores rank top N. In particular, the score caculated in this paper is different from Pedersen and Kulkarni's (2006). We combine pointwise mutual information computing method with term frequency in cluster to compute the score.

The formula of feature scoring for labeling is shown as follows:

$$\begin{aligned}\mathrm{Score}(t_k,C_i)&=\mathrm{MI}(t_k,name)\times\mathrm{MI}_{\mathrm{name}}(t_k,C_i)\\&\times(1+\log(tf(t_k,C_i)))\end{aligned}$$

(10)

The calculation of *MI(*t_k*,name)* is shown as formula (2) in subsection 4.1. *tf(*t_k*,*C_i*)* represents the total occurrence frequency of feature t_k in cluster C_i . The $MI_{name}(t_k,C_i)$ is computed as formula (11):

$$\begin{aligned}\mathrm{MI}_{name}(t_k,\mathrm{C}_i)&=\frac{p(t_k,C_i)}{p(t_k)\times p(C_i)}\\&=\frac{df(t_k,C_i)/|D|}{df(t_k)\times df(C_i)/|D|^2}\\&=\frac{df(t_k,C_i)\times|D|}{df(t_k)\times df(C_i)}\end{aligned}$$

(11)

In formula (10), the weight of stopping words can be reduced by the first item. The second item can increase the weight of words with high distinguishing ability for a certain ambiguous name. The third item of formula (10) gives higher scores to features whose frequency are higher.

5 Experiment

5.1 Data

The dataset is from WWW, and contains 1,669 texts with eleven real ambiguous personal names. Such raw texts containing ambiguous names are collected via search engine[1], and most of them are news. The eleven person-names are, "刘易斯 Liu-Yi-si ‘Lewis’", "刘淑珍 Liu-Shu-zhen ", "李强 Li-Qiang", "李娜 Li-Na", "李桂英 Li-Gui-ying", "米歇尔 Mi-xie-er ‘Michelle’", "玛丽 Ma-Li ‘Mary’", "约翰逊 Yue-han-xun ‘Johnson’", "王涛 Wang-Tao", "王刚 Wang-Gang", " 陈志强 Chen-Zhi-qiang". Names like “Michelle”, “Johnson” are transliterated from English to Chinese, while names like “Liu –Shu-zhen”, “Chen-Zhi-qiang” are original Chinese personal names. Some of these names only have a few persons, while others have more persons.

Table 1 shows our data set. “#text” presents the number of texts with the personal name. “#per” presents the number of entities with the personal name in text dataset. “#max” presents the maximum of texts for an entity with the personal name, and “#min” presents the minimum.

	#text	**#per**	**#max**	**#min**
Lewis	120	6	25	10
Liu-Shu-zhen	149	15	28	3
Li-Qiang	122	7	25	9
Li-Na	149	5	39	21
Li-Gui-ying	150	7	30	10
Michelle	144	7	25	12
Mary	127	7	35	10
Johnson	279	19	26	1
Wang-Gang	125	18	26	1
Wang-Tao	182	10	38	5
Chen-Zhi-qiang	122	4	52	13

Table 1 Statistics of the test dataset

We first convert all the downloaded documents into plain text format to facilitate the test process, and pre-process them by using the segmentation toolkit ICTCLAS[2].

In testing and evaluating, we adopt B-Cubed definition for *Precision*, *Recall* and *F-Measure* as indicators (Bagga, Amit and Baldwin. 1998). *F-Measure* is the harmonic mean of *Precision* and *Recall*.

The definitions are presented as below:

$$precision = \frac{1}{N}\sum\nolimits_{d \in D} precision_d \tag{12}$$

$$recall = \frac{1}{N}\sum\nolimits_{d \in D} recall_d \tag{13}$$

$$F - measure = \frac{2 \times precision \times recall}{precision + recall} \tag{14}$$

where *precision*$_d$ is the precision for a text *d*. Suppose the text *d* is in subset *A*, *precision*$_d$ is the percentage of texts in *A* which indicates the same entity as *d*. *Recall*$_d$ is the recall ratio for a text *d*. *Recall*$_d$ is the ratio of number of texts which indicates the same entity as *d* in *A* to that in corpus *D*. *n* = | *D* |, *D* refers to a collection of texts containing a particular name (such as Wang Tao, e.g. a set of 200 texts, *n* = 200). Subset *A* is a set formed after clustering (text included in class), and *d* refers to a certain text that containing "Wang Tao".

5.2 Result

All the 1669 texts in the dataset are employed during experiment. Each personal name disambiguation process only clusters the texts containing the ambiguous name. After pre-processing, in order to verify the *mi_weight* method for feature weight computing, all the words in texts are used as features.

Using formula (1), (3) and (4) as feature weight computing formula, we can get the evaluation of cluster result shown as table 2. In this step, cluster-stopping measure is not used. Instead, the highest F-measure during clustering is highlighted to represent the efficiency of the feature weight computing method.

Further more, we carry out the experiment on the trade-off point based cluster-stopping measure, and compare its cluster result with highest F-measure and cluster result determined by cluster-stopping measure PK3 proposed by Pedersen and Kulkarni’s. Based on the experiment in Table 2, a structure tree is constructed in the clustering process. Cluster-stopping measures are used to determine where to stop cutting the dendrogram. As shown in Table 3, the TO-CMS method predicts the optimal results of four names in eleven, while PK3 method predicts the optimal result of one name, which are marked in a bold type.

[1] April.2008

[2] http://ictclas.org/

	old_weight			imp_weight			mi_weight		
	#pre	**#rec**	**#F**	**#pre**	**#rec**	**#F**	**#pre**	**#rec**	**#F**
Lewis	0.9488	0.8668.	0.9059	1	1	1	1	1	**1**
Liu-Shu-zhen	0.8004	0.7381	0.7680	0.8409	0.8004	0.8201	0.9217	0.7940	**0.8531**
Li-Qiang	0.8057	0.6886	0.7426	0.9412	0.7968	**0.8630**	0.8962	0.8208	0.8569
Li-Na	0.9487	0.7719	0.8512	0.9870	0.8865	0.9340	0.9870	0.9870	**0.9870**
Li-Gui-ying	0.8871	0.9124	0.8996	0.9879	0.8938	**0.9385**	0.9778	0.8813	0.9271
Michelle	0.9769	0.7205	0.8293	0.9549	0.8146	0.8792	0.9672	0.9498	**0.9584**
Mary	0.9520	0.6828	0.7953	1	0.9290	**0.9632**	1	0.9001	0.9474
Johnson	0.9620	0.8120	0.8807	0.9573	0.8083	0.8765	0.9593	0.8595	**0.9067**
Wang-Gang	0.8130	0.8171	0.8150	0.7804	0.9326	0.8498	0.8143	0.9185	**0.8633**
Wang-Tao	1	0.9323	0.9650	0.9573	0.9485	0.9529	0.9897	0.9768	**0.9832**
Chen-Zhi-qiang	0.9732	0.8401	0.9017	0.9891	0.9403	0.9641	0.9891	0.9564	**0.9725**
Average	0.9153	0.7916	0.8504	0.9451	0.8864	0.9128	0.9548	0.9131	**0.9323**

Table 2 comparison of feature weight computing method (highest F-measure)

	Optimal			TO-CMS			PK3		
	#pre	**#rec**	**#F**	**#pre**	**#rec**	**#F**	**#pre**	**#rec**	**#F**
Lewis	1	1	1	**1**	**1**	**1**	0.8575	1	0.9233
Liu-Shuzhen	0.9217	0.7940	0.8531	0.8466	0.8433	0.8450	0.5451	0.9503	0.6928
Li-Qiang	0.8962	0.8208	0.8569	**0.8962**	**0.8208**	**0.8569**	0.7897	0.9335	0.8556
Li-Na	0.9870	0.9870	0.9870	**0.9870**	**0.9870**	**0.9870**	0.9870	0.9016	0.9424
Li-Gui-ying	0.9778	0.8813	0.9271	**0.9778**	**0.8813**	**0.9271**	0.8750	0.9427	0.9076
Michelle	0.9672	0.9498	0.9584	0.9482	0.9498	0.9490	**0.9672**	**0.9498**	**0.9584**
Mary	1	0.9001	0.9474	0.8545	0.9410	0.8957	0.8698	0.9410	0.9040
Johnson	0.9593	0.8595	0.9067	0.9524	0.8648	0.9066	0.2423	0.9802	0.3885
Wang-Gang	0.8143	0.9185	0.8633	0.9255	0.7102	0.8036	0.5198	0.9550	0.6732
Wang-Tao	0.9897	0.9768	0.9832	0.8594	0.9767	0.9144	0.9700	0.9768	0.9734
Chen-Zhi-qiang	0.9891	0.9564	0.9725	0.8498	1	0.9188	0.8499	1	0.9188
Average	0.9548	0.9131	0.9323	0.9179	0.9068	0.9095	0.7703	0.9574	0.8307

Table 3 comparison of cluster-stopping measures' performance

name	Entity	Created Labels
Lewis	Person-1	巴比特(Babbitt),辛克莱·刘易斯(Sinclair Lewis),阿罗史密斯(Arrow smith),文学奖(Literature Prize),德莱赛(Dresser),豪威尔斯(Howells),瑞典文学院(Swedish Academy),舍伍德·安德森(Sherwood Anderson),埃尔默·甘特利(Elmer Gan Hartley),大街(street),受奖(award),美国文学艺术协会(American Literature and Arts Association)
	Person-2	美国银行(Bank of America),美洲银行(Bank of America),银行(bank),投资者(investors),信用卡(credit card),中行(Bank of China),花旗(Citibank),并购(mergers and acquisitions),建行(Construction Bank),执行官(executive officer),银行业(banking),股价(stock),肯·刘易斯(Ken Lewis)
	Person-3	单曲(Single),丽昂娜(Liana),专辑(album),丽安娜(Liana),丽安娜·刘易斯(Liana Lewis),利昂娜(Liana),空降(airborne),销量(sales),音乐奖(Music Awards),玛丽亚·凯莉(Maria Kelly),榜(List),处子(debut)、
	Person-4	卡尔·刘易斯(Carl Lewis),跳远(long jump),卡尔(Carl),欧文斯(Owens),田径(track and field),伯勒尔(Burrell),美国奥委会(the U.S. Olympic Committee),短跑(sprint),泰勒兹(Taylors),贝尔格莱德(Belgrade),维德·埃克森(Verde Exxon),埃克森(Exxon)

	Person-5	泰森(Tyson),拳王(King of Boxer),击倒(knock down),重量级(heavyweight),唐金(Don King),拳击(boxing),腰带(belt),拳手(Boxing),拳(fist),回合(bout),拳台(Ring),WBC
	Person-6	丹尼尔(Daniel),戴·刘易斯(Day Lewis),血色(Blood),丹尼尔·戴·刘易斯(Daniel Day Lewis),黑金(There Will Be Blood),左脚(left crus),影帝(movie king),纽约影评人协会(New York Film Critics Circles),小金人(the Gold Oscar statues),主角奖(Best Actor in a Leading Role),奥斯卡(Oscar),未血绸缪(There Will Be Blood)

Table 4 Labels for “Lewis” clusters

On the basis of text clustering result that obtained from the Trade-off based cluster-stopping measure experiment in Table 3, we try our labelling method mentioned in subsection 4.3. For each cluster, we choose 12 words with highest score as its label. The experiment result demonstrates that the created label is able to represent the category. Take name “刘易斯 Liu-Yi-si ‘Lewis’” for example, the labeling result shown as Table 4.

5.3 Discussion

From the test result in table 2, we find that our feature weight computing method can improve the Chinese personal name clustering disambiguation performance effectively. For each personal name in test dataset, the performance is improved obviously. The average value of optimal F-measures for eleven names rises from 85.04% to 91.28% by using the whole dataset *D* for calculated *idf*, and rises from 91.28% to 93.23% by using *mi_weight*. Therefore, in the application of Chinese text clustering with constraints, we can compute pointwise mutual information between constraints and feature, and it can be merged with feature weight value to improve the clustering performance.

We can see from table 3 that trade-off point based cluster-stopping measure (*TO_CSM*) performs much better than *PK3*. According to the experimental results, *PK3* measure is not that robust. The optimal number of clusters can be determined for certain data. However, we found that it did not apply to all cases. For example, it obtains the optimal estimation result for data “Michelle”, as for “Liu Shuzhen”, “Wang Gang” and “Johnson”, the results are extremely bad. The better result is achieved by using *TO_CSM* measure, and the selected results are closer to the optimal value. The *PK3* measure uses the mean and the standard deviation to deduce, and its processes are more complicated than *TO_CSM*’s.

Our cluster labeling method computes the features’ score with formula (10). From the labeling results sample shown in Table 4, we can see that all of the labels are representative. Most of them are person and organizations’ name, and the rest are key compound words. Therefore, when the clustering performance is good, the quality of cluster labels created by our method is also good.

6 Future Work

This paper developed a clustering algorithm of multi-document personal name disambiguation, and put forward a novel feature weight computing method for vector space model. This method computes weight with the pointwise mutual information between the personal name and feature. We also study a hybrid criterion function based on trade-off point and put forward the trade-off point cluster-stopping measure. At last, we experiment on our score computing method for cluster labeling.

Unsupervised personal name disambiguation techniques can be extended to address the problem of unsupervised Entity Resolution and unsupervised word sense discrimination. We will attempt to apply the feature weight computing method to these fields.

One of the main directions of our future work will be how to improve the performance of personal name disambiguation. Computing weight based on a window around names may be helpful. Moreover, word-based text features haven’t solved two difficult problems of natural language problems: Synonym and Polysemy, which seriously affect the precision and efficiency of clustering algorithms. Text representation based on concept and topic may solve the problem.

Acknowledgments

This research is supported by National Natural Science Foundation of Chinese (No.60675035) and Beijing Natural Science Foundation (No.4072012)

References

Al-Kamha. R. and D. W. Embley. 2004. Grouping search-engine returned citations for person-name queries. In *Proceedings of WIDM'04*, 96-103, Washington, DC, USA.

Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of 17th International Conference on Computational Linguistics,* 79–85.

Bagga, Amit and B. Baldwin. 1998. A*lgorithms for scoring co-reference chains.* In Proceedings of the First International Conference on Language Resources and Evaluation Workshop on Linguistic co-reference.

Chen Ying and James Martin. 2007. Towards Robust Unsupervised Personal Name Disambiguation, *EMNLP 2007*.

Chen Lifei, Jiang Qingshan, and Wang Shengrui. 2008. A Hierarchical Method for Determining the Number of Clusters. *Journal of Software*, 19(1). [in Chinese]

Chung Heong Gooi and James Allan. 2004. Cross-document co-reference on a large scale corpus. In S. Dumais, D. Marcu, and S. Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, 9–16, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

Gao Huixian. *Applied Multivariate Statistical Analysis*. Peking Univ. Press. 2004.

G. Salton and C. Buckley. 1988. *Term-weighting approaches in automatic text retrieval*. Information Processing and Management,

Kulkarni Anagha and Ted Pedersen. 2006. How Many Different "John Smiths", and Who are They? In *Proceedings of the Student Abstract and Poster Session of the 21st National Conference on Artificial Intelligence, Boston, Massachusetts*.

Mann G. and D. Yarowsky. 2003. Unsupervised personal name disambiguation. In W. Daelemans and M. Osborne, editors, *Proceedings of CoNLL-2003*, 33–40, Edmonton, Canada.

Niu Cheng, Wei Li, and Rohini K. Srihari. 2004. Weakly Supervised Learning for Cross-document Person Name Disambiguation Supported by Information Extraction. In *Proceedings of ACL 2004*.

Ono. Shingo, Issei Sato, Minoru Yoshida, and Hiroshi Nakagawa2. 2008. Person Name Disambiguation in Web Pages Using Social Network, Compound Words and Latent Topics. T. Washio et al. (Eds.): *PAKDD 2008, LNAI 5012*, 260–271.

Song Yang, Jian Huang, Isaac G. Councill, Jia Li, and C. Lee Giles. 2007. Efficient Topic-based Unsupervised Name Disambiguation. *JCDL'07*, June 18–23, 2007, Vancouver, British Columbia, Canada.

Ted Pedersen and Kulkarni Anagha. 2006. Automatic Cluster Stopping with Criterion Functions and the Gap Statistic. In *Proceedings of the Demonstration Session of the Human Language Technology Conference and the Sixth Annual Meeting of the North American Chapter of the Association for Computational Linguistic,* New York City, NY.

Creating a Gold Standard for Sentence Clustering in Multi-Document Summarization

Johanna Geiss
University of Cambridge
Computer Laboratory
15 JJ Thomson Avenue
Cambridge, CB3 0FD, UK
johanna.geiss@cl.cam.ac.uk

Abstract

Sentence Clustering is often used as a first step in Multi-Document Summarization (MDS) to find redundant information. All the same there is no gold standard available. This paper describes the creation of a gold standard for sentence clustering from DUC document sets. The procedure of building the gold standard and the guidelines which were given to six human judges are described. The most widely used and promising evaluation measures are presented and discussed.

1 Introduction

The increasing amount of (online) information and the growing number of news websites lead to a debilitating amount of redundant information. Different newswires publish different reports about the same event resulting in information overlap. Multi-Document Summarization (MDS) can help to reduce the amount of documents a user has to read to keep informed. In contrast to single document summarization information overlap is one of the biggest challenges to MDS systems. While repeated information is a good evidence of importance, this information should be included in a summary only once in order to avoid a repetitive summary. Sentence clustering has therefore often been used as an early step in MDS (Hatzivassiloglou et al., 2001; Marcu and Gerber, 2001; Radev et al., 2000). In sentence clustering semantically similar sentences are grouped together. Sentences within a cluster overlap in information, but they do not have to be identical in meaning. In contrast to paraphrases sentences in a cluster do not have to cover the same amount of information. One sentence represents one cluster in the summary. Either a sentences from the cluster is selected (Aliguliyev, 2006) or a new sentence is regenerated from all/some sentences in a cluster (Barzilay and McKeown, 2005). Usually the quality of the sentence clusters are only evaluated indirectly by judging the quality of the generated summary. There is still no standard evaluation method for summarization and no consensus in the summarization community how to evaluate a summary. The methods at hand are either superficial or time and resource consuming and not easily repeatable. Another argument against indirect evaluation of clustering is that troubleshooting becomes more difficult. If a poor summary was created it is not clear which component e.g. information extraction through clustering or summary generation (using for example language regeneration) is responsible for the lack of quality.

However there is no gold standard for sentence clustering available to which the output of a clustering systems can be compared. Another challenge is the evaluation of sentence clusters. There are a lot of evaluation methods available. Each of them focus on different properties of a set of clusters. We will discuss and evaluate the most widely used and most promising measures. In this paper the main focus is on the development of a gold standard for sentence clustering using DUC clusters. The guidelines and rules that were given to the human annotators are described and the interjudge agreement is evaluated.

2 Related Work

Sentence Clustering is used for different application in NLP. Radev et al. (2000) use it in their MDS system MEAD. The centroids of the clusters are used to create a summary. Only the summary is evaluated, not the sentence clusters. The same applies to Wang et al. (2008). They use symmetric matrix factorisation to group similar sentences together and test their system on DUC2005 and DUC2006 data set, but do not evaluate the clusterings. However Zha (2002) created a gold stan-

Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, pages 96–104,
Suntec, Singapore, 4 August 2009. ©2009 ACL and AFNLP

dard relying on the section structure of web pages and news articles. In this gold standard the section numbers are assumed to give the true cluster label for a sentence. In this approach only sentences within the same document and even within the same paragraph are clustered together whereas our approach is to find similar information between documents.

A gold standard for event identification was built by Naughton (2007). Ten annotators tagged events in a sentence. Each sentence could be assigned more than one event number. In our approach a sentence can only belong to one cluster.

For the evaluation of SIMFINDER Hatzivassiloglou et al. (2001) created a set of 10.535 manually marked pairs of paragraphs. Two human annotator were asked to judge if the paragraphs contained 'common information'. They were given the guideline that only paragraphs that described the same object in the same way or in which the same object was acting the same are to be considered similar. They found significant disagreement between the judges but the annotators were able to resolve their differences. Here the problem is that only pairs of paragraphs are annotated whereas we focus on whole sentences and create not pairs but clusters of similar sentences.

3 Data Set for Clustering

The data used for the creation of the gold standard was taken from the Document Understanding Conference (DUC)[1] document sets. These document clusters were designed for the DUC tasks which range from single-/multi-document summarization to update summaries, where it is assumed that the reader has already read earlier articles about an event and requires only an update of the newer development. Since DUC has moved to TAC in 2008 they focus on the update task. In this paper only clusters designed for the general multi-document summarization task are used.

Our clustering data set consists of four sentence sets. They were created from the document sets d073b (DUC 2002), D0712C (DUC 2007), D0617H (DUC 2006) and d102a (DUC 2003). Especially the newer document clusters e.g. from DUC 2006 and 2007 contain a lot of documents. In order to build good sentence clusters the judges have to compare each sentence to each other sentence and maintain an overview of the topics within the documents. Because of human cognitive limitations the number of documents and sentences have to be reduced. We defined a set of constraints for a sentence set: (i) from one set, (ii) a sentence set should consist of 150 – 200 sentences[2]. To obtain sentence sets that comply with these requirements we designed an algorithm that takes the number of documents in a DUC set, the date of publishing, the number of documents published on the same day and the number of sentences in a document into account. If a document set includes articles published on the same day they were given preference. Furthermore shorter documents (in terms of number of sentences) were favoured. The properties of the resulting sentence sets are listed in table 1. The documents in a set were ordered by date and split into sentences using the sentence boundary detector from RASP (Briscoe et al., 2006).

name	DUC	DUC id	docs	sen
Volcano	2002	D073b	5	162
Rushdie	2007	D0712C	15	103
EgyptAir	2006	D0617H	9	191
Schulz	2003	d102a	5	248

Table 1: Properties of sentence sets

4 Creation of the Gold Standard

Each sentence set was manually clustered by at least three judges. In total there were six judges which were all volunteers. They are all second-language speakers of English and hold at least a Master's degree. Three of them (Judge_A, Judge_J and Judge_O) have a background in computational linguistics. The judges were given a task description and a list of guidelines. They were only using the guidelines given and worked independently. They did not confer with each other or the author. Table 2 gives details about the set of clusters each judge created.

4.1 Guidelines

The following guidelines were given to the judges:

1. Each cluster should contain only one topic.
2. In an ideal cluster the sentences are very similar.

[1] DUC has now moved to the Text Analysis Conference (TAC)

[2] If a DUC set contains only 5 documents all of them are used to create the sentence set, even if that results in more than 200 sentences. If the DUC set contains more than 15 documents, only 15 documents are used for clustering even if the number of 150 sentences is not reached.

judge	Rushdie			Volcano			EgyptAir			Schulz		
	s	c	s/c	s	c	s/c	s	c	s/c	s	c	s/c
Judge_A	70	15	4.6	92	30	3	85	28	3	54	16	3.4
Judge_B	41	10	4.1	57	21	2.7	44	15	2.9	38	11	3.5
Judge_D				46	16	2.9						
Judge_H	74	14	5.3				75	19	3.9			
Judge_J										120	7	17.1
Judge_O							53	20	2.6			

Table 2: Details of manual clusterings: *s* number of sentences in a set, *c* number of clusters, *s/c* average number of sentences in a cluster

3. The information in one cluster should come from as many different documents as possible. The more different sources the better. Clusters of sentences from only one document are not allowed.
4. There must be at least two sentences in a cluster, and more than two if possible.
5. Differences in numbers in the same cluster are allowed (e.g. vagueness in numbers (300,000 - 350,000), update (two killed - four dead))
6. Break off very similar sentences from one cluster into their own subcluster, if you feel the cluster is not homogeneous.
7. Do not use too much inference.
8. Partial overlap – If a sentence has parts that fit in two clusters, put the sentence in the more important cluster.
9. Generalisation is allowed, as long as the sentences are about the same person, fact or event.

The guidelines were designed by the author and her supervisor – Dr Simone Teufel. The starting point was a single DUC document set which was clustered by the author and her supervisor with the task in mind to find clusters of sentences that represent the main topics in the documents. The minimal constraint was that each cluster is specific and general enough to be described in one sentence (see rule 1 and 2). By looking at the differences between the two manual clustering and reviewing the reasons for the differences the other rules were generated and tested on another sentence set.

One rule that emerged early says that a topic can only be included in the summary of a document set if it appears in more than one document (rule 3). From our understanding of MDS and our definition of importance only sentences that depict a topic which is present in more than one source document can be summary worthy. From this it follows that clusters must contain at least two sentences which come from different documents. Sentences that are not in any cluster of at least two are considered irrelevant for the MDS task (rule 4). We defined a spectrum of similarity. In an ideal cluster the sentences would be very similar, almost paraphrases. For our task sentences that are not paraphrases can be in the same cluster (see rule 5, 8, 9). In general there are several constraints that pull against each other. The judges have to find the best compromise.

We also gave the judges a recommended procedure:

1. Read all documents. Start clustering from the first sentence in the list. Put every sentence that you think will attract other sentences into an initial cluster. If you feel, that you will not find any similar sentences to a sentence, put it immediately aside. Continue clustering and build up the clusters while you go through the list of sentences.
2. You can rearrange your clusters at any point.
3. When you are finished with clustering check that all important information from the documents is covered by your clusters. If you feel that a very important topic is not expressed in your clusters, look for evidence for that information in the text, even in secondary parts of a sentence.
4. Go through your sentences which do not belong to any cluster and check if you can find a suitable cluster.
5. Do a quality check and make sure that you wrote down a sentence for each cluster and that the sentences in a cluster are from more than one document.
6. Rank the clusters by importance.

4.2 Differences in manual clusterings

Each judge clustered the sentence sets differently. No two judges came up with the same separation into clusters or the same amount of irrelevant sentences. When analysing the differences between the judges we found three main categories:

Generalisation One judge creates a cluster that from his point of view is homogeneous:

1. Since then, the Rushdie issue has turned into a big controversial problem that hinders the relations between Iran and European countries.
2. The Rushdie affair has been the main hurdle in Iran's efforts to improve ties with the European Union.

3. In a statement issued here, the EU said the Iranian decision opens the way for closer cooperation between Europe and the Tehran government.
4. “These assurances should make possible a much more constructive relationship between the United Kingdom, and I believe the European Union, with Iran, and the opening of a new chapter in our relations,” Cook said after the meeting.

Another judge however puts these sentences into two separate cluster (1,2) and (3,4).The first judge chooses a more general approach and created a cluster about the relationship between Iran and the EU, whereas the other judge distinguishes between the improvement of the relationship and the reason for the problems in the relationship.

Emphasise Two judges can emphasise on different parts of a sentence. For example the sentence ”All 217 people aboard the Boeing 767-300 died when it plunged into the Atlantic off the Massachusetts coast on Oct. 31, about 30 minutes out of New York’s Kennedy Airport on a night flight to Cairo.” was clustered together with other sentence about the number of casualties by one judge. Another judge emphasised on the course of events and put it into a different cluster.

Inference Humans use different level of interference. One judge clustered the sentence ”Schulz, who hated to travel, said he would have been happy living his whole life in Minneapolis.” together with other sentences which said that Schulz is from Minnesota although this sentence does not clearly state this. This judge interfered from ”he would have been happy living his whole life in Minneapolis” that he actually is from Minnesota.

5 Evaluation measures

The evaluation measures will compare a set of clusters to a set of classes. An ideal evaluation measure should reward a set of clusters if the clusters are pure or homogeneous, so that it only contains sentences from one class. On the other hand it should also reward the set if all/most of the sentences of a class are in one cluster (completeness). If sentences that in the gold standard make up one class are grouped into two clusters, the measure should penalise the clustering less than if a lot of irrelevant sentences were in the same cluster. Homogeneity is more important to us.

D is a set of N sentences d_a so that $D = \{d_a|a = 1, ..., N\}$. A set of clusters $L = \{l_j|j = 1, ..., |L|\}$ is a partition of a data set D into disjoint subsets called clusters, so that $l_j \cap l_m = \emptyset$. $|L|$ is the number of clusters in L. A set of clusters that contains only one cluster with all the sentences of D will be called L_{one}. A cluster that contains only one object is called a singleton and a set of clusters that only consists of singletons is called L_{single}.

A set of classes $C = \{c_i|i = 1, ..., |C|\}$ is a partition of a data set D into disjoint subsets called classes, so that $c_i \cap c_m = \emptyset$. $|C|$ is the number of classes in C. C is also called a gold standard of a clustering of data set D because this set contains the ”ideal” solution to a clustering task and other clusterings are compared to it.

5.1 V-measure and V_{beta}

The V-measure (Rosenberg and Hirschberg, 2007) is an external evaluation measure based on conditional entropy:

$$V(L, C) = \frac{(1 + \beta)hc}{\beta h + c} \tag{1}$$

It measures homogeneity (h) and completeness (c) of a clustering solution (see equation 2 where n_j^i is the number of sentences l_j and c_i share, n_i the number of sentences in c_i and n_j the number of sentences in l_j)

$$\begin{gathered} h = 1 - \frac{H(C|L)}{H(C)} \qquad c = 1 - \frac{H(L|C)}{H(L)} \\ H(C|L) = -\sum_{j=1}^{|L|}\sum_{i=1}^{|C|} \frac{n_j^i}{N} log \frac{n_j^i}{n_j} \\ H(C) = -\sum_{i=1}^{|C|} \frac{n^i}{N} log \frac{n^i}{N} \\ H(L) = -\sum_{j=1}^{|L|} \frac{n^j}{N} log \frac{n^j}{N} \\ H(L|C) = -\sum_{i=1}^{|C|}\sum_{j=1}^{|L|} \frac{n_j^i}{N} log \frac{n_j^i}{n_i} \end{gathered} \tag{2}$$

A cluster set is homogeneous if only objects from a single class are assigned to a single cluster. By calculating the conditional entropy of the class distribution given the proposed clustering it can be measured how close the clustering is to complete homogeneity which would result in zero entropy. Because conditional entropy is constrained by the size of the data set and the distribution of the class sizes it is normalized by $H(C)$ (see equation 2). Completeness on the other hand is achieved if all data points from a single class are assigned to a single cluster which results in $H(L|C) = 0$.

The V-measure can be weighted. If $\beta > 1$ the completeness is favoured over homogeneity whereas the weight of homogeneity is increased if $\beta < 1$.

Vlachos et al. (2009) proposes V_{beta} where β is set to $\frac{|L|}{|C|}$. This way the shortcoming of the V-measure to favour cluster sets with many more clusters than classes can be avoided. If $|L| > |C|$ the weight of homogeneity is reduced, since clusterings with large $|L|$ can reach high homogeneity quite easily, whereas $|C| > |L|$ decreases the weight of completeness. V-measure and V_{beta} can range between 0 and 1, they reach 1 if the set of clusters is identical to the set of classes.

5.2 Normalized Mutual Information

Mutual Information (I) measures the information that C and L share and can be expressed by using entropy and conditional entropy:

$$I = H(C) + H(L) - H(C, L) \tag{3}$$

There are different ways to normalise I. Manning et al. (2008) uses

$$NMI = \frac{I(L,C)}{\frac{H(L)+H(C)}{2}} = 2\frac{I(L,C)}{H(L) + H(C)} \tag{4}$$

which represents the average of the two uncertainty coefficients as described in Press et al. (1988).

Generalise NMI to $NMI_\beta = \frac{(1+\beta)I}{\beta H(L)+H(C)}$. Then NMI_β is actually the same as V_β:

$$\begin{aligned} h &= 1 - \frac{H(C|L)}{H(C)} \\ &\Rightarrow H(C)h = H(C) - H(C|L) \\ &= H(C) - H(C,L) + H(L) = I \end{aligned}$$

$$\begin{aligned} c &= 1 - \frac{H(L|C)}{H(L)} \\ &\Rightarrow H(L)c = H(L) - H(L|C) \\ &= H(L) - H(L,C) + H(C) = I \\ V &= \frac{(1+\beta)hc}{\beta h + c} \\ &= \frac{(1+\beta)H(L)H(C)hc}{\beta H(L)H(C)h + H(L)H(C)c} \end{aligned} \tag{5}$$

$H(C)h$ and $H(L)c$ are substituted by I:

$$\begin{aligned} &\frac{(1+\beta)I^2}{\beta H(L)I + H(C)I} \\ &= \frac{(1+\beta)I}{\beta H(L) + H(C)} = NMI_\beta \\ V_1 &= 2\frac{I}{H(L) + H(C)} = NMI \end{aligned} \tag{6}$$

5.3 Variation of Information (VI) and Normalized VI

The VI-measure (Meila, 2007) also measures completeness and homogeneity using conditional entropy. It measure the distance between two clusterings and thereby the amount of information gained in changing from C to L. For this measure the conditional entropies are added up:

$$VI(L, C) = H(C|L) + H(L|C) \tag{7}$$

Remember small conditional entropies mean that the clustering is near to complete homogeneity/ completeness, so the smaller VI the better ($VI = 0$ if $L = C$). The maximum of VI is $log\ N$ e.g. for $VI(L_{single}, C_{one})$. VI can be normalized, then it can range from 0 (identical clusters) to 1.

$$NVI(L, C) = \frac{1}{log\,\mathrm{N}} VI(L, C) \tag{8}$$

V-measure, V_{beta} and VI measure both completeness and homogeneity, no mapping between classes and clusters is needed (Rosenberg and Hirschberg, 2007) and they are only dependent on the relative size of the clusters (Vlachos et al., 2009).

5.4 Rand Index (RI)

The Rand Index (Rand, 1971) compares two clusterings with a combinatorial approach. Each pair of objects can fall into one of four categories:

- TP (true positives) = objects belong to one class and one cluster
- FP (false positives) = objects belong to different classes but to the same cluster
- FN (false negatives) = objects belong to the same class but to different clusters
- TN (true negatives) = objects belong to different classes and to different cluster

By dividing the total number of correctly clustered pairs by the number of all pairs, RI gives the percentage of correct decisions.

$$RI = \frac{TP + TN}{TP + FP + TN + FN} \tag{9}$$

RI can range between 0 and 1 where 1 corresponds to identical clusterings. Meila (2007) mentions that in practise RI concentrates in a small interval near 1 (for more detail see section 5.7). Another shortcoming is that RI gives equal weight to FPs and FNs.

5.5 Entropy and Purity

Entropy and Purity are widely used evaluation measures (Zhao and Karypis, 2001). They both can be used to measure homogeneity of a cluster. Both measures give better values when the number of clusters increase, with the best result for L_{single}. Entropy ranges from 0 for identical clusterings or L_{single} to $log\ N$ e.g. for C_{single} and L_{one}. The values of P can range between 0 and 1, where a value close to 0 represents a bad clustering solution and a perfect clustering solution gets a value of 1.

$$\begin{aligned} Entropy &= \sum_{j=1}^{|L|} \frac{n_j}{N} \left(-\frac{1}{\log |C|} \sum_{i=1}^{|C|} \frac{n_j^i}{n_j} \log \frac{n_j^i}{n_j} \right) \\ Purity &= \frac{1}{N} \sum_{j=1}^{|L|} \max_i \left(n_j^i \right) \end{aligned} \tag{10}$$

5.6 F-measure

The F-measure is a well known metric from IR, which is based on Recall and Precision. The version of the F-score (Hess and Kushmerick, 2003) described here measures the overall Precision and Recall. This way a mapping between a cluster and a class is omitted which may cause problems if $|L|$ is considerably different to $|C|$ or if a cluster could be mapped to more than one class. Precision and Recall here are based on pairs of objects and not on individual objects.

$$\begin{gathered} P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \\ F(L,C) = \frac{2PR}{P + R} \end{gathered} \tag{11}$$

5.7 Discussion of the Evaluation measures

We used one cluster set to analyse the behaviour and quality of the evaluation measures. Variations of that cluster set were created by randomly splitting and merging the clusters. These modified sets were then compared to the original set. This experiment will help to identify the advantages and disadvantages of the measures, what the values reveal about the quality of a set of clusters and how the measures react to changes in the cluster set.

We used the set of clusters created by Judge_A for the Rushdie sentence set. It contains 70 sentences in 15 clusters. This cluster set was modified by splitting and merging the clusters randomly until we got L_{single} with 70 clusters and L_{one} with one cluster. The original set of clusters (C_A) was compared to the modified versions of the set (see figure 1). The evaluation measures reach their best values if $C_A = 15$ clusters is compared to itself.

The F-measure is very sensitive to changes. It is the only measure which uses its full measurement range. $F = 0$ if C_A is compared to $L_{A-single}$, which means that the F-measure considers $L_{A-single}$ to be the opposite of C_A. Usually L_{one} and $L_{A-single}$ are considered to be observe and a measure should only reach its worst possible value if these sets are compared. In other words the F-measure might be too sensitive for our task. The RI stays most of the time in an interval between 0.84 and 1. Even for the comparison between C_A and $L_{A-single}$ the RI is 0.91. This behaviour was also described in Meila (2007) who observed that the RI concentrates in a small interval near 1.

As described in section 5.5 Purity and Entropy both measure homogeneity. They both react to changes slowly. Splitting and merging have almost the same effect on Purity. It reaches ≈ 0.6 when the clusters of the set were randomly split or merged four times. As explained above our ideal evaluation measure should punish a set of clusters which puts sentences of the same class into two clusters less than if sentences are merged with irrelevant ones. Homogeneity decreases if unrelated clusters are merged whereas a decline in completeness follows from splitting clusters. In other words for our task a measure should decrease more if two clusters are merged than if a cluster is split.

Entropy for example is more sensitive to merging than splitting. But Entropy only measures homogeneity and an ideal evaluation measure should also consider completeness.

The remaining measures V_{beta}, $V_{0.5}$ and NVI/VI all fulfil our criteria of a good evaluation measure. All of them are more affected by merging than by splitting and use their measuring range appropriately. $V_{0.5}$ favours homogeneity over completeness, but it reacts to changes less than V_{beta}. The V-measure can also be inaccurate if the $|L|$ is considerably different to $|C|$. V_{beta} (Vlachos et al., 2009) tries to overcome this problem and the tendency of the V-measure to favour clusterings with a large number of clusters.

Since VI is measured in bits with an upper bound of $log\ N$, values for different sets are difficult to compare. NVI tries to overcome this problem by

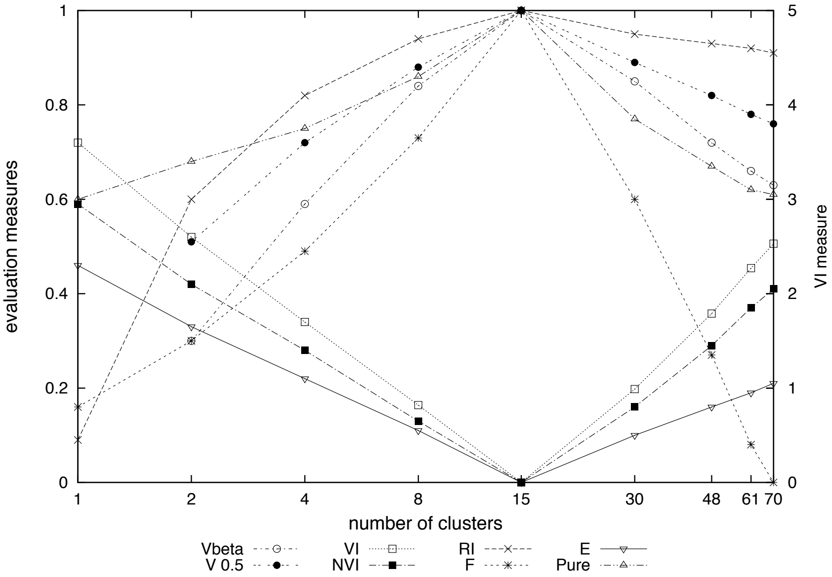


Figure 1: Behaviour of evaluation measure when randomly changed sets of clusters are compared to the original set.

normalising VI by dividing it by $log\ N$. As Meila (2007) pointed out, this is only convenient if the comparison is limited to one data set.

In this paper V_{beta}, $V_{0.5}$ and NVI will be used for evaluation purposes.

6 Comparability of Clusterings

Following our procedure and guidelines the judges have to filter out all irrelevant sentences that are not related to another sentence from a different document. The number of these irrelevant sentences are different for every sentence set and every judge (see table 2). The evaluation measures require the same number of sentences in each set of clusters to compare them. The easiest way to ensure that each cluster set for a sentence set has the same number of sentences is to add the sentences that were filtered out by the judges to the corresponding set of clusters. There are different ways to add these sentences:

1. singletons: Each irrelevant sentence is added to set of clusters as a cluster of its own
2. bucket cluster: All irrelevant sentences are put into one cluster which is added to the set of clusters.

Adding each irrelevant sentence as a singleton seems to be the most intuitive way to handle the problem with the sentences that were filtered out. However this approach has some disadvantages. The judges will be rewarded disproportionately high for any singleton they agreement on. Thereby the disagreement on the more important clustering will be less punished. With every singleton the judges agree on the completeness and homogeneity of the whole set of clusters increases.

On the other hand the sentences in a bucket cluster are not all semantically related to each other and the cluster is not homogeneous which is contradictory to our definition of a cluster. Since the irrelevant sentences are combined to only one cluster, the judges will not be rewarded disproportionately high for their agreement. However two bucket clusters from two different sets of clusters will never be exactly the same and therefore the judges will be punished more for the disagreement on the irrelevant sentences

We have to considers these factors when we interpret the results of the inter-judge agreement.

7 Inter-Judge Agreement

We added the irrelevant sentences to each set of clusters created by the judges as described in section 6. These modified sets were then compared to each other in order to evaluate the agreement between the judges. The results are shown in table 3. For each sentence set 100 random sets of clusters were created and compared to the modified sets (in total 1300 comparisons for each method of adding irrelevant sentences). The average values of these

set	judges	singleton clusters			bucket cluster		
		V$_{beta}$	$V_{0.5}$	NVI	V_{beta}	$V_{0.5}$	NVI
Volcano	A-B	0.92	0.93	0.13	0.52	0.54	0.39
	A-D	0.92	0.93	0.13	0.44	0.49	0.4
	B-D	0.95	0.95	0.08	0.48	0.48	0.31
Rushdie	A-B	0.87	0.88	0.19	0.3	0.31	0.59
	A-H	0.86	*0.86*	*0.2*	**0.69**	**0.69**	0.32
	B-H	*0.85*	0.87	*0.2*	*0.25*	*0.27*	*0.64*
EgyptAir	A-B	0.94	0.95	0.1	0.41	0.45	0.34
	A-H	0.93	0.93	0.12	0.57	0.58	0.31
	A-O	0.94	0.94	0.11	0.44	0.46	0.36
	B-H	0.93	0.94	0.11	0.44	0.46	0.3
	B-O	0.96	0.96	0.08	0.42	0.43	0.28
	H-O	0.93	0.94	0.12	0.44	0.44	0.34
Schulz	A-B	**0.98**	**0.98**	**0.04**	0.54	0.56	**0.15**
	A-J	0.89	0.9	0.17	0.39	0.4	0.34
	B-J	0.89	0.9	0.18	0.28	0.31	0.35
base		0.66	0.75	0.44	0.29	0.28	0.68

Table 3: Inter-judge agreement for the four sentence set.

comparisons are used as a baseline.

The inter-judge agreement is most of the time higher than the baseline. Only for the Rushdie sentence set the agreement between Judge_B and Judge_H is lower for V_{beta} and $V_{0.5}$ if the bucket cluster method is used.

As explained in section 6 the two methods for adding sentences that were filtered out by the judges have a notable influence on the values of the evaluation measures. When adding singletons to the set of clusters the inter-judge agreement is considerably higher than with the bucket cluster method. For example the agreement between Judge_A and Judge_B is 0.98 for V_{beta} and $V_{0.5}$ and 0.04 for NVI when singletons are added. Here the judges filter out the same 185 sentences which is equivalent to 74.6% of all sentences in the set. In other words 185 clusters are already considered to be homogen and complete, which gives the comparison a high score. Five of the 15 clusters Judge_A created contain only sentences there were marked as irrelevant by Judge_B. In total 25 sentences are used in clusters by Judge_A which are singletons in Judge_B's set. Judge_B included nine other sentences that are singletons in the set of Judge_A. Four of the clusters are exactly the same in both sets, they contain 16 sentences. To get from Judge_A's set to the set of Judge_B 37 sentences would have to be deleted, added or moved.

With the bucket cluster method Judge_A and Judge_H for the Rushdie sentence set have the best inter-judge agreement. At the same time this combination receives the worst $V_{0.5}$ and NVI values with the singleton method. The two judges agree on 22 irrelevant sentences, which account for 21.35% of all sentences. Here the singletons have far less influence on the evaluation measures then the first example. Judge_A includes 7 sentences that are filtered out by Judge_H who uses another 11 sentences. Only one cluster is exactly the same in both sets. To get from Judge_A's set to Judge_H's cluster 11 sentences have to be deleted, 7 to be added, one cluster has to be split in two and 11 sentences have to be moved from one cluster to another.

Although the two methods of adding irrelevant sentences to the sets of cluster result in different values for the inter-judge agreement, we can conclude that the agreement between the judges is good and (almost) always exceed the baseline. Overall Judge_B seems to have the highest agreement throughout all sentence sets with all other judges.

8 Conclusion and Future Work

In this paper we presented a gold standard for sentence clustering for Multi-Document Summarization. The data set used, the guidelines and procedure given to the judges were discussed. We showed that the agreement between the judges in sentence clustering is good and exceeds the baseline. This gold standard will be used for further experiments on clustering for Multi-Document Summarization. The next step will be to compared the output of a standard clustering algorithm to the gold standard.

References

Ramiz M. Aliguliyev. 2006. A novel partitioning-based clustering method and generic document summarization. In *WI-IATW '06: Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology*, Washington, DC, USA.

Regina Barzilay and Kathleen R. McKeown. 2005. Sentence Fusion for Multidocument News Summariation. *Computational Linguistics*, 31(3):297–327.

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The Second Release of the RASP System. In *COLING/ACL 2006 Interactive Presentation Sessions*, Sydney, Australien. The Association for Computer Linguistics.

Vasileios Hatzivassiloglou, Judith L. Klavans, Melissa L. Holcombe, Regina Barzilay, Min-Yen Kan, and Kathleen R. McKeown. 2001. SIMFINDER: A Flexible Clustering Tool for Summarization. In *NAACL Workshop on Automatic Summarization*, pages 41–49. Association for Computational Linguistics.

Andreas Hess and Nicholas Kushmerick. 2003. Automatically attaching semantic metadata to web services. In *Proceedings of the 2nd International Semantic Web Conference (ISWC 2003)*, Florida, USA.

Christopher D. Manning, Prabhakar Raghavan, and Heinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Daniel Marcu and Laurie Gerber. 2001. An inquiry into the nature of multidocument abstracts, extracts, and their evaluation. In *Proceedings of the NAACL-2001 Workshop on Automatic Summarization*, Pittsburgh, PA.

Marina Meila. 2007. Comparing clusterings–an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895.

Martina Naughton. 2007. Exploiting structure for event discovery using the mdi algorithm. In *Proceedings of the ACL 2007 Student Research Workshop*, pages 31–36, Prague, Czech Republic, June. Association for Computational Linguistics.

William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. 1988. *Numerical Recipies in C: The art of Scientific Programming*. Cambridge University Press, Cambridge, England.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *In ANLP/NAACL Workshop on Summarization*, pages 21–29, Morristown, NJ, USA. Association for Computational Linguistics.

William M. Rand. 1971. Objective criteria for the evaluation of clustering methods. *American Statistical Association Journal*, 66(336):846–850.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420.

Andreas Vlachos, Anna Korhonen, and Zoubin Ghahramani. 2009. Unsupervised and Constrained Dirichlet Process Mixture Models for Verb Clustering. In *Proceedings of the EACL workshop on GEometrical Models of Natural Language Semantics*.

Dingding Wang, Tao Li, Shenghuo Zhu, and Chris Ding. 2008. Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314, New York, NY, USA. ACM.

Hongyuan Zha. 2002. Generic Summarization and Keyphrase Extraction using Mutual Reinforcement Principle and Sentence Clustering. In *Proceedings of the 25th Annual ACM SIGIR Conference*, pages 113–120, Tampere, Finland.

Ying Zhao and George Karypis. 2001. Criterion functions for document clustering: Experiments and analysis. Technical report, Department of Computer Science, University of Minnesota. (Technical Report #01-40).

Author Index