ACL-05

Proceedings of the Student Research Workshop

June 27, 2005
University of Michigan
Ann Arbor, Michigan, USA

Production and Manufacturing by
Omnipress Inc.
Post Office Box 7214
Madison, WI 53707-7214

Sponsored by
The National Science Foundation

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
75 Paterson Street, Suite 9
New Brunswick, NJ 08901
USA
Tel: +1-732-342-9100
Fax: +1-732-342-9339
`acl@aclweb.org`

Introduction

Welcome to the ACL Student Research Workshop! We had an amazing amount of student participation this year. We received a record number of submissions – over 70 papers from students in 19 countries. In order to accommodate the overwhelming response, weve expanded the Student Research Workshop. In addition to the regular paper presentations it now includes a poster session. The acceptance rate for the regular paper presentations was 11% (more competitive than the main conference!), and the acceptance rate for the poster presentations from the remaining submissions was 28%. These proceedings contain papers for both the regular and the poster presentations.

We are grateful to our faculty advisor, Regina Barzilay, and to Mary Harper of the National Science Foundation for organizing sponsorship from the National Science Foundation for the Student Research Workshop. The NSFs generous grant has paid for the conference registration fees for all of the student workshop participants and a good portion of their travel expenses. We are also grateful for the Don and Betty Walker International Student Fund which has provided assistance for two students who had their work accepted in the main conference.

We would like to extend our thanks to the students and faculty on the program committee who dutifully reviewed the papers and gave useful feedback to everyone who submitted papers, and to all the students who submitted such excellent work.

Enjoy the workshop!

Chris Callison-Burch and Stephen Wan
ACL Student Research Workshop Co-Chairs

Co-Chairs:

Chris Callison-Burch (Univeristy of Edinburgh)
Stephen Wan (Macquarie University)

Faculty Advisor:

Regina Barzilay (MIT)

Program Committee:

Laura Alonso (Barcelona)
Timothy Baldwin (Melbourne)
Colin Bannard (Edinburgh)
Phil Blunsom (Melbourne)
Bernd Bohnet (Stuttgart)
Cem Bozsahin (Middle East Technical University)
Chris Brew (Ohio State)
Marine Carpuat (HKUST)
Stephen Clark (Oxford)
Trevor Cohn (Melbourne)
James Curran (Sydney)
Tiphaine Dalmas (Edinburgh)
Hal Daume III (ISI)
Mona Diab (Stanford)
Mark Dras (Macquarie)
Pablo Duboue (IBM)
Elena Filatova (Columbia)
Erin Fitzgerald (JHU)
Dan Flickinger (Stanford)
Pablo Gamallo (Universidade Nova de Lisboa)
Claire Grover (Edinburgh)
Graeme Hirst (Toronto)
Julia Hockenmaier (U Penn)
Ben Hutchinson (Edinburgh)
Frank Keller (Edinburgh)
Simon King (Edinburgh)
Alistair Knott (Otago)
Philipp Koehn (Edinburgh)
Emiel Krahmer (Tilburg)
Corrin Lakeland (Otago)
Mirella Lapata (Edinburgh)
Alon Lavie (CMU)
Maria Liakata (Oxford)

Daniel Marcu (ISI)
Daniel Midgley (University of Western Australia)
Diego Molla (Macquarie)
Ani Nenkova (Columbia)
Leif Nielsen (King’s College)
Bo Pang (Cornell)
Cecile Paris (CSIRO)
Jean-Philippe Prost (Macquarie)
Steve Renals (Edinburgh)
Philip Resnik (Maryland)
Graeme Ritchie (Aberdeen)
Charles Schafer (JHU)
Advaith Siddharthan (Columbia)
Michel Simard (Xerox)
Matthew Stone (Rutgers)
Mark Swerts (Tilburg)
David Talbot (Edinburgh)
Leonoor van der Beek (Groningen)
Florian Wolf (Cambridge)

Table of Contents

Student Research Workshop Program

Monday June 27, 2005: Presentations

Student Presentations: Session 1

9:00–9:30 *Hybrid methods for POS guessing of Chinese unknown words*
Xiaofei Lu

9:30–10:00 *Understanding the thematic structure of the Qur'an: an exploratory multivariate approach*
Naglaa Thabet

10:00–10:30 *An Extensive Empirical Study of Collocation Extraction Methods*
Pavel Pecina

Student Presentations: Session 2

2:30–3:00 *Jointly Labeling Multiple Sequences: A Factorial HMM Approach*
Kevin Duh

3:00–3:30 *Exploiting Named Entity Taggers in a Second Language*
Thamar Solorio

Student Presentations: Session 3

4:00–4:30 *Automatic Discovery of Intentions in Text and its Application to Question Answering*
Marta Tatu

4:30–5:00 *American Sign Language Generation: Multimodal NLG with Multiple Linguistic Channels*
Matt Huenerfauth

5:00–5:30 *Using Emoticons to reduce Dependency in Machine Learning Techniques for Sentiment Classification*
Jonathon Read

Monday June 27, 2005: Posters

All posters will be on display during the Student Lunch from 12:00–1:30

Learning Meronyms from Biomedical Text
Angus Roberts

Using Readers to Identify Lexical Cohesive Structures in Texts
Beata Beigman Klebanov

Towards an Optimal Lexicalization in a Natural-Sounding Portable Natural Language Generator for Dialog Systems
Inge M. R. De Bleecker

Phrase Linguistic Classification and Generalization for Improving Statistical Machine Translation
Adrià de Gispert

Automatic Induction of a CCG Grammar for Turkish
Ruken Cakici

Dialogue Act Tagging for Instant Messaging Chat Sessions
Edward Ivanovic

Learning Strategies for Open-Domain Natural Language Question Answering
Eugene Grois

Dependency-Based Statistical Machine Translation
Heidi Fox

Minimalist Parsing of Subjects Displaced from Embedded Clauses in Free Word Order Languages
Asad B. Sayeed

Centrality Measures in Text Mining: Prediction of Noun Phrases that Appear in Abstracts
Zhuli Xie

A corpus-based approach to topic in Danish dialog
Philip Diderichsen and Jakob Elming

(posters continued)

Hybrid Methods for POS Guessing of Chinese Unknown Words

Xiaofei Lu
Department of Linguistics
The Ohio State University
Columbus, OH 43210, USA
xflu@ling.osu.edu

Abstract

This paper describes a hybrid model that combines a rule-based model with two statistical models for the task of POS guessing of Chinese unknown words. The rule-based model is sensitive to the type, length, and internal structure of unknown words, and the two statistical models utilize contextual information and the likelihood for a character to appear in a particular position of words of a particular length and POS category. By combining models that use different sources of information, the hybrid model achieves a precision of 89%, a significant improvement over the best result reported in previous studies, which was 69%.

1 Introduction

Unknown words constitute a major source of difficulty for Chinese part-of-speech (POS) tagging, yet relatively little work has been done on POS guessing of Chinese unknown words. The few existing studies all attempted to develop a unified statistical model to compute the probability of a word having a particular POS category for all Chinese unknown words (Chen et al., 1997; Wu and Jiang, 2000; Goh, 2003). This approach tends to miss one or more pieces of information contributed by the type, length, internal structure, or context of individual unknown words, and fails to combine the strengths of different models. The rule-based approach was rejected with the claim that rules are bound to overgenerate (Wu and Jiang, 2000).

In this paper, we present a hybrid model that combines the strengths of a rule-based model with those of two statistical models for this task. The three models make use of different sources of information. The rule-based model is sensitive to the type, length, and internal structure of unknown words, with overgeneration controlled by additional constraints. The two statistical models make use of contextual information and the likelihood for a character to appear in a particular position of words of a particular length and POS category respectively. The hybrid model achieves a precision of 89%, a significant improvement over the best result reported in previous studies, which was 69%.

2 Chinese Unknown Words

The definition of what constitutes a word is problematic for Chinese, as Chinese does not have word delimiters and the boundary between compounds and phrases or collocations is fuzzy. Consequently, different NLP tasks adopt different segmentation schemes (Sproat, 2002). With respect to any Chinese corpus or NLP system, therefore, unknown words can be defined as character strings that are not in the lexicon but should be identified as segmentation units based on the segmentation scheme. Chen and Bai (1998) categorized Chinese unknown words into the following five types: 1) acronyms, i.e., shortened forms of long names, e.g., *bĕi-dà* for *bĕijīng-dàxué* ‘Beijing University’; 2) proper names, including person, place, and organization names, e.g., *Máo-Zédōng*; 3) derived words, which are created through affixation, e.g., *xiàndài-huà* ‘modernize’; 4) compounds, which are created through compounding, e.g., *zhĭ-lăohŭ* ‘paper tiger’; and 5) nu-

Proceedings of the ACL Student Research Workshop, pages 1–6,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

meric type compounds, including numbers, dates, time, etc., e.g., *liǎng-diǎn* 'two o'clock'. Other types of unknown words exist, such as loan words and reduplicated words. A monosyllabic or disyllabic Chinese word can reduplicate in various patterns, e.g., *zǒu-zǒu* 'take a walk' and *piào-piào-liàng-liàng* 'very pretty' are formed by reduplicating *zǒu* 'walk' and *piào-liàng* 'pretty' respectively.

The identification of acronyms, proper names, and numeric type compounds is a separate task that has received substantial attention. Once a character string is identified as one of these, its POS category also becomes known. We will therefore focus on reduplicated and derived words and compounds only. We will consider unknown words of the categories of noun, verb, and adjective, as most unknown words fall under these categories (Chen and Bai, 1998). Finally, monosyllabic words will not be considered as they are well covered by the lexicon.

3 Previous Approaches

Previous studies all attempted to develop a unified statistical model for this task. Chen et al. (1997) examined all unknown nouns[1], verbs, and adjectives and reported a 69.13% precision using Dice metrics to measure the affix-category association strength and an affix-dependent entropy weighting scheme for determining the weightings between prefix-category and suffix-category associations. This approach is blind to the type, length, and context of unknown words. Wu and Jiang (2000) calculated *P(Cat,Pos,Len)* for each character, where *Cat* is the POS of a word containing the character, *Pos* is the position of the character in that word, and *Len* is the length of that word. They then calculated the POS probabilities for each unknown word as the joint probabilities of the *P(Cat,Pos,Len)* of its component characters. This approach was applied to unknown nouns, verbs, and adjectives that are two to four characters long[2]. They did not report results on unknown word tagging, but reported that the new word identification and tagging mechanism increased parser coverage. We will show that this approach suffers reduced recall for multisyllabic words if the training corpus is small. Goh (2003) reported a precision of 59.58% on all unknown words using Support Vector Machines.

Several reasons were suggested for rejecting the rule-based approach. First, Chen et al. (1997) claimed that it does not work because the syntactic and semantic information for each character or morpheme is unavailable. This claim does not fully hold, as the POS information about the component words or morphemes of many unknown words is available in the training lexicon. Second, Wu and Jiang (2000) argued that assigning POS to Chinese unknown words on the basis of the internal structure of those words will "result in massive overgeneration" (p. 48). We will show that overgeneration can be controlled by additional constraints.

4 Proposed Approach

We propose a hybrid model that combines the strengths of different models to arrive at better results for this task. The models we will consider are a rule-based model, the trigram model, and the statistical model developed by Wu and Jiang (2000). Combination of the three models will be based on the evaluation of their individual performances on the training data.

4.1 The Rule-Based Model

The motivations for developing a set of rules for this task are twofold. First, the rule-based approach was dismissed without testing in previous studies. However, hybrid models that combine rule-based and statistical models outperform purely statistical models in many NLP tasks. Second, the rule-based model can incorporate information about the length, type, and internal structure of unknown words at the same time.

Rule development involves knowledge of Chinese morphology and generalizations of the training data. Disyllabic words are harder to generalize than longer words, probably because their monosyllabic component morphemes are more fluid than the longer component morphemes of longer words. It is interesting to see if reduction in the degree of fluidity of its components makes a word more predictable. We therefore develop a separate set of rules for words that are two, three, four, and five

[1] Including proper names and time nouns, which we excluded for the reason discussed in section 2.

[2] Excluding derived words and proper names.

Chars	T1	T2	T3	T4	Total
2	1	2	1	2	6
3	2	6	2	5	15
4	2	2	0	8	12
5+	0	1	0	1	2
Total	5	11	3	16	35

Table 1: Rule distribution

or more characters long. The rules developed fall into the following four types: 1) reduplication rules (T1), which tag reduplicated unknown words based on knowledge about the reduplication process; 2) derivation rules (T2), which tag derived unknown words based on knowledge about the affixation process; 3) compounding rules (T3), which tag unknown compounds based on the POS information of their component words; and 4) rules based on generalizations about the training data (T4). Rules may come with additional constraints to avoid overgeneration. The number of rules in each set is listed in Table 1. The complete set of rules are developed over a period of two weeks.

As will be shown below, the order in which the rules in each set are applied is crucial for dealing with ambiguous cases. To illustrate how rules work, we discuss the complete set of rules for disyllabic words here[3]. These are given in Figure 1, where A and B refer to the component morpheme of an unknown AB. As rules for disyllabic words tend to overgenerate and as we prefer precision over recall for the rule-based model, most rules in this set are accompanied with additional constraints.

```
if A equals B
    if A is a verb morpheme, AB is a verb
    else if A is a noun morpheme, AB is a noun
    else if A is an adjective morpheme, AB is a stative
        adjective/adverb
else if B equals er, AB is a noun
else if B is a categorizing suffix AND A is not a verb
    morpheme, AB is a noun
else if A and B are both noun morphemes but not verb
    morphemes, AB is a noun
else if A occurs verb-initially only AND B is not a noun
    morpheme AND B does not occur noun-finally only,
    AB is a verb
else if B occurs noun-finally only AND A is not a verb
    morpheme AND A does not occur verb-initially only,
    AB is a noun
```

Figure 1: Rules for disyllabic words

In the first reduplication rule, the order of the three cases is crucial in that if A can be both a verb and a noun, AA is almost always a verb. The second rule tags a disyllabic unknown word formed by attaching the diminutive suffix *er* to a monosyllabic root as a noun. This may appear a hasty generalization, but examination of the data shows that *er* rarely attaches to monosyllabic verbs except for the few well-known cases. In the third rule, a categorizing suffix is one that attaches to other words to form a noun that refers to a category of people or objects, e.g., *jiā* '-ist'. The constraint "A is not a verb morpheme" excludes cases where B is polysemous and does not function as a categorizing suffix but a noun morpheme. Thus, this rule tags *bèng-yè* 'water-pump industry' as a noun, but not *lí-yè* leave-job 'resign'. The fourth rule tags words such as *shā-xiāng* 'sand-box' as nouns, but the constraints prevent verbs such as *sōng-kòu* 'loosen-button' from being tagged as nouns. *Sōng* can be both a noun and a verb, but it is used as a verb in this word. The last two rules make use of two lists of characters extracted from the list of disyllabic words in the training data, i.e., those that have only appeared in the verb-initial and noun-final positions respectively. This is done because in Chinese, disyllabic compound verbs tend to be head-initial, whereas disyllabic compound nouns tend to be head-final. The fifth rule tags words such as *dīng-yǎo* 'sting-bite' as verbs, and the additional constraints prevent nouns such as *fú-xiàng* 'lying-elephant' from being tagged as verbs. The last rule tags words such as *xuě-bèi* 'snow-quilt' as nouns, but not *zhāi-shāo* pick-tip 'pick the tips'.

One derivation rule for trisyllabic words has a special status. Following the tagging guidelines of our training corpus, it tags a word ABC as verb/deverbal noun (v/vn) if C is the suffix *huà* '-ize'. Disambiguation is left to the statistical models.

4.2 The Trigram Model

The trigram model is used because it captures the information about the POS context of unknown words and returns a tag for each unknown word. We assume that the unknown POS depends on the previous two POS tags, and calculate the trigram probability $P(t_3|t_1, t_2)$, where t_3 stands for the unknown

[3] Multisyllabic words can have various internal structures, e.g., a disyllabic noun can have a N-N, Adj-N, or V-N structure.

POS, and t_1 and t_2 stand for the two previous POS tags. The POS tags for known words are taken from the tagged training corpus. Following Brants (2000), we first calculate the maximum likelihood probabilities $\hat{P}$ for unigrams, bigrams, and trigrams as in (1-3). To handle the sparse-data problem, we use the smoothing paradigm that Brants reported as delivering the best result for the TnT tagger, i.e., the context-independent variant of linear interpolation of unigrams, bigrams, and trigrams. A trigram probability is then calculated as in (4).

$$\hat{P}(t_3) = f(t_3)/N \tag{1}$$

$$\hat{P}(t_3|t_2) = f(t_2, t_3)/f(t_2) \tag{2}$$

$$\hat{P}(t_3|t_1, t_2) = f(t_1, t_2, t_3)/f(t_1, t_2) \tag{3}$$

$$P(t_3|t_1, t_2) = \lambda_1 \hat{P}(t_3) + \lambda_2 \hat{P}(t_3|t_2) + \lambda_3 \hat{P}(t_3|t_1, t_2) \tag{4}$$

As in Brants (2000), $\lambda_1 + \lambda_2 + \lambda_3 = 1$, and the values of λ_1, λ_2, and λ_3 are estimated by deleted interpolation, following Brants' algorithm for calculating the weights for context-independent linear interpolation when the n-gram frequencies are known.

4.3 Wu and Jiang's (2000) Statistical Model

There are several reasons for integrating another statistical model in the model. The rule-based model is expected to yield high precision, as over-generation is minimized, but it is bound to suffer low recall for disyllabic words. The trigram model covers all unknown words, but its precision needs to be boosted. Wu and Jiang's (2000) model provides a good complement for the two, because it achieves a higher recall than the rule-based model and a higher precision than the trigram model for disyllabic words. As our training corpus is relatively small, this model will suffer a low recall for longer words, but those are handled effectively by the rule-based model. In principle, other statistical models can also be used, but Wu and Jiang's model appears more appealing because of its relative simplicity and higher or comparable precision. It is used to handle disyllabic and trisyllabic unknown words only, as recall drops significantly for longer words.

4.4 Combining Models

To determine the best way to combine the three models, their individual performances are evaluated in the training data first to identify their strengths. Based on that evaluation, we come up with the algorithm in Figure 2. For each unknown word, if the trigram model returns exactly one POS tag, that tag is prioritized, because in the training data, such tags turn out to be always correct. Otherwise, the guess returned by the rule-based model is prioritized, followed by Wu and Jiang's model. If neither of them returns a guess, the guess returned by the trigram model is accepted. This order of priority is based on the precision of the individual models in the training data. If the rule-based model returns the "v/vn" guess, we first check which of the two tags ranks higher in the list of guesses returned by Wu and Jiang's model. If that list is empty, we then check which of them ranks higher in the list of guesses returned by the trigram model.

```
for each unknown word
    if the trigram model returns one single guess, take it
    else if the rule-based model returns a non-v/vn tag, take it
    else if the rule-based model returns a v/vn tag
        if W&J's model returns a list of guesses
            eliminate non-v/vn tags on that list and return the
            rest of it
        else eliminate non-v/vn tags on the list returned by the
            trigram model and return the rest of it
    else if W&J's model returns a list of guesses, take it
    else return the list of guesses returned by the trigram
        model
```

Figure 2: Algorithm for combining models

5 Results

5.1 Experiment Setup

The different models are trained and tested on a portion of the Contemporary Chinese Corpus of Peking University (Yu et al., 2002), which is segmented and POS tagged. This corpus uses a tagset consisting of 40 tags. We consider unknown words that are 1) two or more characters long, 2) formed through reduplication, derivation, or compounding, and 3) in one of the eight categories listed in Table 2. The corpus consists of all the news articles from *People's Daily* in January, 1998. It has a total of 1,121,016 tokens, including 947,959 word tokens and 173,057 punctuation marks. 90% of the data are used for training, and the other 10% are reserved for testing. We downloaded a reference lexicon[4] containing 119,791

[4] From http://www.mandarintools.com/segmenter.html.

entries. A word is considered unknown if it is in the wordlist extracted from the training or test data but is not in the reference lexicon. Given this definition, we first train and evaluate the individual models on the training data and then evaluate the final combined model on the test data. The distribution of unknown words is summarized in Table 3.

Tag	Description
a	Adjective
ad	Deadjectval adverb
an	Deadjectival noun
n	Noun
v	Verb
vn	Deverbal noun
vd	Deverbal adjective
z	Stative adjective and adverb

Table 2: Categories of considered unknown words

Chars	Training Data		Test Data	
	Types	Tokens	Types	Tokens
2	2611	4789	387	464
3	3818	7378	520	764
4	490	1229	74	125
5+	188	698	20	56
Total	7107	14094	1001	1509

Table 3: Unknown word distribution in the data

5.2 Results for the Individual Models

The results for the rule-based model are listed in Table 4. Recall (R) is defined as the number of correctly tagged unknown words divided by the total number of unknown words. Precision (P) is defined as the number of correctly tagged unknown words divided by the number of tagged unknown words. The small number of words tagged "v/vn" are excluded in the count of tagged unknown words for calculating precision, as this tag is not a final guess but is returned to reduce the search space for the statistical models. F-measure (F) is computed as $2 * RP/(R + P)$. The rule-based model achieves very high precision, but recall for disyllabic words is low.

The results for the trigram model are listed in Table 5. Candidates are restricted to the eight POS categories listed in Table 2 for this model. Precision for the best guess in both datasets is about 62%.

The results for Wu and Jiang's model are listed in Table 6. Recall for disyllabic words is much higher than that of the rule-based model. Precision for disyllabic words reaches mid 70%, higher than that of the trigram model. Precision for trisyllabic words is very high, but recall is low.

Chars	Data	R	P	F
2	Training	24.05	96.94	38.54
	Test	27.66	96.89	43.03
3	Training	93.50	99.83	96.56
	Test	93.72	99.86	96.69
4	Training	98.70	99.02	98.86
	Test	99.20	99.20	99.20
5+	Training	99.86	100	99.93
	Test	100	100	100
Total	Training	70.60	99.40	82.56
	Test	69.72	99.34	81.94

Table 4: Results for the rule-based model

Guesses	1-Best	2-Best	3-Best
Training	62.01	93.63	96.21
Test	62.96	92.64	94.30

Table 5: Results for the trigram model

Chars	Data	R	P	F
2	Training	65.19	75.57	67.00
	Test	63.82	77.92	70.17
3	Training	59.50	98.41	74.16
	Test	55.63	99.07	71.25

Table 6: Results for Wu and Jiang's (2000) model

5.3 Results for the Combined Model

To evaluate the combined model, we first define the upper bound of the precision for the model as the number of unknown words tagged correctly by at least one of the three models divided by the total number of unknown words. The upper bound is 91.10% for the training data and 91.39% for the test data. Table 7 reports the results for the combined model. The overall precision of the model reaches 89.32% in the training data and 89.00% in the test data, close to the upper bounds.

6 Discussion and Conclusion

The results indicate that the three models have different strengths and weaknesses. Using rules that do not overgenerate and that are sensitive to the type, length, and internal structure of unknown words,

Chars	Training	Test
2	73.27	74.47
3	97.15	97.25
4	98.78	99.20
5+	100	100
Total	89.32	89.00

Table 7: Results for the combined model

the rule-based model achieves high precision for all words and high recall for longer words, but recall for disyllabic words is low. The trigram model makes use of the contextual information of unknown words and solves the recall problem, but its precision is relatively low. Wu and Jiang's (2000) model complements the other two, as it achieves a higher recall than the rule-based model and a higher precision than the trigram model for disyllabic words. The combined model outperforms each individual model by effectively combining their strengths.

The results challenge the reasons given in previous studies for rejecting the rule-based model. Overgeneration is a problem only if one attempts to write rules to cover the complete set of unknown words. It can be controlled if one prefers precision over recall. To this end, the internal structure of the unknown words provides very useful information. Results for the rule-based model also suggest that as unknown words become longer and the fluidity of their component words/morphemes reduces, they become more predictable and generalizable by rules.

The results achieved in this study prove a significant improvement over those reported in previous studies. To our knowledge, the best result on this task was reported by Chen et al. (1997), which was 69.13%. However, they considered fourteen POS categories, whereas we examined only eight. This difference is brought about by the different tagsets used in the different corpora and the decision to include or exclude proper names and numeric type compounds. To make the results more comparable, we replicated their model, and the results we found were consistent with what they reported, i.e., 69.12% for our training data and 68.79% for our test data, as opposed to our 89.32% and 89% respectively.

Several avenues can be taken for future research. First, it will be useful to identify a statistical model that achieves higher precision for disyllabic words, as this seems to be the bottleneck. It will also be relevant to apply advanced statistical models that can incorporate various useful information to this task, e.g., the maximum entropy model (Ratnaparkhi, 1996). Second, for better evaluation, it would be helpful to use a larger corpus and evaluate the individual models on a held-out dataset, to compare our model with other models on more comparable datasets, and to test the model on other logographic languages. Third, some grammatical constraints may be used for the detection and correction of tagging errors in a post-processing step. Finally, as part of a bigger project on Chinese unknown word resolution, we would like to see how well the general methodology used and the specifics acquired in this task can benefit the identification and sense-tagging of unknown words.

References

Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing*, pages 224–231.

Keh-Jiann Chen and Ming-Hong Bai. 1998. Unknown word detection for Chinese by a corpus-based learning method. *International Journal of Computational Linguistics and Chinese Language Processing*, 3(1):27–44.

Chao-Jan Chen, Ming-Hong Bai, and Keh-Jiann Chen. 1997. Category guessing for Chinese unknown words. In *Proceedings of NLPRS*, pages 35–40.

Chooi-Ling Goh. 2003. Chinese unknown word identification by combining statistical models. Master's thesis, Nara Institute of Science and Technology, Japan.

Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of EMNLP*, pages 133–142.

Richard Sproat. 2002. Corpus-based methods in Chinese morphology. Tutorial at the 19th COLING.

Andy Wu and Zixin Jiang. 2000. Statistically-enhanced new word identification in a rule-based Chinese system. In *Proceedings of the 2nd Chinese Language Processing Workshop*, pages 46–51.

Shiwen Yu, Huiming Duan, Xuefeng Zhu, and Bing Sun. 2002. The basic processing of Contemporary Chinese Corpus at Peking University. Technical report, Institute of Computational Linguistics, Peking University, Beijing, China.

Understanding the thematic structure of the Qur'an: an exploratory multivariate approach

Naglaa Thabet
School of English Literature, Language and Linguistics
University of Newcastle
Newcastle upon Tyne, UK, NE1 7RU
n.a.thabet@ncl.ac.uk

Abstract

In this paper, we develop a methodology for discovering the thematic structure of the Qur'an based on a fundamental idea in data mining and related disciplines: that, with respect to some collection of texts, the lexical frequency profiles of the individual texts are a good indicator of their conceptual content, and thus provide a reliable criterion for their classification relative to one another. This idea is applied to the discovery of thematic interrelationships among the suras (chapters) of the Qur'an by abstracting lexical frequency data from them and then applying hierarchical cluster analysis to that data. The results reported here indicate that the proposed methodology yields usable results in understanding the Qur'an on the basis of its lexical semantics.

1 Introduction

The Qur'an is one of the great religious books of the world, and is at the heart of Islamic culture. Careful, well-informed interpretation of the Qur'an is fundamental both to the faith of millions of Muslims throughout the world, and to the non-Islamic world's understanding of their religion. There is a long tradition of scholarly quranic interpretation, and it has necessarily been based on traditional literary-historical methods of manual textual exegesis. However, developments in electronic text representation and analysis now offer the opportunity of applying the technologies of newly-emerging research areas such as data mining (Hand et al., 2001) to the interpretation of the Qur'an. Studies on computational analyses of the Qur'an are almost lacking. Contributions to this field include the development of a morphological analyser for the Qur'an (Dror et al., 2004).

The Qur'an consists of 114 chapters called suras which range in length from the shortest, Al-Kawthar, consisting of 4 ayat (verses) to the longest, Al-Baqarah, consisting of 286 ayat. There is no obvious reason why the suras are sequenced as they are in the text. They are not in chronological order, and seem, in fact, to be ordered roughly by length, from longest at the beginning of the text to shortest at the end. Given this, apparently arbitrary sequencing, one of the first steps in interpreting the Qur'an as a whole must be to discover thematic interrelationships among the suras. The present paper proposes a methodology for doing this using exploratory multivariate analysis.

The paper is in five parts; the first part is the introduction. The second presents the quranic text and the data preparation prior to the analysis. The third part deals with the application of cluster analysis techniques to the Qur'an and the interpretation of the results. The fourth part draws the conclusion and suggests future research to be undertaken.

2 Data

The data for this study is based on an electronic version of the Qur'an produced by Muslimnet[1]. This version is a Western alphabetic transliteration of the Arabic orthography. The data is transliterated into Latin based ASCII characters, mostly with single-symbol equivalents of the Arabic phonemes and by replacing diacritics and

[1] http://www.usc.edu/dept/MSA/quran/transliteration/

Proceedings of the ACL Student Research Workshop, pages 7–12,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

glyphs which represent short vowels in Arabic orthography with appropriate Roman letters.

A frequency matrix F is constructed in which the rows are the suras, the columns are lexical items, and every cell F_{ij} contains an integer that represents the number of times lexical item j occurs in sura i. Construction of such a matrix is straightforward in principle, but in practice some well known issues arise.

2.1 Tokenization

Given that one wants to count words, what is a word? The answer is surprisingly difficult, and is a traditional problem both in linguistics and in natural language processing (Manning and Shütze, 1999). Even confined to written language, as here, two issues arise:

- Word segmentation: In English text, the commonsensical view that a word is the string of letters bounded by punctuation and/or white space is quite robust, but it is less so for other languages.
- Stemming: In languages with a significant element of morphological elaboration of words stems, do the various morphological variants of a given stem count as different words?

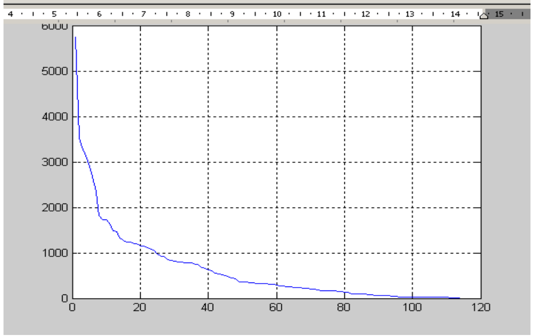
For present purposes, the words segmentation problem is easily resolved in that the Qur'an's orthography is such that words can be reliably identified using the 'string of letters between punctuation and/or white space' criterion. With regard to stemming, morphological variants are treated as single word types, and to achieve this, the electronic text of the Qur'an was processed using a purpose-built stemmer whose characteristics and performance are described in Thabet (2004).

2.2 Keyword selection

Function words like determiners and prepositions were removed from the text, and only content words were retained. In addition, the (many) words with frequency 1 were removed, since these cannot contribute to determination of relationship among suras.

2.3 Standardization for text length

The introduction noted that the suras vary in length from fewer than a dozen to several thousand words. The following plot of number of content words per sura, sorted in order of descending magnitude.



"Figure 1. Plot of number of words per sura"

Clearly, given a word with some probability of occurrence, it is more likely to occur in a long text than a short one. In order to compare the suras meaningfully on the basis of their word frequency profiles, the raw frequencies have to be adjusted to compensate for sura length variation. This was done on the following basis:

$$freq\,'(F_{ij}) = freq\,(F_{ij}) \times \left(\frac{\mu}{l}\right)$$

where *freq*' is the adjusted frequency, F_{ij} is the value at the (i,j) coordinates of the data matrix F, *freq* is the raw frequency, μ is the mean number of words per sura across all 114 suras, and *l* is the number of words in sura *i*.

That said, it has also to be observed that, as text length decreases, so does the probability that any given word will occur even once in it, and its frequency vector will therefore become increasingly sparse, consisting mainly of zeros. Because 0 is non-adjustable, functions that compensate for variable text length generate increasingly unreliable results as length decreases. In the present application, therefore, only relatively long suras are considered for analysis, and more specifically those 24 containing 1000 or more content words.

Sura name	Words	Sura name	Words
Al-Baqarah	5739	Al-Israa	1464
Al-Imran	3316	Al-Kahf	1489
Al-Nisa	3543	Ta-Ha	1265
Al-Maidah	2681	Al-Anbiyaa	1077
Al-An'am	2895	Al-Hajj	1195
Al-A'raf	3127	Al-Nur	1236
Al-Anfal	1156	Al-Shu'araa	1208

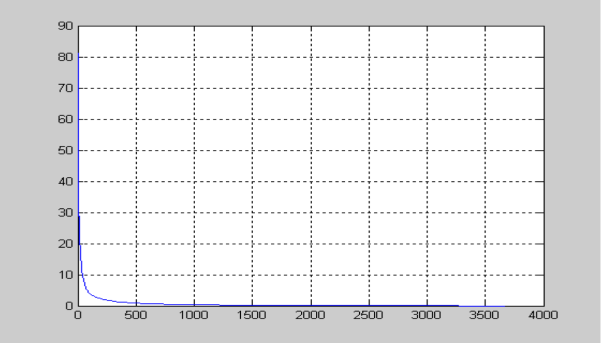
Al-Tawba	2345	Al-Naml	1069
Yunus	1732	Al-Qasas	1332
Hud	1809	Al-Ahzab	1239
Yusuf	1665	Al-Zumr	1107
Al-Nahl	1729	Ghafir	1156

“Table 1. Suras with more than 1000 words”

The choice of 1000 as the length threshold is arbitrary. Arbitrariness does no harm in a methodological paper such as this one. Clearly, however, any legitimate analysis of the Qur’an using this methodology will have to face the problem of which suras, if any, to exclude on length grounds in a principled way.

2.4 Dimensionality reduction

After function words and words with frequency 1 were eliminated and morphological variants stemmed, 3672 significant ‘content’ words remained, requiring a matrix with 3672 columns. Given only 24 data points, this results in an extremely sparsely populated data space whose dimensionality should be reduced as much as possible consistent with the need to represent the data domain adequately. For a discussion of dimensionality issues in data analysis see Verleysen (2003). To do this, the variances for all 3672 columns of the frequency matrix F were calculated, sorted in decreasing order of magnitude, and plotted:



“Figure 2. Plot of variances for 3762 columns”

This is what one would expect from the typical word frequency distribution in natural language text articulated by Zipf’s Law (Manning and Shütze, 1999; 20-29): almost all the variance in the data is concentrated in a small number of variables –the 500 or so on the left. The variance in the remainder is so small that it cannot contribute significantly to differentiating the data matrix rows and, therefore, can be disregarded. The matrix is thus truncated to 500 variables / columns, resulting in a 24 x 500 matrix for cluster analysis.

3 Analysis

3.1 Hierarchical cluster analysis

Cluster analysis aims to identify and graphically represent nonrandomness in the distribution of vectors in a data space such that intra-group distance is small relative to the dimensions of the space, and inter-group distance is relatively large. Detailed accounts of hierarchical cluster analysis are in Everitt (2001), Gordon (1999; 69-109), and Gore (2000). For briefer discussions see Dunn and Everitt (2001; 125-160), Hair et al. (1998; 469-518), Flynn et al. (1999; 275-9), Kachigan (1991; 261-70), Oakes (1998; 110-120). There are two main varieties: hierarchical and nonhierarchical. The former aims not only to discover and graphically represent clusters, but also to show constituency relations among data items and data item clusters as ‘dendrograms’ or trees.

Hierarchical analysis uses relative proximity among vectors as the basis for clustering, where proximity can be measured in terms either of similarity or of distance; distance is most often used, and is adopted here. Assuming the existence of a data matrix containing numerical values such as the one described above, construction of a distance-based cluster tree is a two-stage procedure. In the first step, a table of distances between data items, that is, between row vectors of the data matrix, is generated. A frequently used measure is the Euclidean; there the distance between vectors A and B is calculated using the well known formula: $length(z) = \sqrt{(length(x))^2 + (length(y))^2}$, but there are many others as in Gordon (1999; 15-3) and Flynn et al. (1999; 271-4).

The second step then uses the distance table to build clusters with the following generic algorithm:

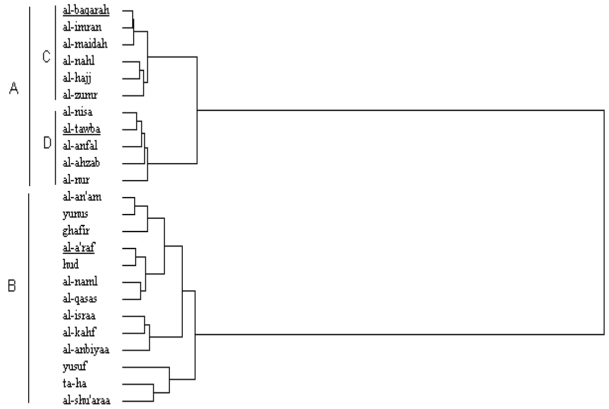
- Initially, every data vector is its own cluster.
- Using as many steps as necessary, at each step combine the two nearest clusters to form a new, composite cluster, thus reducing the number of clusters by 1.

- When only one cluster remains, incorporating all the cases in the distance matrix, stop.

An example of a tree generated by this procedure follows in the next section.

3.2 Cluster analysis of the quranic data

The above generic clustering algorithm glosses over an important point: determination of distances between data items is given by the distance table, but the distances between composite clusters is not, and needs to be calculated at each step. How are these distances calculated? There is no single answer. Various definitions of what constitutes a cluster exist, and, in any given application, one is free to choose among them. The problem is that the numerous combinations of distance measure and cluster definition available to the researcher typically generate different analyses of the same data, and there is currently no objective criterion for choosing among them. This indeterminacy is, in fact, the main drawback in using hierarchical clustering for data analysis. The present discussion sidesteps this important issue on the grounds that its aim is methodological: the intention at this stage of research is not to present a definitive cluster analysis of the Qur'an, but to develop an approach to doing so. One particular combination of distance measure and cluster definition was therefore chosen at random and applied to the data: squared Euclidean distance and Ward's Method. The result was as follows (the A - D labels on the left are for later reference):



"Figure 3. Tree generated by cluster analysis"

3.3 Interpretation

Given that the lengths of the vertical lines in the above tree represent relative distance between subclusters, interpretation of the tree in terms of the constituency relations among suras is obvious: there are two main subclusters A and B; A consists of two subclusters C and D, and so on. Knowing the constituency structure of the suras is a necessary precondition for understanding their thematic interrelationships –the object of this exercise—but it is not sufficient because it provides no information about the thematic characteristics of the clusters and the thematic differences between and among them. This information can be derived from the lexical semantics of the column labels in the data matrix, as follows.

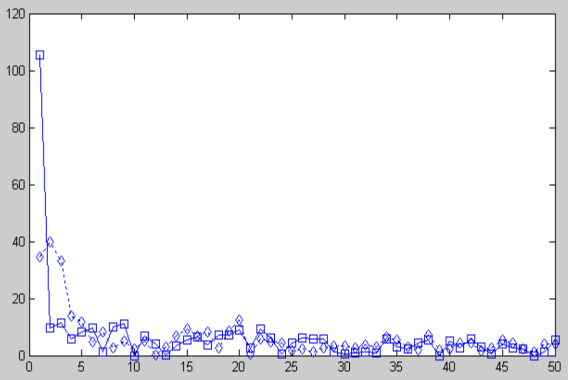
Each row in the data matrix is a lexical frequency profile of the corresponding sura. Since hierarchical analysis clusters the rows of the data matrix in terms of their relative distance from one another in the data space, it follows that the lexical frequency profiles in a given cluster G are closer to one another than to any other profile in the data set. The profiles of G can be summarized by a vector *s* whose dimensionality is that of the data, and each of whose elements contains the mean of the frequencies for the corresponding data matrix column:

$$s_j = \left(\sum_{i=1..n} F_{i,j}\right) / n$$

where j is the index to the jth element of *s*, i indexes the rows of the data matrix F, and n is the total number of rows in cluster G. If *s* is interpreted in terms of the semantics of the matrix column labels, it becomes a thematic profile for G: relative to the frequency range of *s*, a high-frequency word indicates that the suras which constitute G are concerned with the denotation of that word, and the indication for a low-frequency one is the obverse. Such a thematic profile can be constructed for each subcluster, and thematic differences between subclusters can be derived by comparing the profiles.

The general procedure for thematic interpretation of the cluster tree, therefore, is to work through the levels of the tree from the top, constructing and comparing thematic profiles for the subclusters at each level as far down the tree as is felt to be useful.

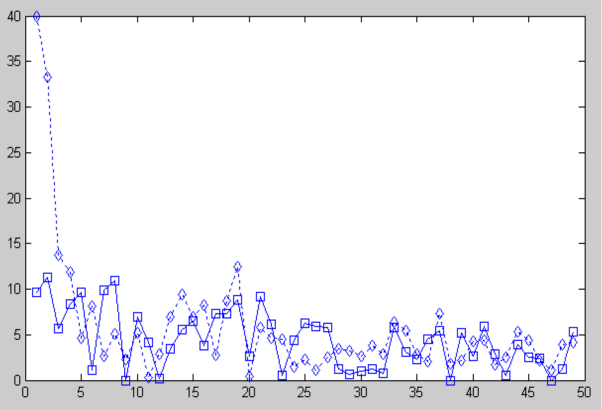
By way of example, consider the application of this general procedure to subtrees A and B in the above cluster tree. Two mean frequency vectors were generated, one for the component suras of cluster A, and one for those of cluster B. These were then plotted relative to one another; the solid line with square nodes represents cluster A, and the dotted line with diamond nodes cluster B; for clarity, only the 50 highest-variance variables are shown, in descending order of magnitude from the left:



“Figure 4. Initial plot of groups A and B”

The suras of cluster A are strikingly more concerned with the denotation of variable 1, the highest-variance variable in the Qur’an, than the suras of cluster B. This variable is the lexical item ‘Allah’, which is central in Islam; the disparity in the frequency of its occurrence in A and B is the first significant finding of the proposed methodology.

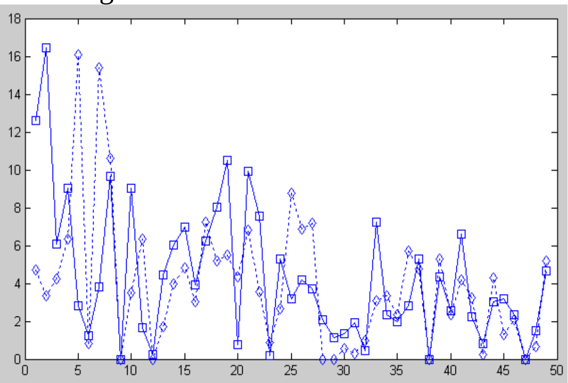
The scaling of the ‘Allah’ variable dominates all the other variables. To gain resolution for the others, ‘Allah’ was eliminated from the lexical frequency vectors, and the vectors were re-plotted:



“Figure 5. Re-plotting of groups A and B”

Awareness of the historical background of the Qur’an’s revelation to Mohamed is crucial at this point of interpretation. The suras revealed to Mohamed before his migration to Madinah are called Makkan suras, whereas those sent down after the migration are called Madinan. Makkan suras stress the unity and majesty of Allah, promise paradise for the righteous and warn wrongdoers of their punishment, confirm the prophethood of Mohamed and the coming resurrection, and remind humanity of the past prophets and events of their times. On the other hand, the Madinan suras outline ritualistic aspects of Islam, lay down moral and ethical codes, criminal laws, social, economic and state policies, and give guidelines for foreign relations and regulations for battles and captives of war. The results emerging from the initial clustering classification in figure 3 highlighted such thematic distiction. All the suras in cluster A are Madinan suras (apart from ‘Al-Nahl’ and ‘Al-Zumr’ which are Makkan suras; yet they do contain some verses that were revealed in Madina). The 13 suras which compose cluster B are all Madinan suras. The distribution of the variables (keywords) in figure 5 is also highly significant, e.g. variable 1 ‘qAl’ (said) is prevalent in the suras of cluster B. The suras of this group contain many narratives which illustrate important aspects of the quranic message, remind of the earlier prophets and their struggle and strengthen Prophet Mohamed’s message of Islam. This signifies the use of the verb ‘qAl’ as a keyword in narrative style. Variable 4 ‘qul’ (say, imperative) is more frequent in group B than group A. Most of the passages of these Makkan suras start with the word ‘qul’, which is an instruction to Prophet Mohamed to address the words following this introduction to his audience in a particular situation, such as in reply to a question that has been raised, or an assertion of a matter of belief. The use of this word was appropriate with Mohamed’s invitation to belief in God and Islam in Makkan suras. Variable 5 ‘mu/min’ (believers), variable 8 ‘Aman’ (believe) and variable 24 ‘ittaq’ (have faith) highly occur in group A. These are the Madinan suras in which prophet Mohamed addresses those who already believed in his message and hence focusing on introducing them to the other social and ethical aspects of Islam. Other variables prevelant in group B are variables 14 and 28 ‘AyAt , Ayat’ (signs/sign). The use of the two words was very important for Prophet Mohamed in the early phase of Islam in Makkah. He had to provide evidence and signs to people to support his invitation to belief in Allah and Islam.

The same procedure of clustering can be applied to the subclusters of A and B. Again, the scaling of Allah' dominates, and removing it from the mean frequency vectors gives better resolution for the remaining variables. Plotting the lexical frequency vectors for C and D, for example, yields the following:



"Figure 6. Plot of groups C and D"

Results from figure 6 are also supportive of the thematic structure of each group. Suras of group C are more abundant in the use of narratives and addressing Mohamed to provide evidence of his message to people. Suras of group B are more concerned with addressing believers about the reward for their righteous conduct. Occurrences of relative variables to those themes are indicative of such distinction.

4 Conclusion and future directions

The above preliminary results indicate that construction and semantic interpretation of cluster trees based on lexical frequency is a useful approach to discovering thematic interrelationships among the suras that constitute the Qur'an. Usable results can, however, only be generated when two main issues have been resolved:

- Standardization of the data for variation in sura length, as discussed in section (2.3)
- Variation in tree structure with different combinations of distance measure and cluster definition, as discussed in section (3.2)

Work on these is ongoing.

To conclude, hierarchical cluster analysis is known to give different results for different distance measure / clustering rule combinations, and consequently cannot be relied on to provide a definitive analysis. The next step is to see if interpretation of the principal components of a principal component analysis of the frequency matrix yields results consistent with those described above. Another multivariate method to be applied to the data is multidimensional scaling. In the longer term, the aim is to use nonlinear methods such as the self organizing map in order to take account of any nonlinearities in the data.

References

Dror, J., Shaharabani, D., Talmon, R., Wintner, S. 2004. *Morphological Analysis of the Qur'an*. Literary and Linguistic Computing, 19(4):431-452.

Dunn, G. and Everitt, B. (2001). *Applied Multivariate Data Analysis*, 2nd ed. Arnold, London.

Everitt, B. (2001). *Cluster Analysis*, 4th ed. Arnold, London.

Flynn, P., Jain, A., and Murty, M. (1999). *Data clustering: A review*. In: ACM Computing Surveys 31, 264–323.

Gordon, A. (1999). *Classification*, 2nd ed. Chapman & Hall, London.

Gore, P. (2000). *Cluster Analysis*. In H. E. A. Tinsley & S. D. Brown (Eds.), Handbook of applied multivariate statistics and mathematical modeling (pp. 297-321). Academic Press, San Diego, CA

Hair, H., Anderson, J., Black, W. and Tatham, R. (1998). *Multivariate Data Analysis*, 5th ed. Prentice-Hall International, London.

Hand, D., Mannila, H., Smyth, P. (2001). *Principles of Data Mining*, MIT Press.

Kachigan, S. (1991). *Multivariate Statistical Analysis. A conceptual introduction*. Radius Press, New York

Manning, C. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. Cambridge, Mass, MIT Press.

Oakes, M. (1998). *Statistics for Corpus Linguistics*. Edinburgh University Press, Edinburgh

Thabet, N. (2004). *"Stemming the Qur'an*". In Proceedings of Arabic Script-Based Languages Workshop, COLING-04, Switzerland, August 2004.

Verleysen, M. (2003). *Learning high-dimensional data*. In: *Limitations and future trends in neural computation*. IOS Press, Amesterdam, pp141-162.

An Extensive Empirical Study of Collocation Extraction Methods

Pavel Pecina
Institute of Formal and Applied Linguistics
Charles University, Prague, Czech Republic
pecina@ufal.mff.cuni.cz

Abstract

This paper presents a *status quo* of an ongoing research study of collocations – an essential linguistic phenomenon having a wide spectrum of applications in the field of natural language processing. The core of the work is an empirical evaluation of a comprehensive list of automatic collocation extraction methods using precision-recall measures and a proposal of a new approach integrating multiple basic methods and statistical classification. We demonstrate that combining multiple independent techniques leads to a significant performance improvement in comparison with individual basic methods.

1 Introduction and motivation

Natural language cannot be simply reduced to lexicon and syntax. The fact that individual words cannot be combined freely or randomly is common for most natural languages. The ability of a word to combine with other words can be expressed either *intensionally* or *extensionally*. The former case refers to *valency*. Instances of the latter case are called *collocations* (Čermák and Holub, 1982). The term collocation has several other definitions but none of them is widely accepted. Most attempts are based on a characteristic property of collocations: *non-compositionality*. Choueka (1988) defines a collocational expression as "a syntactic and semantic unit whose exact and unambiguous meaning or connotation cannot be derived directly from the meaning or connotation of its components".

The term collocation has both linguistic and lexicographic character. It covers a wide range of lexical phenomena, such as phrasal verbs, light verb compounds, idioms, stock phrases, technological expressions, and proper names. Collocations are of high importance for many applications in the field of NLP. The most desirable ones are machine translation, word sense disambiguation, language generation, and information retrieval. The recent availability of large amounts of textual data has attracted interest in automatic collocation extraction from text. In the last thirty years a number of different methods employing various association measures have been proposed. Overview of the most widely used techniques is given e.g. in (Manning and Schütze, 1999) or (Pearce, 2002). Several researches also attempted to compare existing methods and suggested different evaluation schemes, e.g Kita (1994) or Evert (2001). A comprehensive study of statistical aspects of word cooccurrences can be found in (Evert, 2004).

In this paper we present a compendium of 84 methods for automatic collocation extraction. They came from different research areas and some of them have not been used for this purpose yet. A brief overview of these methods is followed by their comparative evaluation against manually annotated data by the means of precision and recall measures. In the end we propose a statistical classification method for combining multiple methods and demonstrate a substantial performance improvement.

In our research we focus on two-word (*bigram*) collocations, mainly for the reason that experiments with longer expressions would require processing of much larger amounts of data and limited scalability of some methods to high order n-grams. The experiments are performed on Czech data.

Proceedings of the ACL Student Research Workshop, pages 13–18,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

2 Collocation extraction

Most methods for collocation extraction are based on verification of typical collocation properties. These properties are formally described by mathematical formulas that determine the degree of association between components of collocation. Such formulas are called *association measures* and compute an *association score* for each collocation candidate extracted from a corpus. The scores indicate a chance of a candidate to be a collocation. They can be used for ranking or for classification – by setting a threshold. Finding such a threshold depends on the intended application.

The most widely tested property of collocations is *non-compositionality*: If words occur together more often than by a chance, then this is the evidence that they have a special function that is not simply explained as a result of their combination (Manning and Schütze, 1999). We think of a corpus as a randomly generated sequence of words that is viewed as a sequence of word pairs. Occurrence frequencies of these bigrams are extracted and kept in contingency tables (Table 1a). Values from these tables are used in several association measures that reflect how much the word coocurrence is accidental. A list of such measures is given in Table 2 and includes: estimation of bigram and unigram probabilities (rows 3–5), mutual information and derived measures (6–11), statistical tests of independence (12–16), likelihood measures (17–18), and various other heuristic association measures and coefficients (19–57).

Another frequently tested property is taken directly from the definition that a collocation is a *syntactic and semantic unit*. For each bigram occurring in the corpus, information of its *empirical context* (frequencies of open-class words occurring within a specified context window) and left and right *immediate contexts* (frequencies of words immediately preceding or following the bigram) is extracted (Table 1b). By determining the entropy of the immediate contexts of a word sequence, the association measures rank collocations according to the assumption that they occur as units in a (information-theoretically) noisy environment (Shimohata et al., 1997) (58–62). By comparing empirical contexts of a word sequence and its components, the association measures rank collocations according to the assumption that semantically non-compositional expressions typically occur in different contexts than their components (Zhai, 1997). Measures (63–76) have information theory background and measures (77–84) are adopted from the field of information retrieval. Context association measures are mainly used for extracting idioms.

a)

$a = f(xy)$	$b = f(x\bar{y})$	$f(x*)$
$c = f(\bar{x}y)$	$d = f(\bar{x}\bar{y})$	$f(\bar{x}*)$
$f(*y)$	$f(*\bar{y})$	N

b)

C_w	empirical context of w
C_{xy}	empirical context of xy
C^l_{xy}	left immediate context of xy
C^r_{xy}	right immediate context of xy

Table 1: a) A contingency table with observed frequencies and marginal frequencies for a bigram xy; $\bar{w}$ stands for any word except w; $*$ stands for any word; N is a total number of bigrams. The table cells are sometimes referred as f_{ij}. Statistical tests of independence work with contingency tables of expected frequencies $\hat{f}(xy) = f(x*)f(*y)/N$. b) Different notions of empirical contexts.

Besides all the association measures described above, we also take into account other recommended measures (1–2) (Manning and Schütze, 1999) and some basic linguistic characteristics used for filtering non-collocations (85–87). This information can be obtained automatically from morphological taggers and syntactic parsers available with reasonably high accuracy for many languages.

3 Empirical evaluation

Evaluation of collocation extraction methods is a complicated task. On one hand, different applications require different setting of association score thresholds. On the other hand, methods give different results within different ranges of their association scores. We need a complex evaluation scheme covering all demands. In such a case, Evert (2001) and other authors suggest using *precision* and *recall* measures on a full reference data or on *n-best* lists.

Data. All the presented experiments were performed on morphologically and syntactically annotated Czech text from the *Prague Dependency Treebank* (PDT) (Hajič et al., 2001). Dependency trees were broken down into *dependency bigrams* consisting of: *lemmas* and *part-of-speech* of the components, and *type of dependence* between the components.

For each bigram type we counted frequencies in its contingency table, extracted empirical and immediate contexts, and computed all the 84 association measures from Table 2. We processed 81 614 sen-

# *Name*	*Formula*
1. **Mean component offset**	$\frac{1}{n}\sum_{i=1}^{n} d_i$
2. **Variance component offset**	$\frac{1}{n-1}\sum_{i=1}^{n}\left(d_i-\bar{d}\right)^2$
3. **Joint probability**	$P(xy)$
4. **Conditional probability**	$P(y\mid x)$
5. **Reverse conditional prob.**	$P(x\mid y)$
⋆6. **Pointwise mutual inform.**	$\log\frac{P(xy)}{P(x*)P(*y)}$
7. **Mutual dependency (*MD*)**	$\log\frac{P(xy)^2}{P(x*)P(*y)}$
8. **Log frequency biased *MD***	$\log\frac{P(xy)^2}{P(x*)P(*y)}+\log P(xy)$
9. **Normalized expectation**	$\frac{2f(xy)}{f(x*)+f(*y)}$
⋆10. **Mutual expectation**	$\frac{2f(xy)}{f(x*)+f(*y)}\cdot P(xy)$
11. **Salience**	$\log\frac{P(xy)^2}{P(x*)P(*y)}\cdot\log f(xy)$
12. **Pearson's χ^2 test**	$\sum_{i,j}\frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}}$
13. **Fisher's exact test**	$\frac{f(x*)!f(\bar{x}*)!f(*y)!f(*\bar{y})!}{N!f(xy)!f(x\bar{y})!f(\bar{x}y)!f(\bar{x}\bar{y})!}$
14. **t test**	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{f(xy)(1-(f(xy)/N))}}$
15. **z score**	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{\hat{f}(xy)(1-(\hat{f}(xy)/N))}}$
16. **Poison significance measure**	$\frac{\hat{f}(xy)-f(xy)\log\hat{f}(xy)+\log f(xy)!}{\log N}$
17. **Log likelihood ratio**	$-2\sum_{i,j} f_{ij}\log\frac{f_{ij}}{\hat{f}_{ij}}$
18. **Squared log likelihood ratio**	$-2\sum_{i,j}\frac{\log {f_{ij}}^2}{\hat{f}_{ij}}$
Association coefficients:	
19. **Russel-Rao**	$\frac{a}{a+b+c+d}$
20. **Sokal-Michiner**	$\frac{a+d}{a+b+c+d}$
⋆21. **Rogers-Tanimoto**	$\frac{a+d}{a+2b+2c+d}$
22. **Hamann**	$\frac{(a+d)-(b+c)}{a+b+c+d}$
23. **Third Sokal-Sneath**	$\frac{b+c}{a+d}$
24. **Jaccard**	$\frac{a}{a+b+c}$
⋆25. **First Kulczynsky**	$\frac{a}{b+c}$
26. **Second Sokal-Sneath**	$\frac{a}{a+2(b+c)}$
27. **Second Kulczynski**	$\frac{1}{2}\left(\frac{a}{a+b}+\frac{a}{a+c}\right)$
28. **Fourth Sokal-Sneath**	$\frac{1}{4}\left(\frac{a}{a+b}+\frac{a}{a+c}+\frac{d}{d+b}+\frac{d}{d+c}\right)$
29. **Odds ratio**	$\frac{ad}{bc}$
30. **Yulle's ω**	$\frac{\sqrt{ad}-\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}$
⋆31. **Yulle's Q**	$\frac{ad-bc}{ad+bc}$
32. **Driver-Kroeber**	$\frac{a}{\sqrt{(a+b)(a+c)}}$
33. **Fifth Sokal-Sneath**	$\frac{ad}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
34. **Pearson**	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
35. **Baroni-Urbani**	$\frac{a+\sqrt{ad}}{a+b+c+\sqrt{ad}}$
36. **Braun-Blanquet**	$\frac{a}{\max(a+b,a+c)}$
37. **Simpson**	$\frac{a}{\min(a+b,a+c)}$
38. **Michael**	$\frac{4(ad-bc)}{(a+d)^2+(b+c)^2}$
39. **Mountford**	$\frac{2a}{2bc+ab+ac}$
40. **Fager**	$\frac{a}{\sqrt{(a+b)(a+c)}}-\frac{1}{2}\max(b,\,c)$
41. **Unigram subtuples**	$\log\frac{ad}{bc}-3.29\sqrt{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$
42. ***U* cost**	$\log(1+\frac{\min(b,c)+a}{\max(b,c)+a})$
43. ***S* cost**	$\log(1+\frac{\min(b,c)}{a+1})^{-\frac{1}{2}}$
44. ***R* cost**	$\log(1+\frac{a}{a+b})\cdot\log(1+\frac{a}{a+c})$
45. ***T* combined cost**	$\sqrt{U\times S\times R}$
46. **Phi**	$\frac{P(xy)-P(x*)P(*y)}{\sqrt{P(x*)P(*y)(1-P(x*))(1-P(*y))}}$
47. **Kappa**	$\frac{P(xy)+P(\bar{x}\bar{y})-P(x*)P(*y)-P(\bar{x}*)P(*\bar{y})}{1-P(x*)P(*y)-P(\bar{x}*)P(*\bar{y})}$
48. ***J* measure**	$\max[P(xy)\log\frac{P(y\mid x)}{P(*y)}+P(x\bar{y})\log\frac{P(\bar{y}\mid x)}{P(*\bar{y})},$ $P(xy)\log\frac{P(x\mid y)}{P(x*)}+P(\bar{x}y)\log\frac{P(\bar{x}\mid y)}{P(\bar{x}*)}]$
49. **Gini index**	$\max[P(x*)(P(y\mid x)^2+P(\bar{y}\mid x)^2)-P(*y)^2$ $+P(\bar{x*})(P(y\mid\bar{x})^2+P(\bar{y}\mid\bar{x})^2)-P(*\bar{y})^2,$ $P(*y)(P(x\mid y)^2+P(\bar{x}\mid y)^2)-P(x*)^2$ $+P(*\bar{y})(P(x\mid\bar{y})^2+P(\bar{x}\mid\bar{y})^2)-P(\bar{x}*)^2]$
50. **Confidence**	$\max[P(y\mid x),\,P(x\mid y)]$
51. **Laplace**	$\max[\frac{NP(xy)+1}{NP(x*)+2},\,\frac{NP(xy)+1}{NP(*y)+2}]$
52. **Conviction**	$\max[\frac{P(x*)P(*y)}{P(x\bar{y})},\,\frac{P(\bar{x}*)P(*y)}{P(\bar{x}y)}]$
53. **Piatersky-Shapiro**	$P(xy)-P(x*)P(*y)$
54. **Certainity factor**	$\max[\frac{P(y\mid x)-P(*y)}{1-P(*y)},\,\frac{P(x\mid y)-P(x*)}{1-P(x*)}]$
55. **Added value (*AV*)**	$\max[P(y\mid x)-P(*y),\,P(x\mid y)-P(x*)]$
⋆56. **Collective strength**	$\frac{P(xy)+P(\bar{x}\bar{y})}{P(x*)P(y)+P(\bar{x}*)P(*y)}\cdot$ $\frac{1-P(x*)P(*y)-P(\bar{x}*)P(*y)}{1-P(xy)-P(\bar{x}\bar{y})}$
57. **Klosgen**	$\sqrt{P(xy)}\cdot AV$
Context measures:	
⋆58. **Context entropy**	$-\sum_w P(w\mid C_{xy})\log P(w\mid C_{xy})$
59. **Left context entropy**	$-\sum_w P(w\mid C^l_{xy})\log P(w\mid C^l_{xy})$
60. **Right context entropy**	$-\sum_w P(w\mid C^r_{xy})\log P(w\mid C^r_{xy})$
⋆61. **Left context divergence**	$P(x*)\log P(x*)$ $-\sum_w P(w\mid C^l_{xy})\log P(w\mid C^l_{xy})$
62. **Right context divergence**	$P(*y)\log P(*y)$ $-\sum_w P(w\mid C^r_{xy})\log P(w\mid C^r_{xy})$
63. **Cross entropy**	$-\sum_w P(w\mid C_x)\log P(w\mid C_y)$
64. **Reverse cross entropy**	$-\sum_w P(w\mid C_y)\log P(w\mid C_x)$
65. **Intersection measure**	$\frac{2\lvert C_x\cap C_y\rvert}{\lvert C_x\rvert+\lvert C_y\rvert}$
66. **Euclidean norm**	$\sqrt{\sum_w(P(w\mid C_x)-P(w\mid C_y))^2}$
67. **Cosine norm**	$\frac{\sum_w P(w\mid C_x)P(w\mid C_y)}{\sum_w P(w\mid C_x)^2\cdot\sum_w P(w\mid C_y)^2}$
68. ***L1* norm**	$\sum_w \lvert P(w\mid C_x)-P(w\mid C_y)\rvert$
69. **Confusion probability**	$\sum_w\frac{P(x\mid C_w)P(y\mid C_w)P(w)}{P(x*)}$
70. **Reverse confusion prob.**	$\sum_w\frac{P(y\mid C_w)P(x\mid C_w)P(w)}{P(*y)}$
⋆71. **Jensen-Shannon diverg.**	$\frac{1}{2}[D(p(w\mid C_x)\Vert\frac{1}{2}(p(w\mid C_x)+p(w\mid C_y)))$ $+D(p(w\mid C_y)\Vert\frac{1}{2}(p(w\mid C_x)+p(w\mid C_y)))]$
72. **Cosine of pointwise *MI***	$\frac{\sum_w MI(w,x)MI(w,y)}{\sqrt{\sum_w MI(w,x)^2}\cdot\sqrt{\sum_w MI(w,y)^2}}$
⋆73. **KL divergence**	$\sum_w P(w\mid C_x)\log\frac{P(w\mid C_x)}{P(w\mid C_y)}$
⋆74. **Reverse KL divergence**	$\sum_w P(w\mid C_y)\log\frac{P(w\mid C_y)}{P(w\mid C_x)}$
75. **Skew divergence**	$D(p(w\mid C_x)\Vert\alpha(w\mid C_y)+(1-\alpha)p(w\mid C_x))$
76. **Reverse skew divergence**	$D(p(w\mid C_y)\Vert\alpha p(w\mid C_x)+(1-\alpha)p(w\mid C_y))$
77. **Phrase word coocurrence**	$\frac{1}{2}\left(\frac{f(x\mid C_{xy})}{f(xy)}+\frac{f(y\mid C_{xy})}{f(xy)}\right)$
78. **Word association**	$\frac{1}{2}\left(\frac{f(x\mid C_y)-f(xy)}{f(xy)}+\frac{f(y\mid C_x)-f(xy)}{f(xy)}\right)$
Cosine context similarity:	$\frac{1}{2}(\cos(\mathbf{c}_x,\mathbf{c}_{xy})+\cos(\mathbf{c}_y,\mathbf{c}_{xy}))$
	$\mathbf{c}_z=(z_i);\ \cos(\mathbf{c}_x,\mathbf{c}_y)=\frac{\sum x_i y_i}{\sqrt{\sum {x_i}^2}\cdot\sqrt{\sum {y_i}^2}}$
⋆79. **in boolean vector space**	$z_i=\delta(f(w_i\mid C_z))$
80. **in tf vector space**	$z_i=f(w_i\mid C_z)$
81. **in $tf\cdot idf$ vector space**	$z_i=f(w_i\mid C_z)\cdot\frac{N}{df(w_i)};\ df(w_i)=\lvert\{x:w_i\,\epsilon\,C_x\}\rvert$
Dice context similarity:	$\frac{1}{2}(\mathrm{dice}(\mathbf{c}_x,\mathbf{c}_{xy})+\mathrm{dice}(\mathbf{c}_y,\mathbf{c}_{xy}))$
	$\mathbf{c}_z=(z_i);\ \mathrm{dice}(\mathbf{c}_x,\mathbf{c}_y)=\frac{2\sum x_i y_i}{\sum {x_i}^2+\sum {y_i}^2}$
⋆82. **in boolean vector space**	$z_i=\delta(f(w_i\mid C_z))$
⋆83. **in tf vector space**	$z_i=f(w_i\mid C_z)$
⋆84. **in $tf\cdot idf$ vector space**	$z_i=f(w_i\mid C_z)\cdot\frac{N}{df(w_i)};\ df(w_i)=\lvert\{x:w_i\,\epsilon\,C_x\}\rvert$
Linguistic features:	
⋆85. **Part of speech**	{*Adjective:Noun, Noun:Noun, Noun:Verb,* …}
⋆86. **Dependency type**	{*Attribute, Object, Subject,* …}
87. **Dependency structure**	$\{\nearrow,\nwarrow\}$

Table 2: Association measures and linguistic features used in bigram collocation extraction methods. ⋆ denotes those selected by the attribute selection method discussed in Section 4. References can be found at the end of the paper.

tences with 1 255 590 words and obtained a total of 202 171 different dependency bigrams.

Krenn (2000) argues that collocation extraction methods should be evaluated against a reference set of collocations manually extracted from the full candidate data from a corpus. However, we reduced the full candidate data from PDT to 21 597 bigram by filtering out any bigrams which occurred 5 or less times in the data and thus we obtained a *reference data set* which fulfills requirements of a sufficient size and a minimal frequency of observations which is needed for the assumption of normal distribution required by some methods.

We manually processed the entire reference data set and extracted bigrams that were considered to be collocations. At this point we applied *part-of-speech* filtering: First, we identified *POS patterns* that never form a collocation. Second, all dependency bigrams having such a *POS* pattern were removed from the reference data and a final reference set of 8 904 bigrams was created. We no longer consider bigrams with such patterns to be collocation candidates.

This data set contained 2 649 items considered to be collocations. The a priori probability of a bigram to be a collocation was 29.75 %. A stratified one-third subsample of this data was selected as *test data* and used for evaluation and testing purposes in this work. The rest was taken apart and used as *training data* in later experiments.

Evaluation metrics. Since we manually annotated the entire reference data set we could use the suggested *precision* and *recall* measures (and their harmonic mean *F-measure*). A collocation extraction method using any association measure with a given threshold can be considered a classifier and the measures can be computed in the following way:

$$\begin{aligned} Precision &= \frac{\#\ correctly\ classified\ collocations}{\#\ total\ predicted\ as\ collocations} \\ Recall &= \frac{\#\ correctly\ classified\ collocations}{\#\ total\ collocations} \end{aligned}$$

The higher these scores, the better the classifier is. By changing the threshold we can tune the classifier performance and "trade" recall for precision. Therefore, collocation extraction methods can be thoroughly compared by comparing their *precision--recall curves*: The closer the curve to the top right corner, the better the method is.

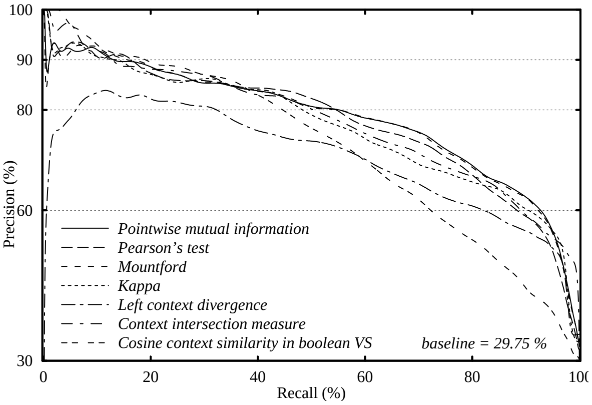


Figure 1: Precision-recall curves for selected assoc. measures.

Results. Presenting individual results for all of the 84 association measures is not possible in a paper of this length. Therefore, we present precision-recall graphs only for the best methods from each group mentioned in Section 2; see Figure 1. The baseline system that classifies bigrams randomly, operates with a precision of 29.75 %. The overall best result was achieved by *Pointwise mutual information*: 30 % recall with 85.5 % precision (F-measure 44.4), 60 % recall with 78.4 % precision (F-measure 68.0), and 90 % recall with 62.5 % precision (F-measure 73.8).

4 Statistical classification

In the previous section we mentioned that collocation extraction is a classification problem. Each method classifies instances of the candidate data set according to the values of an association score. Now we have several association scores for each candidate bigram and want to combine them together to achieve better performance. A motivating example is depicted in Figure 3: Association scores of *Pointwise mutual information* and *Cosine context similarity* are independent enough to be linearly combined to provide better results. Considering all association measures, we deal with a problem of high-dimensional classification into two classes.

In our case, each bigram x is described by the *attribute vector* $\mathbf{x} = (x_1, \ldots, x_{87})$ consisting of linguistic features and association scores from Table 2. Now we look for a function assigning each bigram one class : $f(\mathbf{x}) \rightarrow$ {*collocation*, *non-collocation*}. The result of this approach is similar to setting a threshold of the association score in methods us-

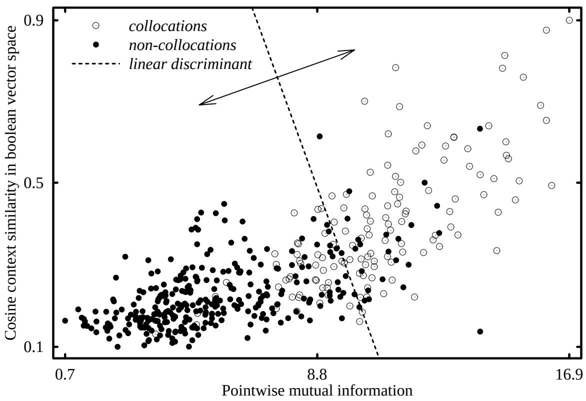


Figure 2: Data visualization in two dimensions. The dashed line denotes a linear discriminant obtained by logistic linear regression. By moving this boundary we can tune the classifier output (a 5 % stratified sample of the test data is displayed).

ing one association measure, which is not very usefull for our purpose. Some classification methods, however, output also the predicted probability $P(x\,is\,collocation)$ that can be considered a regular association measure as described above. Thus, the classification method can be also tuned by changing a threshold of this probability and can be compared with other methods by the same means of precision and recall.

One of the basic classification methods that gives a predicted probability is *Logistic linear regression*. The model defines the predicted probability as:

$$P(x\,is\,collocation) = \frac{\exp^{\beta_0+\beta_1 x_1 \ldots + \beta_n x_n}}{1+\exp^{\beta_0+\beta_1 x_1 \ldots + \beta_n x_n}}$$

where the coefficients β_i are obtained by the *iteratively reweighted least squares* (IRLS) algorithm which solves the weighted least squares problem at each iteration. Categorial attributes need to be transformed to numeric *dummy variables*. It is also recommended to *normalize* all numeric attributes to have zero mean and unit variance.

We employed the datamining software *Weka* by Witten and Frank (2000) in our experiments. As *training data* we used a two-third subsample of the reference data described above. The *test data* was the same as in the evaluation of the basic methods.

By combining all the 87 attributes, we achieved the results displayed in Table 3 and illustrated in Figure 3. At a recall level of 90 % the relative increase in precision was 35.2 % and at a precision level of 90 % the relative increase in recall was impressive 242.3 %.

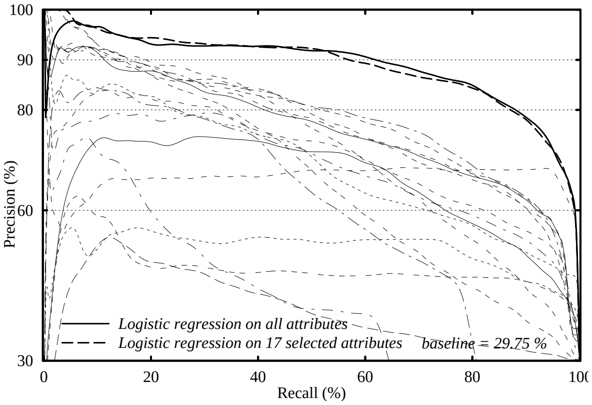


Figure 3: Precision-recall curves of two classifiers based on i) logistic linear regression on the full set of 87 attributes and ii) on the selected subset with 17 attributes. The thin unlabeled curves refer to the methods from the 17 selected attributes

Attribute selection. In the final step of our experiments, we attempted to reduce the attribute space of our data and thus obtain an attribute subset with the same prediction ability. We employed a *greedy stepwise* search method with attribute subset evaluation via logistic regression implemented in Weka. It performs a greedy search through the space of attribute subsets and iteratively merges subsets that give the best results until the performance is no longer improved.

We ended up with a subset consisting of the following 17 attributes: (6, 10, 21, 25, 31, 56, 58, 61, 71, 73, 74, 79, 82, 83, 84, 85, 86) which are also marked in Table 2. The overview of achieved results is shown in Table 3 and precision-recall graphs of the selected attributes and their combinations are in Figure 3.

5 Conclusions and future work

We implemented 84 automatic collocation extraction methods and performed series of experiments on morphologically and syntactically annotated data. The methods were evaluated against a reference set of collocations manually extracted from the

	Recall			*Precision*		
	30	60	90	70	80	90
P. mutual information	85.5	78.4	62.5	78.0	56.0	16.3
Logistic regression-17	92.6	89.5	84.5	96.7	86.7	55.8
Absolute improvement	7.1	11.1	**22.0**	17.7	30.7	**39.2**
Relative improvement	8.3	14.2	**35.2**	23.9	54.8	**242.3**

Table 3: Precision (the 3 left columns) and recall (the 3 right columns) scores (in %) for the best individual method and linear combination of the 17 selected ones.

same source. The best method (*Pointwise mutual information*) achieved 68.3 % recall with 73.0 % precision (F-measure 70.6) on this data. We proposed to combine the association scores of each candidate bigram and employed *Logistic linear regression* to find a linear combination of the association scores of all the basic methods. Thus we constructed a collocation extraction method which achieved 80.8 % recall with 84.8 % precision (F-measure 82.8). Furthermore, we applied an attribute selection technique in order to lower the high dimensionality of the classification problem and reduced the number of regressors from 87 to 17 with comparable performance. This result can be viewed as a kind of evaluation of basic collocation extraction techniques. We can obtain the smallest subset that still gives the best result. The other measures therefore become uninteresting and need not be further processed and evaluated.

The reseach presented in this paper is in progress. The list of collocation extraction methods and association measures is far from complete. Our long term goal is to collect, implement, and evaluate all available methods suitable for this task, and release the toolkit for public use.

In the future, we will focus especially on improving quality of the training and testing data, employing other classification and attribute-selection techniques, and performing experiments on English data. A necessary part of the work will be a rigorous theoretical study of all applied methods and appropriateness of their usage. Finally, we will attempt to demonstrate contribution of collocations in selected application areas, such as machine translation or information retrieval.

Acknowledgments

This research has been supported by the Ministry of Education of the Czech Republic, project MSM 0021620838. I would also like to thank my advisor, Dr. Jan Hajič, for his continued support.

References

Y. Choueka. 1988. Looking for needles in a haystack or locating interesting collocational expressions in large textual databases. In *Proceedings of the RIAO*, pages 43–38.

I. Dagan, L. Lee, and F. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 34.

T. E. Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

S. Evert and B. Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *Proceedings 39th Annual Meeting of the Association for Computational Linguistics*, pages 188–195.

S. Evert. 2004. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Ph.D. thesis, University of Stuttgart.

J. Hajič, E. Hajičová, P. Pajas, J. Panevová, P. Sgall, and B. Vidová-Hladká. 2001. Prague dependency treebank 1.0. Published by LDC, University of Pennsylvania.

K. Kita, Y. Kato, T. Omoto, and Y. Yano. 1994. A comparative study of automatic extraction of collocations from corpora: Mutual information vs. cost criteria. *Journal of Natural Language Processing*, 1(1):21–33.

B. Krenn. 2000. Collocation Mining: Exploiting Corpora for Collocation Idenfication and Representation. In *Proceedings of KONVENS 2000*.

L. Lee. 2001. On the effectiveness of the skew divergence for statistical language analysis. *Artificial Inteligence and Statistics*, pages 65–72.

C. D. Manning and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.

D. Pearce. 2002. A comparative evaluation of collocation extraction techniques. In *Third International Conference on language Resources and Evaluation*, Las Palmas, Spain.

T. Pedersen. 1996. Fishing for exactness. In *Proceedings of the South Central SAS User's Group Conference*, pages 188–200, Austin, TX.

S. Shimohata, T. Sugio, and J. Nagata. 1997. Retrieving collocations by co-occurrences and word order constraints. In *Proc. of the 35th Annual Meeting of the ACL and 8th Conference of the EACL*, pages 476–81, Madrid. Spain.

P. Tan, V. Kumar, and J. Srivastava. 2002. Selecting the right interestingness measure for association patterns. In *Proceedings of the Eight A CM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

A. Thanopoulos, N. Fakotakis, and G. Kokkinakis. 2002. Comparative evaluation of collocation extraction metrics. In *3rd International Conference on Language Resources and Evaluation*, volume 2, pages 620–625, Las Palmas, Spain.

F. Čermák and J. Holub. 1982. *Syntagmatika a paradigmatika česk eho slova: Valence a kolokabilita*. Státní pedagogické nakladatelství, Praha.

I. H. Witten and E. Frank. 2000. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco.

C. Zhai. 1997. Exploiting context to identify lexical atoms – A statistical view of linguistic context. In *International and Interdisciplinary Conference on Modelling and Using Context (CONTEXT-97)*.

Jointly Labeling Multiple Sequences: A Factorial HMM Approach

Kevin Duh
Department of Electrical Engineering
University of Washington, USA
duh@ee.washington.edu

Abstract

We present new statistical models for jointly labeling multiple sequences and apply them to the combined task of part-of-speech tagging and noun phrase chunking. The model is based on the Factorial Hidden Markov Model (FHMM) with distributed hidden states representing part-of-speech and noun phrase sequences. We demonstrate that this joint labeling approach, by enabling information sharing between tagging/chunking subtasks, outperforms the traditional method of tagging and chunking in succession. Further, we extend this into a novel model, Switching FHMM, to allow for explicit modeling of cross-sequence dependencies based on linguistic knowledge. We report tagging/chunking accuracies for varying dataset sizes and show that our approach is relatively robust to data sparsity.

1 Introduction

Traditionally, various sequence labeling problems in natural language processing are solved by the cascading of well-defined subtasks, each extracting specific knowledge. For instance, the problem of information extraction from sentences may be broken into several stages: First, part-of-speech (POS) tagging is performed on the sequence of word tokens. This result is then utilized in noun-phrase and verb-phrase chunking. Finally, a higher-level analyzer extracts relevant information based on knowledge gleaned in previous subtasks.

The decomposition of problems into well-defined subtasks is useful but sometimes leads to unnecessary errors. The problem is that errors in earlier subtasks will propagate to downstream subtasks, ultimately deteriorating overall performance. Therefore, a method that allows the *joint labeling* of subtasks is desired. Two major advantages arise from simultaneous labeling: First, there is more robustness against error propagation. This is especially relevant if we use probabilities in our models. Cascading subtasks inherently "throws away" the probability at each stage; joint labeling preserves the uncertainty. Second, information between simultaneous subtasks can be shared to further improve accuracy. For instance, it is possible that knowing a certain noun phrase chunk may help the model infer POS tags more accurately, and vice versa.

In this paper, we propose a solution to the joint labeling problem by representing multiple sequences in a single Factorial Hidden Markov Model (FHMM) (Ghahramani and Jordan, 1997). The FHMM generalizes hidden Markov models (HMM) by allowing separate hidden state sequences. In our case, these hidden state sequences represent the POS tags and phrase chunk labels. The links between the two hidden sequences model dependencies between tags and chunks. Together the hidden sequences generate an observed word sequence, and the task of the tagger/chunker is to invert this process and infer the original tags and chunks.

Previous work on joint tagging/chunking has shown promising results. For example, Xun et

Proceedings of the ACL Student Research Workshop, pages 19–24,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

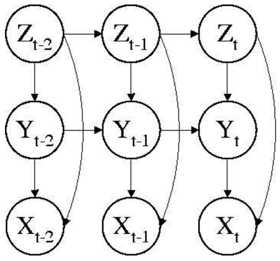


Figure 1: Baseline FHMM. The two hidden sequences $y_{1:t}$ and $z_{1:t}$ can represent tags and chunks, respectively. Together they generate $x_{1:t}$, the observed word sequence.

al. (2000) uses a POS tagger to output an N-best list of tags, then a Viterbi search to find the chunk sequence that maximizes the joint tag/chunk probability. Florian and Ngai (2001) extends transformation-based learning tagger to a joint tagger/chunker by modifying the objective function such that a transformation rule is evaluated on the classification of all simultaneous subtasks. Our work is most similar in spirit to Dynamic Conditional Random Fields (DCRF) (Sutton et al., 2004), which also models tagging and chunking in a factorial framework. Some main differences between our model and DCRF may be described as 1) directed graphical model vs. undirected graphical model, and 2) generative model vs. conditional model. The main advantage of FHMM over DCRF is that FHMM requires considerably less computation and exact inference is easily achievable for FHMM and its variants.

The paper is structured as follows: Section 2 describes in detail the FHMM. Section 3 presents a new model, the Switching FHMM, which represents cross-sequence dependencies more effectively than FHMMs. Section 4 discusses the task and data and Section 5 presents various experimental results Section 6 discusses future work and concludes.

2 Factorial HMM

2.1 Basic Factorial HMM

A Factorial Hidden Markov Model (FHMM) is a hidden Markov model with a distributed state representation. Let $x_{1:T}$ be a length T sequence of observed random variables (e.g. words) and $y_{1:T}$ and $z_{1:T}$ be the corresponding sequences of hidden state variables (e.g. tags, chunks). Then we define the FHMM as the probabilistic model:

$$\begin{aligned} & p(x_{1:T}, y_{1:T}, z_{1:T}) \\ = \;& \pi_0 \prod_{t=2}^{T} p(x_t|y_t, z_t) p(y_t|y_{t-1}, z_t) p(z_t|z_{t-1}) \end{aligned} \tag{1}$$

where $\pi_0 = p(x_0|y_0, z_0)p(y_0|z_0)p(z_0)$. Viewed as a generative process, we can say that the **chunk model** $p(z_t|z_{t-1})$ generates chunks depending on the previous chunk label, the **tag model** $p(y_t|y_{t-1}, z_t)$ generates tags based on the previous tag and current chunk, and the **word model** $p(x_t|y_t, z_t)$ generates words using the tag and chunk at the same time-step.

This equation corresponds to the graphical model of Figure 1. Although the original FHMM developed by Ghahramani (1997) does not explicitly model the dependencies between the two hidden state sequences, here we add the edges between the y and z nodes to reflect the interaction between tag and chunk sequences. Note that the FHMM can be collapsed into a hidden Markov model where the hidden state is the cross-product of the distributed states y and z. Despite this equivalence, the FHMM is advantageous because it requires the estimation of substantiatially fewer parameters.

FHMM parameters can be calculated via maximum likelihood (ML) estimation if the values of the hidden states are available in the training data. Otherwise, parameters must be learned using approximate inference algorithms (e.g. Gibbs sampling, variational inference), since exact Expectation-Maximization (EM) algorithm is computationally intractable (Ghahramani and Jordan, 1997). Given a test sentence, inference of the corresponding tag/chunk sequence is found by the Viterbi algorithm, which finds the tag/chunk sequence that maximizes the joint probability, i.e.

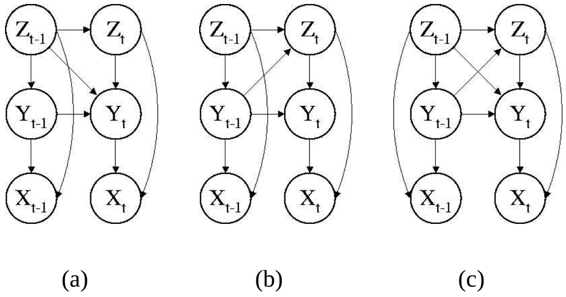
$$\arg \max_{y_{1:T}, z_{1:T}} p(x_{1:T}, y_{1:T}, z_{1:T}) \tag{2}$$

2.2 Adding Cross-Sequence Dependencies

Many other structures exist in the FHMM framework. Statistical modeling often involves the iterative process of finding the best set of dependencies that characterizes the data effectively. As shown in Figures 2(a), 2(b), and 2(c), dependencies can be added between the y_t and z_{t-1}, between z_t and y_{t-1}, or both. The model in Fig. 2(a) corresponds to changing the tag model in Eq. 1 to $p(y_t|y_{t-1}, z_t, z_{t-1})$; Fig. 2(b) corresponds to changing the chunk model to $p(z_t|z_{t-1}, y_{t-1})$; Fig. 2(c), corresponds to changing both tag and chunk models, leading to the probability model:

$$\prod_{t=1}^{T} p(x_t|y_t, z_t)p(y_t|y_{t-1}, z_t, z_{t-1})p(z_t|z_{t-1}, y_{t-1}) \quad (3)$$

We name the models in Figs. 2(a) and 2(b) as FHMM-T and FHMM-C due to the added dependencies to the **t**ag and **c**hunk models, respectively. The model of Fig. 2(c) and Eq. 3 will be referred to as FHMM-CT. Intuitively, the added dependencies will improve the predictive power across chunk and tag sequences, provided that enough training data are available for robust parameter estimation.



(a) (b) (c)

Figure 2: FHMMs with additional cross-sequence dependencies. The models will be referred to as (a) FHMM-T, (b) FHMM-C, and (c) FHMM-CT.

3 Switching Factorial HMM

A reasonable question to ask is, “How exactly does the chunk sequence interact with the tag sequence?” The approach of adding dependencies in Section 2.2 acknowledges the existence of cross-sequence interactions but does not explicitly specify the type of interaction. It relies on statistical learning to find the salient dependencies, but such an approach is feasable only when sufficient data are available for parameter estimation.

To answer the question, we consider how the chunk sequence affects the generative process for tags: First, we can expect that the unigram distribution of tags changes depending on whether the chunk is a noun phrase or verb phrase. (In a noun phrase, nouns and adjective tags are more common; in a verb phrase, verbs and adverb tags are more frequent.) Similarly, a bigram distribution $p(y_t|y_{t-1})$ describing tag transition probabilities differs depending on the bigram’s location in the chunk sequence, such as whether it is within a noun phrase, verb phrase, or at a phrase boundary. In other words, the chunk sequence interacts with tags by switching the particular generative process for tags. We model this interaction explicitly using a Switching FHMM:

$$\begin{aligned} & p(x_{1:T}, y_{1:T}, z_{1:T}) \quad (4) \\ = & \prod_{t=1}^{T} p(x_t|y_t, z_t)p_\alpha(y_t|y_{t-1})p_\beta(z_t|z_{t-1}) \end{aligned}$$

In this new model, the chunk and tag are now generated by bigram distributions parameterized by α and β. For different values of α (or β), we have different distributions for $p(y_t|y_{t-1})$ (or $p(z_t|z_{t-1})$). The crucial aspect of the model lies in a function $\alpha = f(z_{1:t})$, which summarizes information in $z_{1:t}$ that is relevant for the generation of y, and a function $\beta = g(y_{1:t})$, which captures information in $y_{1:t}$ that is relevant to the generation of z.

In general, the functions $f(\cdot)$ and $g(\cdot)$ partition the space of all tag or chunk sequences into several equivalence classes, such that all instances of an equivalence class give rise to the same generative model for the cross sequence. For instance, all consecutive chunk labels that indicate a noun phrase can be mapped to one equivalence class, while labels that indicate verb phrase can be mapped to another. The mapping can be specified manually or learned automatically. Section 5 discusses a linguistically-motivated mapping that is used for the experiments.

Once the mappings are defined, the parameters $p_\alpha(y_t|y_{t-1})$ and $p_\beta(z_t|z_{t-1})$ are obtained via maximum likelihood estimation in a fashion similar to that of the FHMM. The only exception is that now the training data are partitioned according to the mappings, and each α- and β- specific generative model is estimated separately. Inference of the tags and chunks for a test sentence proceeds similarly to FHMM inference. We call this model a Switching FHMM since the distribution of a hidden sequence ”switches” dynamically depending on the values of the other hidden sequence.

An idea related to the Switching FHMM is the Bayesian Multinet (Geiger and Heckerman, 1996;

Bilmes, 2000), which allows the dynamic switching of conditional variables. It can be used to implement switching from a higher-order model to a lower-order model, a form of backoff smoothing for dealing with data sparsity. The Switching FHMM differs in that it switches among models of the same order, but these models represent different generative processes. The result is that the model no longer requires a time-homogenous assumption for state transitions; rather, the transition probabilities change dynamically depending on the influence across sequences.

4 POS Tagging and NP Chunking

4.1 The Tasks

POS tagging is the task of assigning words the correct part-of-speech, and is often the first stage of various natural language processing tasks. As a result, POS tagging has been one of the most active areas of research, and many statistical and rule-based approach have been tried. The most notable of these include the trigram HMM tagger (Brants, 2000), maximum entropy tagger (Ratnaparkhi, 1996), transformation-based tagger (Brill, 1995), and cyclic dependency networks (Toutanova et al., 2003).

Accuracy numbers for POS tagging are often reported in the range of 95% to 97%. Although this may seem high, note that a tagger with 97% accuracy has only a 63% chance of getting all tags in a 15-word sentence correct, whereas a 98% accurate tagger has 74% (Manning and Schütze, 1999). Therefore, small improvements can be significant, especially if downstream processing requires correctly-tagged sentences. One of the most difficult problems with POS tagging is the handling of out-of-vocabulary words.

Noun-phrase (NP) chunking is the task of finding the non-recursive (base) noun-phrases of sentences. This segmentation task can be achieved by assigning words in a sentence to one of three tokens: B for "Begin-NP", I for "Inside-NP", or O for "Outside-NP" (Ramshaw and Marcus, 1995). The "Begin-NP" token is used in the case when an NP chunk is immediately followed by another NP chunk. The state-of-the-art chunkers report F1 scores of 93%-94% and accuracies of 87%-97%. See, for example, NP chunkers utilizing conditional random fields (Sha and Pereira, 2003) and support vector machines (Kudo and Matsumoto, 2001).

4.2 Data

The data comes from the CoNLL 2000 shared task (Sang and Buchholz, 2000), which consists of sentences from the Penn Treebank Wall Street Journal corpus (Marcus et al., 1993). The training set contains a total of 8936 sentences with 19k unique vocabulary. The test set contains 2012 sentences and 8k vocabulary. The out-of-vocabulary rate is 7%.

There are 45 different POS tags and 3 different NP labels in the original data. An example sentence with POS and NP tags is shown in Table 1.

The	move	could	pose	a	challenge
DT	NN	MD	VB	DT	NN
I	I	O	O	I	I

Table 1: Example sentence with POS tags (2nd row) and NP labels (3rd row). For NP, I = Inside-NP, O=Outside-NP.

5 Experiments

We report two sets of experiments. Experiment 1 compares several FHMMs with cascaded HMMs and demonstrates the benefit of joint labeling. Experiment 2 evaluates the Switching FHMM for various training dataset sizes and shows its robustness against data sparsity. All models are implemented using the Graphical Models Toolkit (GMTK) (Bilmes and Zweig, 2002).

5.1 Exp1: FHMM vs Cascaded HMMs

We compare the four FHMMs of Section 2 to the traditional approach of cascading HMMs in succession, and compare their POS and NP accuracies in Table 2. In this table, the first row "Oracle HMM" is an oracle experiment which shows what NP accuracies can be achieved if perfectly correct POS tags are available in a cascaded approach. The second row "Cascaded HMM" represents the traditional approach of doing POS tagging and NP chunking in succession; i.e. an NP chunker is applied to the output of a POS tagger that is 94.17% accurate. The next four rows show the results of joint labeling using various FHMMs. The final row "DCRF" are comparable results from Dynamic Conditional Random Fields (Sutton et al., 2004).

There are several observations: First, it is important to note that FHMM outperforms the cascaded HMM in terms of NP accuracy for all but one model. For instance, FHMM-CT achieves an NP accuracy of 95.93%, significantly higher than both the cascaded HMM (93.90%) and the oracle HMM (94.67%). This confirms our hypothesis that joint labeling helps prevent POS errors from propagating to NP chunking. Second, the fact that several FHMM models achieve NP accuracies higher than the *oracle* HMM implies that information sharing between POS and NP sequences gives even more benefit than having only perfectly correct POS tags. Thirdly, the fact that the most complex model (FHMM-CT) performs best suggests that it is important to avoid data sparsity problems, as it requires more parameters to be estimated in training.

Finally, it should be noted that although the DCRF outperforms the FHMM in this experiment, the DCRF uses significantly more word features (e.g. capitalization, existence in a list of proper nouns, etc.) and a larger context (previous and next 3 tags), whereas the FHMM considers the word as its sole feature, and the previous tag as its only context. Further work is required to see whether the addition of these features in the FHMM's generative framework will achieve accuracies close to that of DCRF. The take-home message is that, in light of the computational advantages of generative models, the FHMM should not be dismissed as a potential solution for joint labeling. In fact, recent results in the discriminative training of FHMMs (Bach and Jordan, 2005) has shown promising results in speech processing and it is likely that such advanced techniques, among others, may improve the FHMM's performance to state-of-the-art results.

5.2 Exp2: Switching FHMM and Data Sparsity

We now compare the Switching FHMM to the best model of Experiment 1 (FHMM-CT) for varying amounts of training data. The Switching FHMM uses the following α and β mapping. The mapping $\alpha = f(z_{1:t})$ partitions the space of chunk history $z_{1:t}$ into five equivalence classes based on the two most recent chunk labels:

Model	POS	NP
Oracle HMM	–	94.67
Cascaded HMM	94.17	93.90
Baseline FHMM	93.82	93.56
FHMM-T	93.73	94.07
FHMM-C	**94.16**	95.76
FHMM-CT	94.15	**95.93**
DCRF	98.92	97.36

Table 2: POS and NP Accuracy for Cascaded HMM and FHMM Models.

Class1. $\{z_{1:t} : z_{t-1} = I, z_t = I\}$
Class2. $\{z_{1:t} : z_{t-1} = O, z_t = O\}$
Class3. $\{z_{1:t} : z_{t-1} = \{I, B\}, z_t = O\}$
Class4. $\{z_{1:t} : z_{t-1} = O, z_t = \{I, B\}\}$
Class5. $\{z_{1:t} : (z_{t-1}, z_t) = \{(I, B), (B, I)\}\}$

Class1 and Class2 are cases where the tag is located strictly inside or outside an NP chunk. Class3 and Class4 are situations where the tag is leaving or entering an NP, and Class5 is when the tag transits between consecutive NP chunks. Class-specific tag bigrams $p_\alpha(y_t|y_{t-1})$ are trained by dividing the training data according to the mapping. On the other hand, the mapping $\beta = g(y_{1:t})$ is not used to ensure a single point of comparison with FHMM-CT; we use FHMM-CT's chunk model $p_{(}z_t|z_{t-1}, y_{t-1})$ in place of $p_\beta(z_t|z_{t-1})$.

The POS and NP accuracies are plotted in Figures 3 and 4. We report accuracies based on the average of five different random subsets of the training data for datasets of sizes 1000, 3000, 5000, and 7000 sentences. Note that for the Switching FHMM, POS and NP accuracy remains relatively constant despite the reduction in data size. This suggests that a more explicit model for cross sequence interaction is essential especially in the case of insufficient training data. Also, for the very small datasize of 1000, the accuracies for Cascaded HMM are 84% for POS and 70% for NP, suggesting that the general FHMM framework is still beneficial.

6 Conclusion and Future Work

We have demonstrated that joint labeling with an FHMM can outperform the traditional approach of cascading tagging and chunking in NLP. The new Switching FHMM generalizes the FHMM by allow-

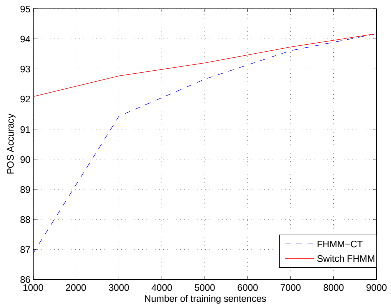


Figure 3: POS Accuracy for varying data sizes

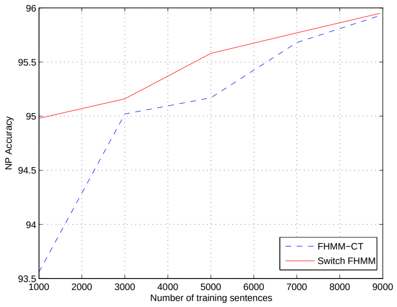


Figure 4: NP Accuracy for varying data sizes

ing dynamically changing generative models and is a promising approach for modeling the type of interactions between hidden state sequences.

Three directions for future research are planned: First, we will augment the FHMM such that its accuracies are competitive with state-of-the-art taggers and chunkers. This includes adding word features to improve accuracy on OOV words, augmenting the context from bigram to trigram, and applying advanced smoothing techniques. Second, we plan to examine the Switching FHMM further, especially in terms of automatic construction of the α and β function. A promising approach is to learn the mappings using decision trees or random forests, which has recently achieved good results in a similar problem in language modeling (Xu and Jelinek, 2004). Finally, we plan to integrate the tagger/chunker in an end-to-end system, such as a Factored Language Model (Bilmes and Kirchhoff, 2003), to measure the overall merit of joint labeling.

Acknowledgments

The author would like to thank Katrin Kirchhoff, Jeff Bilmes, and Gang Ji for insightful discussions, Chris Bartels for support on GMTK, and the two anonymous reviewers for their constructive comments. Also, the author gratefully acknowledges support from NSF and CIA under NSF Grant No. IIS-0326276.

References

Francis Bach and Michael Jordan. 2005. Discriminative training of hidden Markov models for multiple pitch tracking. In *Proc. Intl. Conf. Acoustics, Speech, Signal Processing*.

J. Bilmes and K. Kirchhoff. 2003. Factored language models and generalized parallel backoff. In *Proc. of HLT/NACCL*.

J. Bilmes and G. Zweig. 2002. The Graphical Models Toolkit: An open source software system for speech and time-series processing. In *Intl. Conf. on Acoustics, Speech, Signal Proc.*

Jeff Bilmes. 2000. Dynamic bayesian multi-networks. In *The 16th Conference on Uncertainty in Artificial Intelligence*.

Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the Applied NLP*.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

Radu Florian and Grace Ngai. 2001. Multidimensional transformation-based learning. In *Proc. CoNLL*.

D. Geiger and D. Heckerman. 1996. Knowledge representation and inference in similarity netwrosk and Bayesian multinets. *Artificial Intelligence*, 82:45–74.

Z. Ghahramani and M. I. Jordan. 1997. Factorial hidden Markov models. *Machine Learning*, 29:245–275.

T. Kudo and Y. Matsumoto. 2001. Chunking with support vector machines. In *Proceedings of NAACL-2001*.

C. D. Manning and H. Schütze, 1999. *Foundations of Statistical Natural Language Processing*, chapter 10. MIT Press.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

L. A. Ramshaw and M. P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora (ACL-95)*.

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP-1996*.

E. F. Tjong Kim Sang and S. Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. CoNLL*.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL*.

C. Sutton, K. Rohanimanesh, and A. McCallum. 2004. Dynamic conditional random fields. In *Intl. Conf. Machine Learning (ICML 2004)*.

K. Toutanova, D. Klein, C. Manning, and Y. Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of HLT-NAACL*.

Peng Xu and Frederick Jelinek. 2004. Random forests in language modeling. In *Proc. EMNLP*.

E. Xun, C. Huang, and M. Zhou. 2000. A unified statistical model for the identification of English BaseNP. In *Proc. ACL*.

Exploiting Named Entity Taggers in a Second Language

Thamar Solorio
Computer Science Department
National Institute of Astrophysics, Optics and Electronics
Luis Enrique Erro #1, Tonantzintla, Puebla
72840, Mexico

Abstract

In this work we present a method for Named Entity Recognition (NER). Our method does not rely on complex linguistic resources, and apart from a hand coded system, we do not use any language-dependent tools. The only information we use is automatically extracted from the documents, without human intervention. Moreover, the method performs well even without the use of the hand coded system. The experimental results are very encouraging. Our approach even outperformed the hand coded system on NER in Spanish, and it achieved high accuracies in Portuguese.

1 Introduction

Given the usefulness of Named Entities (NEs) in many natural language processing tasks, there has been a lot of work aimed at developing accurate named entity extractors (Borthwick, 1999; Velardi et al., 2001; Arévalo et al., 2002; Zhou and Su, 2002; Florian, 2002; Zhang and Johnson, 2003). Most approaches however, have very low portability, they are designed to perform well over a particular collection or type of document, and their accuracies will drop considerably when used in different domains. The reason for this is that many NE extractor systems rely heavily on complex linguistic resources, which are typically hand coded, for example regular expressions, grammars, gazetteers and the like. Adapting a system of this nature to a different collection or language requires a lot of human effort, involving tasks such as rewriting the grammars, acquiring new dictionaries, searching trigger words, and so on. Even if one has the human resources and the time needed for the adaptation process, there are languages that lack the linguistic resources needed, for instance, dictionaries are available in electronic form for only a handful of languages. We believe that, by using machine learning techniques, we can adapt an existing hand coded system to different domains and languages with little human effort.

Our goal is to present a method that will facilitate the task of increasing the coverage of named entity extractor systems. In this setting, we assume that we have available an NE extractor system for Spanish, and we want to adapt it so that it can perform NER accurately in documents from a different language, namely Portuguese. It is important to emphasize here that we try to avoid the use of complex and costly linguistic tools or techniques, besides the existing NER system, given the language restrictions they pose. Although, we do need a corpus of the target language. However, we consider the task of gathering a corpus much easier and faster than that of developing linguistic tools such as parsers, part-of-speech taggers, grammars and the like.

In the next section we present some recent work related to NER. Section 3 describes the data sets used in our experiments. Section 4 introduces our approach to NER, and we conclude in Section 5 giving a brief discussion of our findings and proposing research lines for future work.

Proceedings of the ACL Student Research Workshop, pages 25–30,
Ann Arbor, Michigan, June 2005.

2 Related Work

There has been a lot of work on NER, and there is a remarkable trend towards the use of machine learning algorithms. Hidden Markov Models (HMM) are a common choice in this setting. For instance, Zhou and Su trained HMM with a set of attributes combining internal features such as gazetteer information, and external features such as the context of other NEs already recognized (Zhou and Su, 2002). (Bikel et al., 1997) and (Bikel et al., 1999) are other examples of the use of HMMs.

Previous methods for increasing the coverage of hand coded systems include that of Borthwick, he used a maximum entropy approach where he combined the output of three hand coded systems with dictionaries and other orthographic information (Borthwick, 1999). He also adapted his system to perform NER in Japanese achieving impressive results.

Spanish resources for NER have been used previously to perform NER on a different language. Carreras et al. presented results of a NER system for Catalan using Spanish resources (Carreras et al., 2003a). They explored several methods for building NER for Catalan. Their best results are achieved using cross-linguistic features. In this method the NER system is trained on mixed corpora and performs reasonably well on both languages. Our work follows Carreras et al. approach, but differs in that we apply directly the NER system for Spanish to Portuguese and train a classifier using the output and the real classes.

In (Petasis et al., 2000) a new method for automating the task of extending a proper noun dictionary is presented. The method combines two learning approaches: an inductive decision-tree classifier and unsupervised probabilistic learning of syntactic and semantic context. The attributes selected for the experiments include POS tags as well as morphological information whenever available.

One work focused on NE recognition for Spanish is based on discriminating among different kinds of named entities: core NEs, which contain a trigger word as nucleus, syntactically simple weak NEs, formed by single noun phrases, and syntactically complex named entities, comprised of complex noun phrases. Arévalo and colleagues focused on the first two kinds of NEs (Arévalo et al., 2002). The method is a sequence of processes that uses simple attributes combined with external information provided by gazetteers and lists of trigger words. A context free grammar, manually coded, is used for recognizing syntactic patterns.

3 Data sets

In this paper we report results of experimenting with two data sets. The corpus in Spanish is that used in the CoNLL 2002 competitions for the NE extraction task. This corpus is divided into three sets: a training set consisting of 20,308 NEs and two different sets for testing, *testa* which has 4,634 NEs and *testb* with 3,948 NEs, the former was designated to tune the parameters of the classifiers (development set), while *testb* was designated to compare the results of the competitors. We performed experiments with *testa* only.

For evaluating NER on Portuguese we used the corpus provided by “HAREM: Evaluation contest on named entity recognition for Portuguese”. This corpus contains newspaper articles and consists of 8,551 words with 648 NEs.

4 Two-step Named Entity Recognition

Our approach to NER consists in dividing the problem into two subproblems that are addressed sequentially. We first solve the problem of determining boundaries of named entities, we called this process Named Entity Delimitation (NED). Once we have determined which words belong to named entities, we then get to the task of classifying the named entities into categories, this process is what we called Named Entity Classification (NEC). We explain the two procedures in the following subsections.

4.1 Named Entity Delimitation

We used the BIO scheme for delimiting named entities. In this approach each word in the text is labeled with one out of three possible classes: The B tag is assigned to words believed to be the beginning of a NE, the I tag is for words that belong to an entity but that are not at the beginning, and the O tag is for all words that do not satisfy any of the previous two conditions.

Table 1: An example of the attributes used in the learning setting for NER in Spanish. The fragment presented in the table, *"El Ejército Mexicano puso en marcha el Plan DN-III"*, translates as "The Mexican Army launched the DN-III plan"

Internal Features			External Features		
Word	Caps	Position	POS tag	BIO tag	Class
El	3	1	DA	O	O
Ejército	2	2	NC	B	B
Mexicano	2	3	NC	I	I
puso	2	4	VM	O	O
en	2	5	SP	O	O
marcha	2	6	NC	O	O
el	3	7	DA	O	O
Plan	2	8	NC	B	B
DN-III	3	9	NC	I	I

In our approach, NED is tackled as a learning task. The features used as attributes are automatically extracted from the documents and are used to train a machine learning algorithm. We used a modified version of C4.5 algorithm (Quinlan, 1993) implemented within the WEKA environment (Witten and Frank, 1999).

For each word we combined two types of features: internal and external; we consider as internal features the word itself, orthographic information and the position in the sentence. The external features are provided by the hand coded NER system for Spanish, these are the Part-of-Speech tag and the BIO tag. Then, the attributes for a given word w are extracted using a window of five words anchored in the word w, each word described by the internal and external features mentioned previously.

Within the orthographic information we consider 6 possible states of a word. A value of 1 in this attribute means that the letters in the word are all capitalized. A value of 2 means the opposite: all letters are lower case. The value 3 is for words that have the initial letter capitalized. 4 means the word has digits, 5 is for punctuation marks and 6 refers to marks representing the beginning and end of sentences.

The hand coded system used in this work was developed by the TALP research center (Carreras and Padró, 2002). They have developed a set of NLP analyzers for Spanish, English and Catalan that include practical tools such as POS taggers, semantic analyzers and NE extractors. This NER system is based on hand-coded grammars, lists of trigger words and gazetteer information.

In contrast to other methods we do not perform binary classifications, as (Carreras et al., 2003b), thus we do not build specialized classifiers for each of the tags. Our classifier learns to discriminate among the three classes and assigns labels to all the words, processing them sequentially. In Table 1 we present an example taken from the data used in the experiments where internal and external features are extracted for each word in a sentence.

4.1.1 Experimental Results

For all results reported here we show the overall average of several runs of 10-fold cross-validation. We used common measures from information retrieval: precision, recall and F_1 and we present results from individual classes as we believe it is important in a learning setting such as this, where nearly 90% of the instances belong to one class.

Table 2 presents comparative results using the Spanish corpus. We show four different sets of results, the first ones are from the hand coded system, they are labeled *NER system for Spanish*. Then we present results of training a classifier with only the internal features described above, these results are labeled *Internal features*. In a third experiment we trained the classifier using only the output of the NER system, these are under column *External features*. Finally, the results of our system are presented in column labeled *Our method*. We can see that even though the NER system performs very well by itself, by training the C4.5 algorithm on its outputs we improve performance in all the cases, with the exception of precision for class B. Given that the hand coded system was built for this collection, it is very encouraging to see our method outperforming this system. In Table 3 we show results of applying our method to the Portuguese corpus. In this case the improvements are much more impressive, particularly for class B, in all the cases the best results are obtained from our technique. This was expected as we are using a system developed for a different language. But we can see that our method yields very competitive results for Portuguese, and although by using only the internal features we can outperform the hand coded system, by combining the information using our method we can increase accuracies.

Table 2: Comparison of results for Spanish NE delimitation

	NER system for Spanish			Internal features			External features			Our method		
Class	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1
B	92.8	89.3	91.7	87.1	89.3	88.2	93.9	91.5	92.7	93.5	92.9	**93.2**
I	84.3	85.2	84.7	89.5	77.1	82.9	87.8	87.8	85.7	90.6	87.4	**89.0**
O	98.6	98.9	98.8	98.1	98.9	98.5	98.7	99	98.9	98.9	99.2	**99.1**
overall	**91.9**	**91.1**	**91.7**	**91.5**	**88.4**	**89.8**	**93.4**	**92.7**	**92.4**	**94.3**	**93.1**	**93.7**

Table 3: Experimental results for NE delimitation in Portuguese

	NER system for Spanish			Internal features			External features			Our method		
Class	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1
B	60.0	68.8	64.1	82.4	85.8	84.1	75.9	81.0	78.4	82.1	87.8	**84.9**
I	64.5	73.3	68.6	80.1	76.8	78.4	73.8	70.3	72.0	80.9	77.8	**79.3**
O	97.2	95.5	96.4	98.7	98.5	**98.6**	98.1	97.7	97.9	98.8	98.4	**98.6**
overall	**73.9**	**79.2**	**76.3**	**87.0**	**87.0**	**87.0**	**82.6**	**83.0**	**82.7**	**87.2**	**88.0**	**87.6**

From the results presented above, it is clear that the method can perform NED in Spanish and Portuguese with very high accuracy. Another insight suggested by these results is that in order to perform NED in Portuguese we do not need an existing NED system for Spanish, the internal features performed well by themselves, but if we have one available, we can use the information provided by it to build a more accurate NED method.

4.2 Named Entity Classification

As mentioned previously, we build our NE classifiers using the output of a hand coded system. Our assumption is that by using machine learning algorithms we can improve performance of NE extractors without a considerable effort, as opposed to that involved in extending or rewriting grammars and lists of trigger words and gazetteers. Another assumption underlying this approach is that of believing that the misclassifications of the hand coded system for Spanish will not affect the learner. We believe that by having available the correct NE classes in the training corpus, the learner will be capable of generalizing error patterns that will be used to assign the correct NE. If this assumption holds, learning from other's mistakes, the learner will end up outperforming the hand coded system.

In order to build a training set for the learner, each instance is described with the same attributes as for the NED task described in section 4.1, with the addition of a new attribute. Since NEC is a more difficult task, we consider useful adding as attribute the suffix of each word. Then, for each instance word we consider its suffix, with a maximum size of 5 characters.

Another important difference between this classification task and NED relies in the set of target values. For the Spanish corpus the possible class values are the same as those used in CoNLL-2002 competition task: *person*, *organization*, *location* and *miscellaneous*. However, for the Portuguese corpus we have 10 possible classes: *person*, *object*, *quantity*, *event*, *organization*, *artifact*, *location*, *date*, *abstraction* and *miscellaneous*. Thus the task of adapting the system for Spanish to perform NEC in Portuguese is much more complex than that of NED given that the Spanish system only discerns the four NE classes defined on the CoNLL-2002. Regardless of this, we believe that the learner will be capable of achieving good accuracies by using the other attributes in the learning task.

4.2.1 Experimental Results

Similarly to the NED case we trained C4.5 classifiers for the NEC task, results are presented in Tables 4 and 5. Again, we perform comparisons between the hand coded system and the use of different subsets of attributes. For the case of Spanish NEC, we can see in Table 4, that our method using internal and external features presents the best results. The improvements are impressive, specially for the NE class *Miscellaneous* where the hand coded system achieved an F measure below 1 while our system achieved an F measure of 56.7. In the case of NEC in Portuguese the results are very encouraging. The

Table 4: NEC performance on the Spanish development set

	NER system for Spanish			Internal features			External features			Our method		
Class	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1
Per	84.7	93.2	88.2	94.0	62.9	75.3	88.3	93.1	90.6	88.2	95.4	**91.7**
Org	78.7	88.7	82.9	61.7	90.0	73.2	77.7	91.9	84.2	83.4	89.0	**86.1**
Loc	78.7	76.2	76.9	78.4	65.1	71.2	80.3	80.3	80.3	82.0	82.5	**82.2**
Misc	24.9	.004	.008	75.5	42.0	54.0	52.9	23.4	33.5	71.6	46.9	**56.7**
overall	**66.7**	**64.5**	**62.0**	**77.4**	**65.0**	**68.4**	**74.8**	**72.1**	**72.1**	**81.3**	**78.4**	**79.1**

hand coded system performed poorly but by training a C4.5 algorithm results are improved considerably, even for the classes that the hand coded system was not capable of recognizing. As expected, the external features did not solve the NEC by themselves but contribute for improving the performance. This, and the results from using only internal features, suggest that we do not need complex linguistic resources in order to achieve good results. Additionally, we can see that for some cases the classifiers were not able of performing an accurate classification, as in the case of classes *object* and *miscellaneous*. This may be due to a poor representation of the classes in the training set, for instance the class *object* has only 4 instances. We believe that if we have more instances available the learners will improve these results.

5 Conclusions

Named entities have a wide usage in natural language processing tasks. For instance, it has been shown that indexing NEs within documents can help increase precision of information retrieval systems (Mihalcea and Moldovan, 2001). Other applications of NEs are in Question Answering (Mann, 2002; Pérez-Coutiño et al., 2004) and Machine Translation (Babych and Hartley, 2003). Thus it is important to have accurate NER systems, but these systems must be easy to port and robust, given the great variety of documents and languages for which it is desirable to have these tools available.

In this work we have presented a method for performing named entity recognition. The method uses a hand coded system and a set of lexical and orthographic features to train a machine learning algorithm. Apart from the hand coded system our method does not require any language dependent features, we do not make use of lists of trigger words, neither we use any gazetteer information. The only information used in this approach is automatically extracted from the documents, without human intervention. Yet, the results presented here are very encouraging. We were able to achieve good accuracies for NEC in Portuguese, where we needed to classify NEs into 10 possible classes, by exploiting a hand-coded system for Spanish targeted to only 4 classes. This achievement gives evidence of the flexibility of our method. Additionally we outperform the hand coded system on NER in Spanish. Thus, our method has shown to be robust and easy to port to other languages. The only requirement for using our method is a tokenizer for languages that do not separate words with white spaces, the rest can be used pretty straightforward.

We are interested in exploring the use of this method to perform NER in English, we would like to determine to what extent our system is capable of achieving competitive results without the use of language dependent resources, such as dictionaries and lists of words. Another research direction is the adaptation of this method to cross language NER. We are very interested in exploring if, by training a classifier with mixed language corpora, we can perform NER in more than one language simultaneously.

References

Montse Arévalo, Xavier Carreras, Lluís Màrquez, Toni Martí, Lluís Padró, and Maria José Simon. 2002. A proposal for wide-coverage Spanish named entity recognition. *Sociedad Española para el Procesamiento del Lenguaje Natural*, (28):63–80, May.

Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the EACL 2003 Workshop on MT and Other Language Technology Tools*, pages 1–8.

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high perfor-

Table 5: NEC performance on the Portuguese set

	NER system for Spanish			Internal features			External features			Our method		
Class	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1	**P**	**R**	F_1
Pessoa (Person)	34.8	72.5	46.6	49.1	92.0	**64.0**	46.9	64.6	54.4	45.5	91.1	60.7
Coisa (Object)	0	0	0	0	0	0	0	0	0	0	0	0
Valor (Quantity)	0	0	0	82.1	47.1	59.8	74.6	69.1	71.8	77.6	76.5	**77.0**
Acontecimento (Event)	0	0	0	33.3	21.4	26.1	14.3	7.1	9.5	50.0	21.4	**30.0**
Organização (Organization)	41.4	38.4	39.3	70.7	56.9	**63.1**	45.7	56.9	50.7	79.3	49.2	60.8
Obra (Artifact)	0	0	0	76.6	64.3	**69.9**	29.4	8.9	13.7	74.4	57.1	64.6
Local (Location)	52.5	16.5	24.8	72.6	32.6	**45.0**	43.6	38.5	40.9	67.4	32.1	43.5
Tempo (Date)	0	0	0	74.0	86.6	79.8	85.5	83.9	84.7	87.0	83.9	**85.5**
Abstracção (Abstraction)	0	0	0	82.1	41.8	**55.4**	22.2	3.6	6.3	79.3	41.8	54.8
Variado (Miscellaneous)	0	0	0	1	15.4	**26.7**	0	0	0	1	15.4	**26.7**
overall	**12.8**	**12.7**	**11.0**	**54.1**	**45.8**	**48.9**	**36.2**	**33.2**	**33.2**	**56.1**	**46.8**	**50.3**

mance learning name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 194–201.

Daniel M. Bikel, Richard Schwartz, and Ralph Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning, Special Issue on Natural Language Learning*, 34(1–3):211–231, February.

Andrew Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University, New York, September.

Xavier Carreras and Lluís Padró. 2002. A flexible distributed architecture for natural language analyzers. In *Proceedings of LREC'02*, Las Palmas de Gran Canaria, Spain.

Xavier Carreras, Lluís Màrquez, and Lluís Padró. 2003a. Named entity recognition for Catalan using Spanish resources. In *10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, Budapest, Hungary, April.

Xavier Carreras, Lluís Màrquez, and Lluís Padró. 2003b. A simple named entity extractor using adaboost. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 152–155. Edmonton, Canada.

Radu Florian. 2002. Named entity recognition as a house of cards: Classifier stacking. In *Proceedings of CoNLL-2002*, pages 175–178. Taipei, Taiwan.

Gideon S. Mann. 2002. Fine-grained proper noun ontologies for question answering. In *SemaNet'02: Building and Using Semantic Networks*, Taipei, Taiwan.

Rada Mihalcea and Dan Moldovan. 2001. Document indexing using named entities. *Studies in Informatics and Control*, 10(1), January.

Manuel Pérez-Coutiño, Thamar Solorio, Manuel Montes y Gómez, Aurelio López López, and Luis Villaseñor Pineda. 2004. Question answering for Spanish based on lexical and context annotation. In Christian Lemaître, Carlos Reyes, and Jesús A. González, editors, *Advances in Artificial Intelligence – IBERAMIA 2004*, Lecture Notes in Artificial Intelligence 3315, pages 325–333, Puebla, Mexico, November. Springer.

Georgios Petasis, Alessandro Cucchiarelli, Paola Velardi, Georgios Paliouras, Vangelis Karkaletsis, and Constantine D. Spyropoulos. 2000. Automatic adaptation of proper noun dictionaries through cooperation of machine learning and probabilistic methods. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 128–135. ACM Press.

J. R. Quinlan. 1993. C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufmann.

Thamar Solorio. 2005. *Improvement of Named Entity Tagging by Machine Learning*. Ph.D. thesis, Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Puebla, Mexico, (to appear).

Paola Velardi, Paolo Fabriani, and Michel Missikoff. 2001. Using text processing techniques to automatically enrich a domain ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems*, pages 270–284. ACM Press.

Ian H. Witten and Eibe Frank. 1999. *Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations*. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann.

Tong Zhang and David Johnson. 2003. A robust risk minimization based named entity recognition system. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 204–207. Edmonton, Canada.

Guodong Zhou and Jian Su. 2002. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of ACL'02*, pages 473–480.

Automatic Discovery of Intentions in Text and its Application to Question Answering

Marta Tatu
Human Language Technology Research Institute
Department of Computer Science
University of Texas at Dallas
Richardson, TX 75080, USA
marta@hlt.utdallas.edu

Abstract

Semantic relations between text concepts denote the core elements of lexical semantics. This paper presents a model for the automatic detection of INTENTION semantic relation. Our approach first identifies the syntactic patterns that encode intentions, then we select syntactic and semantic features for a SVM learning classifier. In conclusion, we discuss the application of INTENTION relations to Q&A.

1 Introduction

1.1 Problem description

Intentions comprise of semantic relationships that express a human's goal-oriented private states of mind, including intents, objectives, aims, and purposes. As a relation, it encodes information that might not be explicitly stated in text and its detection might require inferences and human judgment. The answer to the question *What was Putin trying to achieve by increasing military cooperation with North Korea?* is found in the sentence *Putin is attempting to restore Russia's influence in the East Asian region*. Extracting the exact answer *to restore Russia's influence in the East Asian region* becomes easier if this is recognized as Putin's intention which matches the question's expected answer.

In this paper, we describe a method that identifies *intentions* in domain independent texts. We employed two machine learning algorithms to create models that locate intentions in a given paragraph using a set of six syntactic and semantic features.

1.2 Motivation

The current state-of-the-art NLP systems cannot extract *intentions* from open text and, as we saw in the example, their detection benefits Question Answering. An intention is the answer to general questions like *What is the goal of X?*, *What does X plan to do?*, or *What does X aim for?* The INTENTION semantic relation is one of the most challenging relations because text fragments may convey unstated intentions. These are most pervasive in dialogues, communication specific to humans. For example, in the following conversation, the vendor infers the client's unstated intention of buying the cups.

Customer: *Where do you have the $1 cups?*
Salesman: *How many do you want?*

Intentions are closely related to other semantic relations such as beliefs, motives, desires, or plans. In the above example, the context tells us that this takes place in a superstore, well-known as a place where people buy things from. The clerk's answer emerges from our common beliefs and background knowledge as well as from his desire to help a customer. Intentions are the framework for plans. Many philosophers and artificial intelligence researchers studied the intentions as parts of coordinating plans (Bratman, 1987; Pollack, 1990) because people establish plans for future times.

In this paper, we regard *intentions* as expressions of a particular action that shall take place in the future, in which the speaker is some sort of agent (Anscombe, 1957). For example, the sentence *Mary is going to buy a TV set* shows Mary's intention. Anscombe (1957) considers intentions as a subclass of predictions, besides commands and

Proceedings of the ACL Student Research Workshop, pages 31–36,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

prophecies. *John is going to be sick* is usually a prophecy, *John, go for a walk!* is an order, and *John plans to take a walk* expresses an intention.

1.3 Previous work

Various methodologies have been proposed and used over the years for the task of extracting semantic relations from text. Purely probabilistic models, empirical methods, or hand-coded constraints were some of the approaches that do not use machine learning algorithms. Later on, methods that use decision tree, neural networks, memory-based learning, or support vector machines were introduced. Currently, there is also a increased interest in shallow semantic parsing of open texts and automatic labeling of semantic roles. Wiebe et al. (2004) focused on the detection of subjective language such as opinions, evaluations, or emotions in text. Using clues of subjectivity (low-frequency words, collocations), they identify opinion piece texts such as editorials, letters to the editor, or arts and leisure reviews.

There exists an immense literature in philosophy about the different types of intentions and their characteristics. Bratman (1987) tries to find the relationship between the two distinct phenomena of *doing something intentionally* and *intending to do something*. Numerous philosophical studies discuss how intentions relate to other psychological concepts, such as, beliefs, desires, hopes, or expectations (Audi, 1973; Bratman, 1981; Bratman, 1987). Intentions are consistent with the person's beliefs, and, unlike ordinary desires, require consistency (Bratman, 1987). They can generate reasons for or against future intentions (Bratman, 1981; Bratman, 1987). As plan elements, intentions require a certain stability. Their side effects need not be intended, even if they were taken into consideration in the first place[1] (Bratman, 1990).

2 Syntax and Semantics of Intention

2.1 Syntactic patterns

Because, in all the cases that we encountered, intentions were conveyed by phrases, we took a closer look at how intentions can be expressed in the written text. For our investigations, we chose the SemCor text collection (Miller et al., 1993), a subset of the Brown corpus manually tagged with WordNet senses (37,176 sentences in 352 newspaper articles). After manually classifying the first 2,700 sentences from SemCor into sentences that contain or not intentions, only 46 examples were identified. The syntactic patterns listed in Table 1 cover 95.65% of them. Because the first pattern comprises more than half of the studied examples, our algorithm focuses on detecting intentions encoded by VB_1 *to* VB_2. We note that this pattern is ambiguous and may convey other semantics. For instance, *Mary began to play with the dog*, *He told her to meet you* are encoded by our pattern, but do not express intentions.

Pattern	Example	Frequency
VB_1 *to* VB_2	plan to go for a walk	27 (58.69)
NN to VB	strivings to give up drink	6 (13.04)
VB PP VP	He resigned so that he can work for the school campaign	5 (10.87)
goal/purpose is to VB	his goal is to leave the country	4 (8.69)
ADJ to VB	eager to end a pitching slump	2 (4.34)

Table 1: INTENTION syntactic patterns

2.2 Semantics of intentions

From the semantic point of view, an intention may be very specific, it may contain a future time or a location (*John intends to meet Mary today*), but every intention must specify a future action. Hence, we propose the following representation for the INTENTION semantic relation: $\text{INT}(e_1, x_1, e_2)$ where e_1 is the event denoting the intention, x_1 denotes the person that has the intention and e_2 is the intended action or event. If the intention is more specific then we will identify instances of other semantic relations[2]. $John(x_1) \wedge \text{INT}(e_1, x_1, e_2) \wedge meet(e_2) \wedge Mary(x_2) \wedge today(x_3) \wedge \text{THEME}(e_2, x_2) \wedge \text{TIME}(e_2, x_3)$ represents a more specific intention.

The semantics of the INTENTION relation allows the derivation of inference rules which show that INTENTION dominates other semantic relations such as PURPOSE, ENTAIL, or ISA. For example, if a person x_1 intends to perform action e_2 and this action has a purpose e_3, then we can say that x_1 intends to do e_3[3]. Formally, we can express the above relations

[1] Due to space limitations, we couldn't include detailed examples. Please see the cited articles for examples.

[2] The list of semantic relations that can specialize an INT includes THEME, LOCATION, TEMPORAL, MANNER, INSTRUMENT, SOURCE, MEANS, and FREQUENCY. Their arguments are e_2, the intention verb, and a corresponding x_i.

[3] Similar statements can be made for the ENTAIL and ISA

with the following set of implications[4]:

$\text{INT}(e_1, x_1, e_2) \wedge \text{PURPOSE}(e_2, e_3) \Rightarrow \text{INT}(e_4, x_1, e_3)$

$\text{INT}(e_1, x_1, e_2) \wedge \text{ENTAIL}(e_2, e_3) \Rightarrow \text{INT}(e_4, x_1, e_3)$

$\text{INT}(e_1, x_1, e_2) \wedge \text{IS-A}(e_2, e_3) \Rightarrow \text{INT}(e_4, x_1, e_3)$

$\text{INT}(e_1, x_1, e_2) \wedge \text{PURPOSE}(e_3, e_2) \not\Rightarrow \text{INT}(e_4, x_1, e_3)$

$\text{INT}(e_1, x_1, e_2) \wedge \text{CAUSE}(e_2, e_3) \not\Rightarrow \text{INT}(e_4, x_1, e_3)$

The first three implications formalize the above inference rules. If *John intends to start his car to go to the park*, then John intends to go to the park. Similarly, if *John intends to buy a car*, then we can say that he intends *to pay for it*. The sentences *John intends to go to the park. He's starting his car right now* express John's intention to go to the park (e_2). The purpose of starting the car (e_3) is to go to the park. We cannot say that John intends to start his car. This is just an intentional action done to achieve his objective. The fifth rule tries to eliminate the effects (e_3) of an intention (e_2) from being considered as intentions or objectives. If *John intends to swim in the pool* (e_2) even if he knows that he is going to *catch a cold* (e_3) because the water is too cold, we cannot say that *John intends to catch a cold.*[5] The traditional relational properties (*reflexivity*, *symmetry*, or *transitivity*) do not hold for the INTENTION semantic relation.

3 Learning Model

3.1 Experimental data

We applied the most frequent syntactic pattern that expresses intentions in text (VB_1 *to* VB_2) on the first 10,000 sentences of the SemCor2.0 collection and we extracted 1,873 sentences. These sentences contain 115 intentions (manually identified by a graduate student, not the author). The data consisting of these positives and 258 arbitrarily selected negative examples, was randomly divided into a training set that contains 80% of the examples and a test set with the remaining 20% instances. The statistics are shown in Table 2.

	Intentions	Non-Intentions	Total
Training	92	208	300
Testing	23	50	73

Table 2: Experiments Data Division

3.2 Features for intention

After analyzing our training data, we pinpointed a set of features to help us identify the *intentions* encoded by the pattern VB_1 *to* VB_2. The WordNet senses needed to extract the semantic features were taken from SemCor. We will use *Mary intends to revise the paper* to show each feature's value.

The semantic class of the the VB_1 verb's agent or **specializations of it.** Intentions and objectives are specific to humans. Thus, the semantic class of the VB_1 agent bears a high importance. We used an in-house semantic parser to retrieve the AGENT of the VB_1 verb. The feature's value is its WordNet semantic class. *Mary* names a person. Thus, the semantic class that we are seeking is *entity#1*.

We chose this semantic generalization because nouns and verbs belong to open part-of-speech classes. There can be an enormous number of possibilities and any models built using them as feature values will not be able to generalize beyond the training examples. Therefore, we introduce a bias in our learning framework based on the assumption: noun and verb concepts will semantically behave as the concepts that subsume them in the WordNet structures. But, by generalizing concepts, we lose some of their semantic properties. Hence, we specialize the semantic class s of a concept w by replacing it with its immediate hyponym (h) that subsumes w. We can further increase the semantic level by specializing h. We note that the number of values is still finite even though we specialized the general concepts. As the specialization level increases, there will be words w that cannot be further specialized (*entity#1* cannot be specialized even once). In such cases, we add w to the set of feature values.

The semantic class of the VB_1 verb or **its specializations.** The intention phrase is subordinated to a verb (VB_1). The semantic class of this verb is the system's second feature. In our example, VB_1 (*intend#1*) semantic class is *wish#3*.

The semantic class of the VB_2 verb's agent, if this agent differs from the VB_1 verb's agent; otherwise, a common value (*equal*) is given. We identify the AGENT of the VB_2 verb. The specializations of its semantic class will be used if the top noun proves to be too general. In the sample sentence, the agent of *revise* is *Mary*. We can have a different agent for

semantic relations.

[4] e_1 and e_4 represent different intentions of the same person.

[5] A more detailed example can be found in (Bratman, 1990).

Semantic class of the VB_1's agent	Semantic class of the VB_1 verb (%)								
	no specialization			1^{st} level of specialization			2^{nd} level of specialization		
	Semantic class of the VB_2 verb			Semantic class of the VB_2 verb			Semantic class of the VB_2 verb		
	no spec.	1^{st} level	2^{nd} level	no spec.	1^{st} level	2^{nd} level	no spec.	1^{st} level	2^{nd} level
no spec.	87.67	80.82	87.67	**90.41**	87.67	87.67	86.30	83.56	84.93
1^{st} level	89.04	82.19	87.67	87.67	89.04	87.67	87.67	86.30	84.93
2^{nd} level	87.67	83.56	87.67	**90.41**	**90.41**	89.04	89.04	87.67	86.30

Table 3: Accuracy of models using the 2^{nd} specialization level for the VB_2 agent semantic class

the VB_2 verb (*Mary intends John to revise the paper*). Let's assume that Mary is John's supervisor and she can make him revise the document. The sentence expresses Mary's intention of persuading John to revise the paper, but this objective is not encoded by the pattern we considered.

The semantic class of the VB_2 verb or **its specializations.** The VB_2 verb expresses the future action or behavior that the agent intends. We extract this feature using WordNet hierarchies. *Revise#1* belongs to the *act#1* semantic class.

A flag indicating if the VB_1 verb has an affirmative or a negative form. We want to differentiate between sentences like *John wants to go for a walk* and *John doesn't want to go for a walk.* The first sentence expresses John's intention, while, in the second one, no intention can be identified.

The type of the analyzed sentence. This feature is primarily concerned with questions. A question like *Where do you plan to go for a walk?* indicates the intention of *going for a walk*, unlike the question *Do you plan to go for a walk?* which might express an intention if the answer is "yes". This feature's values are the *wh-words* that begin a question or *n/a* for the other types of English sentences.

We did not analyze the affirmative versus the negative form of the VB_2 verb because it does not affect the objective attribute of the intention. The sentence *John intends not to go for a walk* expresses a negative intention. This sentence is much stronger than *John doesn't intend to go for a walk.* In the former context, John has set a goal for himself , while in the second sentence, the objective does not exist.

4 Experimental Results

4.1 Impact of specialization

The first experiment was performed using the LIBSVM package[6] and the WordNet semantic classes. These features yield an accuracy of 87.67%. Trying to improve the performance, we specialized the semantic classes. When the VB_2's agent semantic class was specialized, the accuracy remained constant. If we replace the VB_2's semantic class with its direct hyponyms, the accuracy drops 5.48%. But, the specialization of the VB_1 agent's semantic class brings an improvement of 1.37% and the specialization of the VB_1's class produces an increase in accuracy of 2.74%. Given this fluctuation in performance, we performed 81 different experiments which create SVM models using the same training data annotated with more general or more specific feature values. For each feature, we analyzed the first two semantic specialization levels.

From our experiments, we noticed that the specialization of the VB_2's agent semantic class does not influence the performance. Out of the 27 experiment triplets in which this specialization level changes, in only 4, it influences the result and, in 3 of them, the accuracy increases with the specialization level. Thus, our third feature is the second specialization level of the VB_2's agent class. Table 3 shows the results obtained when the values of the radial kernel parameters were chosen to optimize the 5-fold-cross-validation on the training data. The best models are described in Table 4.

Model	Level of specialization for the features
A	semantic class of the VB_1 agent, 1^{st} level of specialization for the VB_1's semantic class, and semantic class of the VB_2 verb
B	2^{nd} semantic level for the VB_1 agent class, 1^{st} level of the VB_1's semantic class, and the semantic class of the VB_2 verb
C	2^{nd} level of the VB_1 agent's semantic class and 1^{st} specialization levels for the VB_1 and VB_2 semantic classes

Table 4: The best three intention classifiers

4.2 Learning curves

We further analyzed our data and models and tried to see how many training examples are needed to reach 90.41% accuracy. We varied the training data

[6] http://www.csie.ntu.edu.tw/˜cjlin/libsvm/index.html

	Semantic class of the VB_1's agent	Semantic class of the VB_1 verb	Semantic class of the VB_2's agent	Semantic class of the VB_2 verb	VB_1 verb form	Sentence type
Model A	2.74	16.44	1.37	0	2.74	4.11
Model B	2.74	15.07	1.37	0	4.11	2.74
Model C	1.37	16.44	4.11	0	4.11	2.74

Table 5: The improvement (%) brought by each feature to the three best SVM models

size and validated the new models using our previous test set. Figure 1 shows the performance variation of three models that use feature sets identical in terms of specialization levels to the ones of the A, B, and C classifiers. All three models exhibit a similar behavior with respect to the change in the training set size. Therefore, our features create a stable algorithm. The highest accuracy models use all 300 training examples. Thus, we did not reach the saturation point, but, considering the performance curve, this point is not very far.

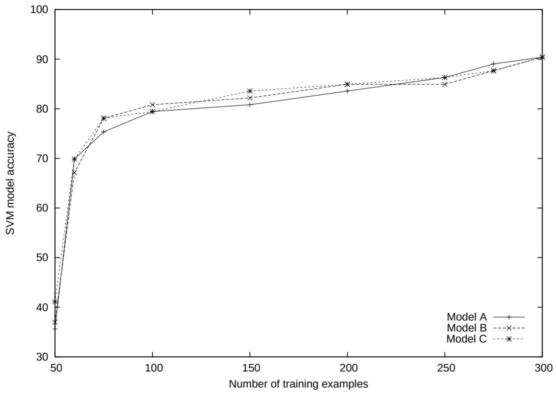


Figure 1: Testing set is constant

4.3 Feature impact on the SVM models

All our previous experiments used the entire set of features. Now, we investigate the relative contribution of each feature. We performed experiments that use only five out of the six features. In Table 5, we list the accuracy increase that is gained by the inclusion of each feature. The most influential attribute is the VB_1 verb's semantic class or its specializations. The intention's description verb does not influence the classification result. Because intentions consist of a future action and verbs express actions, there are very few verbs, such as *dream* or *snore* (involuntary actions) that cannot occupy the VB_2 verb's position. The syntactic features bring an average increase in accuracy of 3.50%.

4.4 Impact of word sense disambiguation

Perfect word sense disambiguation might be a too strong assumption. In this section, we examine the effects of weaker disambiguation. Table 6 shows the accuracies of the best three models when each concept is tagged with its first WordNet sense (**No** WSD) and when the senses are given by an in-house WSD system with an accuracy of 69% computed on the SemCor data (**Automatic** WSD).

	No WSD	**Automatic** WSD	**Gold** WSD
Model A	72.60	79.45	90.41
Model B	73.97	79.45	90.41
Model C	72.60	80.82	90.41

Table 6: Best models performance (%)

4.5 C5 results

After examining the SVM results, we applied the C5 machine learning algorithm (Quinlan, 2004) to the same training data annotated with the same feature set, in a similar manner. Again, we specialized the four semantic classes, independently, and tested the decision trees against the testing data. Table 7 shows their accuracy. The highest values were obtained for the first level of specialization of the VB_1 verb semantic class. The specialization levels of the other semantic classes do not influence the accuracy of the decision trees. The most tested attribute is the VB_1 verb. This further substantiates our observation, made during our SVM models analysis, that this feature has the greatest importance in the intention classification process. Our error analysis of the C5 results indicates that, because of the relatively small numbers of training instances, C5 ignores some of the features and makes wrong decisions.

5 Application to Question Answering

Questions involving intentions cannot be answered only by keyword-based or simple surface-level matching techniques. Table 8 lists two questions for

Q_1:	What was Putin trying to achieve by increasing military cooperation with North Korea?
QLF_1:	Putin(x_1) & INT(e_0, x_1, X) & ANS(X) & MANNER(X, e_2) & increase(e_2, x_1, x_2) & military(x_2) & cooperation (x_2) & with(x_2, x_3) & North_Korea(x_3)
A_1:	Putin is attempting [to restore Russia's influence in the East Asian region][INT]. The report said, the possibility remains that Russia could increase military cooperation with North Korea based on their treaty.
ALF_1:	Putin(x_1) & INT(e_0, x_1, e_1) & restore(e_1, x_1, x_2) & Russia(x_3) & 's(x_3, x_2) & influence(x_2) & LOCATION(x_2, x_4) & East(x_4) & Asian(x_4) & region(x_4) & report(x_5) & say(e_2, x_5, e_3) & possibility(x_6) & remains(e_3, x_6, e_4) & increase(e_4, x_3, x_7) & military(x_7) & cooperation(x_7) & with(x_7, x_8) & North_Korea(x_8) & base(e_4, x_9) & treaty(x_9)
Q_2:	From where does al Qaeda intend [to purchase weapons of mass destruction][INT]?
QLF_2:	alQaeda(x_1) & INT(e_0, x_1, e_1) & ANS(X) & LOCATION(e_1, X) & purchase(e_1, x_1, x_2) & weapons_of_mass_destruction(x_2)
A_2:	It is known that Osama bin Laden's al Qaeda network has tried [to buy ingredients for weapons of mass destruction in Russia][INT].
ALF_2:	Osama_bin_Laden(x_1) & 's(x_1, x_2) & al_Qaeda(x_2) & network(x_3) & IS-A(x_2, x_3) & INT(e_0, x_2, e_1) & buy(e_1, x_3, x_4) & ingredient(x_4) & PURPOSE(x_4, x_5) & weapons_of_mass_destruction(x_5) & LOCATION(e_1, x_6) & Russia(x_6)

Table 8: Question and answer pair examples

Semantic class of the VB_1's agent	**Semantic class of the VB_1 verb**		
	no spec.	**1^{st} level**	**2^{nd} level**
no spec.	79.45	**87.67**	84.93
1^{st} level	68.49	**87.67**	84.93
2^{nd} level	79.45	**87.67**	84.93

Table 7: C5 models accuracy (%)

which finding the correct answer primarily depends on the discovery of the INTENTION relation.

The answer type for the question Q_1 is the INTENTION argument itself. The question processing module will detect that the answer being sought is Putin's intention. The semantic relations module processes A_1's text and discovers the INTENTION relation. The question is searching for the intent of Putin with regards to North Korea and the answer text reveals Putin's intention to restore Russia's influence in the area. Question Q_2 is searching for a location as its answer type and the correct answer is one which involves al Qaeda intending to purchase weapons of mass destruction. The candidate answer text (A_2) reveals the organization's past intent to buy (synonym with *purchase*) weapons in Russia. Because the two intentions have the same agent, future action and theme, the two semantically enhanced logic forms can now be unified and we can pin down the location of the intent (*Russia*).

6 Conclusions

We proposed a method to detect the INTENT relation encoded by the sentence-level pattern VB_1 *to* VB_2 with a 90.41% accuracy. We plan to investigate the other INTENTION patterns as well as other semantic relations such as MOTIVE, IMPLICATION, or MEANING which, currently, cannot be identified by the state-of-the-art NLP systems. These relationships need to be analyzed to provide a complete coverage of the underlying semantics of text documents. We intend to incorporate our INTENTION detection module into a Question Answering system and show its impact.

References

Anscombe, G.E.M. 1957. *Intention*. Cornell University Press, Ithaca, New York.

Audi, Robert. 1973. Intending. *The Journal of Philosophy*, 70(13):387–403.

Bratman, Michael E. 1981. Intention and means-end reasoning. *The Philosophical Review*, 90(2):252–265.

Bratman, Michael E. 1987. *Intention, Plans, and Practical Reason*. Harvard University Press, Cambridge, Massachusetts.

Bratman, Michael E. 1990. What is intention? In *Intentions in Communication*. MIT Press.

Miller, George A., Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proceedings of the ARPA Human Language Technology Workshop*

Miller, George A. 1995. Wordnet: A lexical database. *Communication of the ACM*, 38(11):39–41.

Pollack, Martha E. 1990. Plans as complex mental attitudes. In *Intentions in Communication*. MIT Press.

Quinlan, Ross. 2004. Data Mining Tools See5 and C5.0. *http://www.rulequest.com/see5-info.html*

Wiebe, Janyce M., Theresa Wilson, Rebecca F. Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Computational Linguistics*, 30(3):277–308.

American Sign Language Generation: Multimodal NLG with Multiple Linguistic Channels

Matt Huenerfauth
Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104 USA
matt@huenerfauth.com

Abstract

Software to translate English text into American Sign Language (ASL) animation can improve information accessibility for the majority of deaf adults with limited English literacy. ASL natural language generation (NLG) is a special form of multimodal NLG that uses multiple linguistic output channels. ASL NLG technology has applications for the generation of gesture animation and other communication signals that are not easily encoded as text strings.

1 Introduction and Motivations

American Sign Language (ASL) is a full natural language – with a linguistic structure distinct from English – used as the primary means of communication for approximately one half million deaf people in the United States (Neidle et al., 2000, Liddell, 2003; Mitchell, 2004). Without aural exposure to English during childhood, a majority of deaf U.S. high school graduates (age 18) have only a fourth-grade (age 10) English reading level (Holt, 1991). Technology for the deaf rarely addresses this literacy issue; so, many deaf people find it difficult to read text on electronic devices. Software for translating English text into animations of a computer-generated character performing ASL can make a variety of English text sources accessible to the deaf, including: TV closed captioning, teletype telephones, and computer user-interfaces (Huenerfauth, 2005). Machine translation (MT) can also be used in educational software for deaf children to help them improve their English literacy skills.

This paper describes the design of our English-to-ASL MT system (Huenerfauth, 2004a, 2004b, 2005), focusing on ASL generation. This overview illustrates important correspondences between the problem of ASL natural language generation (NLG) and related research in Multimodal NLG.

1.1 ASL Linguistic Issues

In ASL, several parts of the body convey meaning in parallel: hands (location, orientation, shape), eye gaze, mouth shape, facial expression, head-tilt, and shoulder-tilt. Signers may also interleave lexical signing (LS) with classifier predicates (CP) during a performance. During LS, a signer builds ASL sentences by syntactically combining ASL lexical items (arranging individual signs into sentences). The signer may also associate entities under discussion with locations in space around their body; these locations are used in pronominal reference (pointing to a location) or verb agreement (aiming the motion path of a verb sign to/from a location).

During CPs, signers' hands draw a 3D scene in the space in front of their torso. One could imagine invisible placeholders floating in front of a signer representing real-world objects in a scene. To represent each object, the signer places his/her hand in a special handshape (used specifically for objects of that semantic type: moving vehicles, seated animals, upright humans, etc.). The hand is moved to show a 3D location, movement path, or surface contour of the object being described. For example, to convey the English sentence "the car parked next to the house," signers would indicate a location in space to represent the house using a special handshape for 'bulky objects.' Next, they would use a 'moving vehicle' handshape to trace a 3D path for the car which stops next to the house.

Proceedings of the ACL Student Research Workshop, pages 37–42,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

1.2 Previous ASL MT Systems

There have been some previous English-to-ASL MT projects – see survey in (Huenerfauth, 2003). Amid other limitations, none of these systems address how to produce the 3D locations and motion paths needed for CPs. A fluent, useful English-to-ASL MT system cannot ignore CPs. ASL sign-frequency studies show that signers produce a CP from 1 to 17 times per minute, depending on genre (Morford and MacFarlane, 2003). Further, it is those English sentences whose ASL translation uses a CP that a deaf user with low English literacy would need an MT system to translate. These English sentences look structurally different than their ASL CP counterpart – often making the English sentence difficult to read for a deaf user.

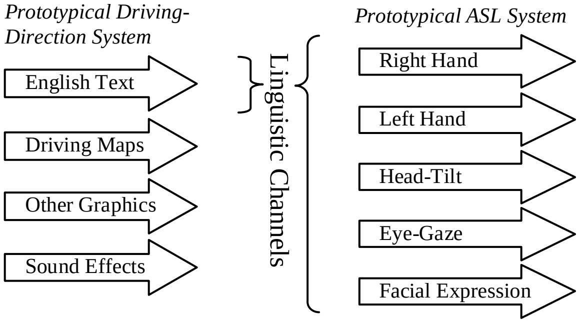
2 ASL NLG: A Form of Multimodal NLG

NLG researchers think of communication signals in a variety of ways: some as a written text, other as speech audio (with prosody, timing, volume, and intonation), and those working in Multimodal NLG as text/speech with coordinated graphics (maps, charts, diagrams, etc). Some Multimodal NLG focuses on "embodied conversational agents" (ECAs), computer-generated animated characters that communicate with users using speech, eye gaze, facial expression, body posture, and gestures (Cassell et al., 2000; Kopp et al., 2004).

The output of any NLG system could be represented as a stream of values (or features) that change over time during a communication signal; some NLG systems specify more values than others. Because the English writing system does not record a speaker's prosody, facial expression or gesture[1], a text-based NLG system specifies fewer communication stream values in its output than does a speech-based or ECA system. A text-based NLG system requires *literate* users, to whom it can transfer some of the processing burden; they must mentally reconstruct more of the language performance than do users of speech or ECA systems.

Since most writing systems are based on strings, text-based NLG systems can easily encode their output as a single stream, namely a sequence of words/characters. To generate more complex signals, multimodal systems decompose their output into several sub-streams – we'll refer to these as "channels." Dividing a communication signal into channels can make it easier to represent the various choices the generator must make; generally, a different processing component of the system will govern the output of each channel. The trade-off is that these channels must be coordinated over time.

Figure 1: Linguistic Channels in Multimodal Systems



Instead of thinking of channels as dividing a communication signal, we can think of them as groupings of individual values in the data stream that are related in some way. The channels of a multimodal NLG system generally correspond to natural perceptual/conceptual groupings called "modalities." Coarsely, audio and visual parts of the output are thought of as separate modalities. When parts of the output appear on different portions of the display, then they are also generally considered separate modalities. For instance, a multimodal NLG system for automobile driving directions may have separate processing channels for text, maps, other graphics, and sound effects. An ECA system may have separate channels for eye gaze, facial expression, manual gestures, and speech audio of the animated character.

When a language has no commonly-known writing system – as is the case for ASL – then it's not possible to build a text-based NLG system. We must produce an animation of a character (like an ECA) performing ASL; so, we must specify how the hands, eye gaze, mouth shape, facial expression, head-tilt, and shoulder-tilt are coordinated over time. With no conventional string-encoding of ASL (that would compress the signal into a single stream), an ASL signal is spread over multiple channels of the output – a departure from most Multimodal NLG systems, which have a single linguistic channel/modality that is coordinated with other non-linguistic resources (Figure 1).

[1] Some punctuation marks loosely correspond to intonation or pauses, but most prosodic information is lost. Facial expression and gesture is generally not conveyed in writing, except perhaps for the occasional use of "emoticons." ;-)

Of course, we could invent a string-based notation for ASL so that we could use traditional text-based NLG technology. (Since ASL has no writing system, we would have to invent an artificial notation.) Unfortunately, since the users of the system wouldn't be trained in this new writing system, it could not be used as output; we would still need to generate a multimodal animation output. An artificial writing system could only be used for internal representation and processing, However, flattening a naturally multichannel signal into a single-channel string (prior to generating a multichannel output) can introduce its own complications to the ASL system's design. For this reason, this project has been exploring ways to represent the hierarchical linguistic structure of information on multiple channels of ASL performance (and how these structures are coordinated or uncoordinated across channels over time).

Some multimodal systems have explored using linguistic structures to control (to some degree) the output of multiple channels. Research on generating animations of a speaking ECA character that performs meaningful gestures (Kopp et al., 2004) has similarities to this ASL project. First of all, the channels in the signal are basically the same; an animated human-like character is shown onscreen with information about eye, face, and arm movements being generated. However, an ASL system has no audio speech channel but potentially more fine-grained channels of detailed body movement.

The less superficial similarity is that (Kopp et. al, 2004) have attempted to represent the semantic meaning of some of the character's gestures and to synchronize them with the speech output. This means that, like an ASL NLG system, several channels of the signal are being governed by the linguistic mechanisms of a natural language. Unlike ASL, the gesture system uses the speech audio channel to convey nearly all of the meaning to the user; the other channels are generally used to convey additional/redundant information. Further, the internal structure of the gestures is not generally encoded in the system; they are typically atomic/lexical gesture events which are synchronized to co-occur with portions of speech output. A final difference is that gestures which co-occur with English speech (although meaningful) can be somewhat vague and are certainly less systematic and conventional than ASL body movements. So, while both systems may have multiple linguistic channels, the gesture system still has one primary linguistic channel (audio speech) and a few channels controlled in only a partially linguistic way.

3 This English-to-ASL MT Design

The linguistic and multimodal issues discussed above have had important consequences on the design of our English-to-ASL MT system. There are several unique features of this system caused by: (1) ASL having multiple linguistic channels that must be coordinated during generation, (2) ASL having both an LS and a CP form of signing, (3) CP signing visually conveying 3D spatial relationships in front of the signer's torso, and (4) ASL lacking a conventional written form. While ASL-particular factors influenced this design, section 5 will discuss how this design has implications for NLG of traditional written/spoken languages.

3.1 Coordinating Linguistic Channels

Section 2 mentioned that this project is developing multichannel (non-string) encodings of ASL animation; these encodings must coordinate multiple channels of the signal as they are generated by the linguistic structures and rules of ASL. Kopp et al. (2004) have explored how to coordinate meaningful gestures with speech signal during generation; however, their domain is somewhat simpler. Their gestures are atomic events without internal hierarchical structure. Our project is currently developing grammar-like coordination formalisms that allow complex linguistic signals on multiple channels to be conveniently represented.[2]

3.2 ASL Computational Linguistic Models

This project uses representations of discourse, semantics, syntax, and (sign) phonology tailored to ASL generation (Huenerfauth, 2004b). In particular, since this MT system will generate animations of classifier predicates (CPs), the system consults a 3D model of real-world scenes under discussion. Further, since multimodal NLG requires a form of scheduling (events on multiple channels are coordinated over a performance timeline), all of the linguistic models consulted and modified during ASL generation are time-indexed according to a timeline of the ASL performance being produced.

[2] Details of this work will be described in future publication.

Previous ASL phonological models were designed to represent non-CP ASL, but CPs use a reduced set of handshapes, standard eye-gaze and head-tilt patterns, and more complex orientations and motion paths. The phonological model developed for this system makes it easier to specify CPs.

Because ASL signers can use the space in front of their body to visually convey information, it is possible during CPs to show the exact 3D layout of objects being discussed. (The use of channels representing the hands means that we can now indicate 3D visual information – not possible with speech or text.) To represent this 3D detailed form of meaning, this system has an unusual *semantic* model for generating CPs. We populate the volume of space around the signer's torso with invisible 3D objects representing entities discussed by CPs being generated (Huenerfauth, 2004b). The semantic model is the set of placeholders around the signer (augmented with the CP handshape used for each). Thus, the semantics of the "car parked next to the house" example (section 1.1) is that a 'bulky' object occupies a particular 3D location and a 'vehicle' object moves toward it and stops.

Of course, the system will also need more traditional semantic representations of the information to be conveyed during generation, but this 3D model helps the system select the proper 3D motion paths for the signers' hands when "drawing" the 3D scenes during CPs. The work of (Kopp et al., 2004) studies gestures to convey spatial information during an English speech performance, but unlike this system, they use a logical-predicate-based semantics to represent information about objects referred to by gesture. Because ASL CPs indicate 3D layout in a linguistically conventional and detailed way, we use an actual 3D model of the objects being discussed. Such a 3D model may also be useful for ECA systems that wish to generate more detailed 3D spatial gesture animations.

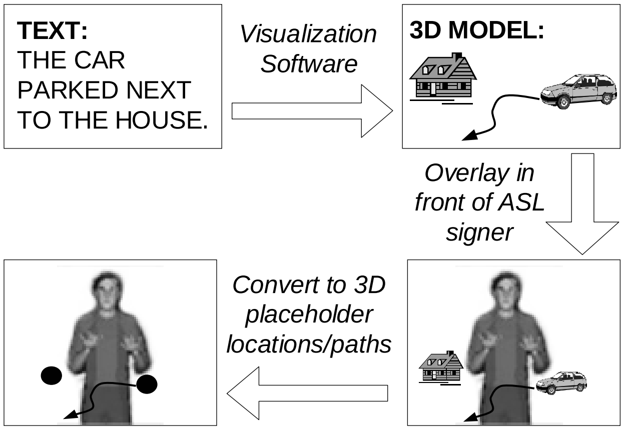
The discourse model in this ASL system records features not found in other NLG systems. It tracks whether a 3D location has been assigned to each discourse entity, where that location is around the signer, and whether the latest location of the entity has been indicated by a CP. The discourse model is not only relevant during CP performance; since ASL LS performance also assigns 3D locations to objects under discussion (for pronouns and verbal agreement), this model is also used for LS.

3.3 Generating 3D Classifier Predicates

An essential step in producing an animation of an ASL CP is the selection of 3D motion paths for the computer-generated signer's hands, eye gaze, and head tilt. The motion paths of objects in the 3D model described above are used to select corresponding motion paths for these parts of the signer's body during CPs. To build the 3D placeholder model, this system uses preexisting scene-visualization software to analyze an English text describing the motion of real-world objects and build a 3D model of how the objects mentioned in text are arranged and move (Huenerfauth, 2004b). This model is "overlaid" onto the volume in front of the ASL signer (Figure 2). For each object in the scene, a corresponding invisible placeholder is positioned in front of the signer; the layout of placeholders mimics the layout of objects in the 3D scene. In the "car parked next to the house" example, a miniature invisible object representing a 'house' is positioned in front of the signer's torso, and another object (with a motion path terminating next to the 'house') is added to represent the 'car.'

The locations and orientations of the placeholders are later used by the system to select the locations and orientations for the signer's hands while performing CPs about them. So, the motion path calculated for the car will be the basis for the 3D motion path of the signer's hand during the classifier predicate describing the car's motion. Given the information in the discourse/semantic models, the system generates the hand motions, head-tilt, and eye-gaze for a CP. It stores a library containing templates representing a prototypical form of each CP the system can produce. The templates

Figure 2: Converting English Text to 3D Placeholder



are planning operators (with logical pre-conditions, monitored termination conditions, and effects), allowing the system to “trigger” other elements of ASL signing performance that may be required during a CP. A planning-based NLG approach, described in (Huenerfauth, 2004b), is used to select a template, fill in its missing parameters, and build a schedule of the animation events on multiple channels needed to produce a sequence of CPs.

3.4 A Multi-Path Architecture

A multimodal NLG system may have several presentation styles it could use to convey information to its user; these styles may take advantage of the various output channels to different degrees. In ASL, there are multiple channels in the linguistic portion of the signal, and not surprisingly, the language has multiple sub-systems of signing that take advantage of the visual modality in different ways. ASL signers can select whether to convey information using lexical signing (LS) or classifier predicates (CPs) during an ASL performance (section 1.1). These two sub-systems use the space around the signer differently; during CPs, locations in space associated with objects under discussion must be laid out in a 3D manner corresponding to the topological layout of the real-world scene under discussion. Locations associated with objects during LS (used for pronouns and verb agreement) have no topological requirement. The layout of the 3D locations during LS may be arbitrary.

The CP generation approach in section 3.3 is computationally expensive; so, we would only like to use this processing pathway when necessary. English input sentences not producing classifier predicates would not need to be processed by the visualization software; in fact, most of these sentences could be handled using the more traditional MT technologies of previous systems. For this reason, our English-to-ASL MT system has multiple processing pathways (Huenerfauth, 2004a). The pathway for handling English input sentences that produce CPs includes the scene visualization software, while other input sentences undergo less sophisticated processing using a traditional MT approach (that is easier to implement). In this way, our CP generation component can actually be layered on top of a pre-existing English-to-ASL MT system to give it the ability to produce CPs. This multi-path design is equally applicable to the architecture of written-language MT systems. The design allows an MT system to combine a resource-intensive deep-processing MT method for difficult (or important) inputs and a resource-light broad-coverage MT method for other inputs.

3.5 Evaluation of Multichannel NLG

The lack of an ASL writing system and the multichannel nature of ASL can make NLG or MT systems which produce ASL animation output difficult to evaluate using traditional automatic techniques. Many such approaches compare a string produced by a system to some human-produced ‘gold-standard’ string. While we could invent an artificial ASL writing system for the system to produce as output, it’s not clear that human ASL signers could accurately or consistently produce written forms of ASL sentences to serve as ‘gold standards’ for such an evaluation. And of course, real users of the system would never be shown artificial “written ASL”; they would see full animations instead. User-based studies (where ASL signers evaluate animation output directly) may be a more meaningful measure of an ASL system.

We are planning such an evaluation of a prototype CP-generation module of the system during the summer/fall of 2005. Members of the deaf community who are native ASL signers will view animations of classifier predicates produced by the system. As a control, they will also be shown animations of CPs produced using 3D motion capture technology to digitally record the performance of CPs by other native ASL signers. Their evaluation of animations from both sources will be compared to measure the system’s performance. The multichannel nature of the signal also makes other interesting experiments possible. To study the system’s ability to animate the signer’s hands only, motion-captured ASL could be used to animate the head/body of the animated character, and the NLG system can be used to control only the hands of the character. Thus, channels of the NLG system can be isolated for evaluation – an experimental design only available to a multichannel NLG system.

4 Unique Design Features for ASL NLG

The design portion of this English-to-ASL project is nearly complete, and the implementation of the system is ongoing. Evaluations of the system will be available after the user-based study discussed above; however, the design itself has highlighted interesting issues about the requirements of NLG software for sign languages like ASL.

The multichannel nature of ASL has led this project to study mechanisms for coordinating the values of the linguistic models used during generation (including the output animation specification itself). The need to handle both the LS and CP subsystems of the language has motivated: a multipath MT architecture, a discourse model that stores data relevant to both subsystems, a model of the space around the signer capable of storing both LS and CP placeholders, and a phonological model whose values can be specified by either subsystem.

Since this English-to-ASL MT system is the first to address ASL classifier predicates, designing an NLG process capable of producing the 3D locations and paths in a CP animation has been a major design focus for this project. These issues have been addressed by the system's use of a 3D model of placeholders produced by scene-visualization software and a planning-based NLG process operating on templates of prototypical CP performance.

5 Applications Beyond Sign Language

Sign language NLG requires 3D spatial representations and multichannel coordinated output, but it's not unique in this requirement. In fact, generation of a communication signal for any language may require these capabilities (even for spoken languages like English). We have mentioned throughout this paper how gesture/speech ECA researchers may be interested in NLG technologies for ASL – especially if they wish to produce gestures that are more linguistically conventional, internally complex, or 3D-topologically precise.

Many other computational linguistic applications could benefit from an NLG design with multiple linguistic channels (and indirectly benefit from ASL NLG technology). For instance, NLG systems producing speech output could encode prosody, timing, volume, intonation, or other vocal data as multiple linguistically-determined channels of the output (in addition to a channel for the string of words being generated). And so, ASL NLG research not only has exciting accessibility benefits for deaf users, but it also serves as a research vehicle for NLG technology to produce a variety of richer-than-text linguistic communication signals.

Acknowledgments

I would like to thank my advisors Mitch Marcus and Martha Palmer for their guidance, discussion, and revisions during the preparation of this work.

References

Cassell, J., Sullivan, J., Prevost, S., and Churchill, E. (Eds.). 2000. Embodied Conversational Agents. Cambridge, MA: MIT Press.

Holt, J. 1991. Demographic, Stanford Achievement Test - 8th Edition for Deaf and Hard of Hearing Students: Reading Comprehension Subgroup Results.

Huenerfauth, M. 2003. Survey and Critique of ASL Natural Language Generation and Machine Translation Systems. Technical Report MS-CIS-03-32, Computer and Information Science, University of Pennsylvania.

Huenerfauth, M. 2004a. A Multi-Path Architecture for Machine Translation of English Text into American Sign Language Animation. In Proceedings of the Student Workshop of the Human Language Technologies conference / North American chapter of the Association for Computational Linguistics annual meeting: HLT/NAACL 2004, Boston, MA, USA.

Huenerfauth, M. 2004b. Spatial and Planning Models of ASL Classifier Predicates for Machine Translation. 10th International Conference on Theoretical and Methodological Issues in Machine Translation: TMI 2004, Baltimore, MD.

Huenerfauth, M. 2005. American Sign Language Spatial Representations for an Accessible User-Interface. In 3rd International Conference on Universal Access in Human-Computer Interaction. Las Vegas, NV, USA.

Kopp, S., Tepper, P., and Cassell, J. 2004. Towards Integrated Microplanning of Language and Iconic Gesture for Multimodal Output. Int'l Conference on Multimodal Interfaces, State College, PA, USA.

Liddell, S. 2003. Grammar, Gesture, and Meaning in American Sign Language. UK: Cambridge U. Press.

Mitchell, R. 2004. How many deaf people are there in the United States. Gallaudet Research Institute, Grad School & Prof. Progs. Gallaudet U. June 28, 2004. http://gri.gallaudet.edu/Demographics/deaf-US.php

Morford, J., and MacFarlane, J. 2003. Frequency Characteristics of ASL. Sign Language Studies, 3:2.

Neidle, C., Kegl, J., MacLaughlin, D., Bahan, B., and Lee R.G. 2000. The Syntax of American Sign Language: Functional Categories and Hierarchical Structure. Cambridge, MA: The MIT Press.

Using Emoticons to reduce Dependency in Machine Learning Techniques for Sentiment Classification

Jonathon Read
Department of Informatics
University of Sussex
United Kingdom
j.l.read@sussex.ac.uk

Abstract

Sentiment Classification seeks to identify a piece of text according to its author's general feeling toward their subject, be it positive or negative. Traditional machine learning techniques have been applied to this problem with reasonable success, but they have been shown to work well only when there is a good match between the training and test data with respect to topic. This paper demonstrates that match with respect to domain and time is also important, and presents preliminary experiments with training data labeled with emoticons, which has the potential of being independent of domain, topic and time.

1 Introduction

Recent years have seen an increasing amount of research effort expended in the area of understanding sentiment in textual resources. A sub-topic of this research is that of Sentiment Classification. That is, given a problem text, can computational methods determine if the text is generally *positive* or generally *negative*? Several diverse applications exist for this potential technology, ranging from the automatic filtering of abusive messages (Spertus, 1997) to an in-depth analysis of market trends and consumer opinions (Dave et al., 2003). This is a complex and challenging task for a computer to achieve — consider the difficulties involved in instructing a computer to recognise sarcasm, for example.

Previous work has shown that traditional text classification approaches can be quite effective when applied to the sentiment analysis problem. Models such as Naïve Bayes (NB), Maximum Entropy (ME) and Support Vector Machines (SVM) can determine the sentiment of texts. Pang et al. (2002) used a bag-of-features framework (based on unigrams and bigrams) to train these models from a corpus of movie reviews labelled as positive or negative. The best accuracy achieved was 82.9%, using an SVM trained on unigram features. A later study (Pang and Lee, 2004) found that performance increased to 87.2% when considering only those portions of the text deemed to be subjective.

However, Engström (2004) showed that the bag-of-features approach is **topic-dependent**. A classifier trained on movie reviews is unlikely to perform as well on (for example) reviews of automobiles. Turney (2002) noted that the unigram *unpredictable* might have a positive sentiment in a movie review (e.g. *unpredictable* plot), but could be negative in the review of an automobile (e.g. *unpredictable* steering). In this paper, we demonstrate how the models are also **domain-dependent** — how a classifier trained on product reviews is not effective when evaluating the sentiment of newswire articles, for example. Furthermore, we show how the models are **temporally-dependent** — how classifiers are biased by the trends of sentiment apparent during the time-period represented by the training data.

We propose a novel source of training data based on the language used in conjunction with emoticons in Usenet newsgroups. Training a classifier using this data provides a breadth of features that, while it

Proceedings of the ACL Student Research Workshop, pages 43–48,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

		Testing		
		FIN	M&A	MIX
Training				
NB	FIN	**80.3**	**75.5**	74.0
	M&A	77.5	75.3	75.8
	MIX	70.7	62.9	**84.6**
SVM	FIN	**78.8**	72.7	68.9
	M&A	74.5	**75.5**	75.5
	MIX	72.0	68.9	**81.1**

Figure 1: Topic dependency in sentiment classification. Accuracies, in percent. Best performance on a test set for each model is highlighted in bold.

does not perform to the state-of-the-art, could function independent of domain, topic and time.

2 Dependencies in Sentiment Classification

2.1 Experimental Setup

In this section, we describe experiments we have carried out to determine the influence of domain, topic and time on machine learning based sentiment classification. The experiments use our own implementation of a Naïve Bayes classifier and Joachim's (1999) SVM^{light} implementation of a Support Vector Machine classifier. The models were trained using unigram features, accounting for the presence of feature types in a document, rather than the frequency, as Pang et al. (2002) found that this is the most effective strategy for sentiment classification.

When training and testing on the same set, the mean accuracy is determined using three-fold cross-validation. In each case, we use a paired-sample t-test over the set of test documents to determine whether the results produced by one classifier are statistically significantly better than those from another, at a confidence interval of at least 95%.

2.2 Topic Dependency

Engström (2004) demonstrated how machine-learning techniques for sentiment classification can be topic dependent. However, that study focused on a three-way classification (positive, negative and neutral). In this paper, for uniformity across different data sets, we focus on only positive and negative sentiment. This experiment also provides an opportunity to evaluate the Naïve Bayes classifier as the previous work used SVMs.

We use subsets of a *Newswire* dataset (kindly provided by Roy Lipski of Infonic Ltd.) that relate to the topics of Finance (FIN), Mergers and Aquisitions (M&A) and a mixture of both topics (MIX). Each subset contains further subsets of articles of positive and negative sentiment (selected by independent trained annotators), each containing 100 stories. We trained a model on a dataset relating to one topic and tested that model using the other topics. Figure 1 shows the results of this experiment.

The tendency seems to be that performance in a given topic is best if the training data is from the same topic. For example, the Finance-trained SVM classifier achieved an accuracy of 78.8% against articles from Finance, but only 72.7% when predicting the sentiment of articles from M&A. However, statistical testing showed that the results are not significantly different when training on one topic and testing on another. It is interesting to note, though, that providing a dataset of mixed topics (the sub-corpus MIX) does not necessarily reduce topic dependency. Indeed, the performance of the classifiers suffers a great deal when training on mixed data (confidence interval 95%).

		Testing	
		Newswire	Polarity 1.0
Training			
NB	Newswire	**78.2**	57.6
	Polarity 1.0	53.2	**78.9**
SVM	Newswire	**78.2**	63.2
	Polarity 1.0	63.6	**81.5**

Figure 2: Domain dependency in sentiment classification. Accuracies, in percent. Best performance on a test set for each model is highlighted in bold.

2.3 Domain Dependency

We conducted an experiment to compare the accuracy when training a classifier on one domain (newswire articles or movie reviews from the *Polarity 1.0* dataset used by Pang et al. (2002)) and testing on the other domain. In Figure 2, we see a clear indication that models trained on one domain do not perform as well on another domain. All differences are significant at a confidence interval of 99.9%.

2.4 Temporal Dependency

To investigate the effect of time on sentiment classification, we constructed a new set of movie re-

		Testing	
Training		Polarity 1.0	Polarity 2004
NB	Polarity 1.0	**78.9**	71.8
	Polarity 2004	63.2	**76.5**
SVM	Polarity 1.0	**81.5**	77.5
	Polarity 2004	76.5	**80.8**

Figure 3: Temporal dependency in sentiment classification. Accuracies, in percent. Best performance on a test set for each model is highlighted in bold.

views, following the same approach used by Pang et al. (2002) when they created the Polarity 1.0 dataset. The data source was the Internet Movie Review Database archive[1] of movie reviews. The reviews were categorised as positive or negative using automatically extracted ratings. A review was ignored if it was not written in 2003 or 2004 (ensuring that the review was written after any in the Polarity 1.0 dataset). This procedure yielded a corpus of 716 negative and 2,669 positive reviews. To create the *Polarity 2004*[2] dataset we randomly selected 700 negative reviews and 700 positive reviews, matching the size and distribution of the Polarity 1.0 dataset.

The next experiment evaluated the performance of the models first against movie reviews from the same time-period as the training set and then against reviews from the other time-period. Figure 3 shows the resulting accuracies.

These results show that while the models perform well on reviews from the same time-period as the training set, they are not so effective on reviews from other time-periods (confidence interval 95%). It is also apparent that the Polarity 2004 dataset performs worse than the Polarity 1.0 dataset (confidence interval 99.9%). A possible reason for this is that Polarity 2004 data is from a much smaller time-period than that represented by Polarity 1.0.

3 Sentiment Classification using Emoticons

One way of overcoming the domain, topic and time problems we have demonstrated above would be to find a source of much larger and diverse amounts of general text, annotated for sentiment. Users of electronic methods of communication have developed visual cues that are associated with emotional states in an attempt to state the emotion that their text represents. These have become known as *smileys* or *emoticons* and are glyphs constructed using the characters available on a standard keyboard, representing a facial expression of emotion — see Figure 4 for some examples. When the author of an electronic communication uses an emoticon, they are effectively marking up their own text with an emotional state. This marked-up text can be used to train a sentiment classifier if we assume that a *smile* indicates generally positive text and a *frown* indicates generally negative text.

Glyph	*Meaning*	*Frequency*
:-)	smile	3.8739
;-)	wink	2.4350
:-(	frown	0.4961
:-D	wide grin	0.1838
:-P	tongue sticking out	0.1357
:-O	surprise	0.0171
:-|	disappointed	0.0146
:'(	crying	0.0093
:-S	confused	0.0075
:-@	angry	0.0038
:-$	embarrassed	0.0007

Figure 4: Examples of emoticons and the frequency of usage observed in Usenet articles, in percent. For example, 2.435% of downloaded Usenet articles contained a wink emoticon.

3.1 Emoticon Corpus Construction

We collected a corpus of text marked-up with emoticons by downloading Usenet newsgroups and saving an article if it contained an emoticon listed in Figure 4. This process resulted in 766,730 articles being stored, from 10,682,455 messages in 49,759 newsgroups inspected. Figure 4 also lists the percentage of documents containing each emoticon type, as observed in the Usenet newsgroups.

We automatically extracted the paragraph(s) containing the emoticon of interest (a smile or a frown) from each message and removed any superfluous formatting characters (such as those used to indicate article quotations in message threads). In order to prevent quoted text from being considered more than once, any paragraph that began with exactly the same thirty characters as a previously observed paragraph was disregarded. Finally, we used the classifier developed by Cavnar and Trenkle (1994) to filter

[1] `http://reviews.imdb.com/Reviews/`

[2] The new datasets described in this paper are available at `http://www.sussex.ac.uk/Users/jlr24/data`

	Finance	M&A	Mixed
NB	46.0 ± 2.1	55.8 ± 3.8	49.0 ± 1.6
SVM	50.3 ± 1.7	57.8 ± 6.5	55.5 ± 2.7

Figure 5: Performance of Emoticon-trained classifier across topics. Mean accuracies with standard deviation, in percent.

	Newswire	Polarity 1.0
NB	50.3 ± 2.2	56.8 ± 1.8
SVM	54.4 ± 2.8	54.0 ± 0.8

Figure 6: Performance of Emoticon-trained classifiers across domains. Mean accuracies with standard deviation, in percent.

out any paragraphs of non-English text. This process yielded a corpus of 13,000 article extracts containing frown emoticons. As investigating skew between positive and negative distributions is outside the scope of this work, we also extracted 13,000 article extracts containing smile emoticons. The dataset is referred to throughout this paper as *Emoticons* and contains 748,685 words.

3.2 Emoticon-trained Sentiment Classification

This section describes how the Emoticons corpus[3] was optimised for use as sentiment classification training data. 2,000 articles containing smiles and 2,000 articles containing frowns were held-out as optimising test data. We took increasing amounts of articles from the remaining dataset (from 2,000 to 22,000 in increments of 1,000, an equal number being taken from the positive and negative sets) as optimising training data. For each set of training data we extracted a context of an increasing number of tokens (from 10 to 1,000 in increments of 10) both before and in a window[4] around the smile or frown emoticon. The models were trained using this extracted context and tested on the held-out dataset.

The optimisation process revealed that the best-performing settings for the Naïve Bayes classifier was a window context of 130 tokens taken from the largest training set of 22,000 articles. Similarly, the best performance for the SVM classifier was found using a window context of 150 tokens taken from 20,000 articles.

	Polarity 1.0	Polarity 2004
NB	56.8 ± 1.8	56.7 ± 2.2
SVM	54.0 ± 0.8	57.8 ± 1.8

Figure 7: Performance of Emoticon-trained classifier across time-periods. Mean accuracies with standard deviation, in percent.

The classifiers' performance in predicting the smiles and frowns of article extracts was verified using these optimised parameters and ten-fold cross-validation. The mean accuracy of the Naïve Bayes classifier was 61.5%, while the SVM classifier was 70.1%.

Using these same classifiers to predict the sentiment of movie reviews in Polarity 1.0 resulted in accuracies of 59.1% (Naïve Bayes) and 52.1% (SVM).

We repeated the optimisation process using a held-out set of 100 positive and 100 negative reviews from the Polarity 1.0 dataset, as it is possible that this test needs different parameter settings. This revealed an optimum context of a window of 50 tokens taken from a training set of 21,000 articles for the Naïve Bayes classifier. Interestingly, the optimum context for the SVM classifier appeared to be a window of only 20 tokens taken from a mere 2,000 training examples. This is clearly an anomaly, as these parameters resulted in an accuracy of 48.9% when testing against the reserved reviews of Polarity 1.0. We attribute this to the presence of noise, both in the training set and in the held-out set, and discuss this below (Section 4.2). The second-best parameters according to the optimisation process were a context of 510 tokens taken before an emoticon, from a training set of 20,000 examples.

We used these optimised parameters to evaluate the sentiments of texts in the test sets used to evaluate dependency in Section 2. Figures 5, 6 and 7 show the final, optimised results across topics, domains and time-periods respectively. These tables report the average accuracies over three folds, with the standard deviation as a measure of error.

4 Discussion

The emoticon-trained classifiers perform well (up to 70% accuracy) when predicting the sentiment of article extracts from the Emoticons dataset, which is encouraging when one considers the high level of

[3] Note that in these experiments the emoticons are used as anchors from which context is extracted, but are removed from texts before they are used as training or test data.

[4] Context taken after an emoticon was also investigated, but was found to be inferior. This is because approximately two-thirds of article extracts end in an emoticon so when using after-context few features are extracted.

Training	Testing	Coverage
Polarity 1.0	Polarity 1.0 (three-fold cross-validation)	69.8
Emoticons	FIN	54.9
	M&A	58.1
	MIX	60.2
	Newswire	46.1
	Polarity 1.0	41.1
	Polarity 2004	42.6

Figure 8: Coverage of classifiers, in percent.

noise that is likely to be present in the dataset.

However, they perform only a little better than one would expect by chance when classifying movie reviews, and are not effective in predicting the sentiment of newswire articles. This is perhaps due to the nature of the datasets — one would expect language to be informal in movie reviews, and even more so in Usenet articles. In contrast, language in newswire articles is far more formal. We might therefore infer a further type of dependence in sentiment classification, that of language-style dependency.

Also, note that neither machine-learning model consistently out-performs the other. We speculate that this, and the generally mediocre performance of the classifiers, is due (at least) to two factors; poor coverage of the features found in the test domains and a high level of noise found in Usenet article extracts. We investigate these factors below.

4.1 Coverage

Figure 8 shows the coverage of the Emoticon-trained classifiers on the various test sets. In these experiments, we are interested in the coverage in terms of unique token types rather than the frequency of features, as this more closely reflects the training of the models (see Section 2.1). The mean coverage of the Polarity 1.0 dataset during three-fold cross-validation is also listed as an example of the coverage one would expect from a better-performing sentiment classifier. The Emoticon-trained classifier has much worse coverage in the test sets.

We analysed the change in coverage of the Emoticon-trained classifiers on the Polarity 1.0 dataset. We found that the coverage continued to improve as more training data was provided; the coverage of unique token types was improving by about 0.6% per 1,000 training examples when the Emoticons dataset was exhausted.

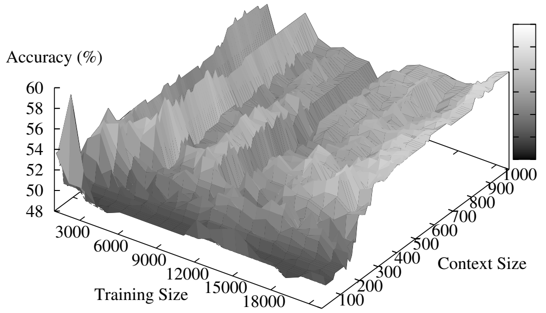


Figure 9: Change in Performance of the SVM Classifier on held-out reviews from Polarity 1.0, varying training set size and window context size. The datapoints represent 2,200 experiments in total.

It appears possible that more training data will improve the performance of the Emoticon-trained classifiers by increasing the coverage. Potential sources for this include online bulletin boards, chat forums, and further newsgroup data from Usenet and Google Groups[5]. Future work will utilise these sources to collect more examples of emoticon use and analyse any improvement in coverage and accuracy.

4.2 Noise in Usenet Article Extracts

The article extracts collected in the Emoticons dataset may be noisy with respect to sentiment. The SVM classifier seems particularly affected by this noise. Figure 9 depicts the change in performance of the SVM classifier when varying the training set size and size of context extracted. There are significant spikes apparent for the training sizes of 2,000, 3,000 and 6,000 article extracts (as noted in Section 3.2), where the accuracy suddenly increases for the training set size, then quickly decreases for the next set size. This implies that the classifier is discovering features that are useful in classifying the held-out set, but the addition of more, noisy, texts soon makes the information redundant.

Some examples of noise taken from the Emoticons dataset are: mixed sentiment, e.g.

[5] `http://groups.google.com`

> "Sorry about venting my frustration here but I just lost it. :-(Happy thanks giving everybody :-)",

sarcasm, e.g.

> "Thank you so much, that's really encouraging :-(",

and spelling mistakes, e.g.

> "The movies where for me a major desapointment :-(".

In future work we will investigate ways to remove noisy data from the Emoticons dataset.

5 Conclusions and Future Work

This paper has demonstrated that dependency in sentiment classification can take the form of domain, topic, temporal and language style. One might suppose that dependency is occurring because classifiers are learning the semantic sentiment of texts rather than the general sentiment of language used. That is, the classifiers could be learning authors' sentiment towards named entities (e.g. actors, directors, companies, etc.). However, this does not seem to be the case. In a small experiment, we part-of-speech tagged the Polarity 2004 dataset and automatically replaced proper nouns with placeholders. Retraining on this modified text did not significantly affect performance.

But it may be that something more subtle is happening. Possibly, the classifiers are learning the words associated with the semantic sentiment of entities. For example, suppose that there has been a well-received movie about mountaineering. During this movie, there is a particularly stirring scene involving an ice-axe and most of the reviewers mention this scene. During training, the word 'ice-axe' would become associated with a positive sentiment, whereas one would suppose that this word does not in general express any kind of sentiment.

In future work we will perform further tests to determine the nature of dependency in machine learning techniques for sentiment classification. One way of evaluating the 'ice-axe' effect could be to build a 'pseudo-ontology' of the movie reviews — a map of the sentiment-bearing relations that would enable the analysis of the dependencies created by the training process. Other extensions of this work are to collect more text marked-up with emoticons, and to experiment with techniques to automatically remove noisy examples from the training data.

Acknowledgements

This research was funded by a UK EPSRC studentship. I am very grateful to Thorsten Joachims, Roy Lipski, Bo Pang and John Trenkle for kindly making their data or software available, and to the anonymous reviewers for their constructive comments. Thanks also to Nick Jacobi for his discussion of the 'ice-axe' effect. Special thanks to my supervisor, John Carroll, for his continued advice and encouragement.

References

W. B. Cavnar and J. M. Trenkle. 1994. N-Gram-Based Text Categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, Nevada.

Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In *Proceedings of the International World Wide Web Conference*, Budapest, Hungary.

Charlotta Engström. 2004. Topic Dependence in Sentiment Classification. Master's thesis, University of Cambridge, July.

T. Joachims. 1999. Making large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

Bo Pang and Lillian Lee. 2004. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, University of Pennsylvania.

Ellen Spertus. 1997. Smokey: Automatic Recognition of Hostile Messages. In *Proceedings of the Innovative Applications of Artificial Intelligence*.

Peter D. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 417–424, Philadelphia, Pennsylvania.

Learning Meronyms from Biomedical Text

Angus Roberts
Department of Computer Science, University of Sheffield,
Regent Court, 211 Portobello Street, Sheffield S1 4DP
a.roberts@dcs.shef.ac.uk

Abstract

The part-whole relation is of special importance in biomedicine: structure and process are organised along partitive axes. Anatomy, for example, is rich in part-whole relations. This paper reports preliminary experiments on part-whole extraction from a corpus of anatomy definitions, using a fully automatic iterative algorithm to learn simple lexico-syntactic patterns from multiword terms. The experiments show that meronyms can be extracted using these patterns. A failure analysis points out factors that could contribute to improvements in both precision and recall, including pattern generalisation, pattern pruning, and term matching. The analysis gives insights into the relationship between domain terminology and lexical relations, and into evaluation strategies for relation learning.

1 Introduction

We are used to seeing words listed alphabetically in dictionaries. In terms of meaning, this ordering has little relevance beyond shared roots. In the OED, `jam` is sandwiched between `jalpaite` (a sulphide) and `jama` (a cotton gown). It is a long way from `bread` and `raspberry`[1]. Vocabularies, however, do have a natural structure: one that we rely on for language understanding. This structure is defined in part by lexical, or sense, relations, such as the familiar relations of synonymy and hyponymy (Cruse, 2000). Meronymy relates the lexical item for a part to that for a whole, equivalent to the conceptual relation of *partOf*[2]. Example 1 shows a meronym. When we read the text, we understand that the `frontal lobes` are not a new entity unrelated to what has gone before, but part of the previously mentioned `brain`.

(1) MRI sections were taken through the <u>brain</u>. <u>Frontal lobe</u> shrinkage suggests a generalised cerebral atrophy.

The research described in this paper considers meronymy, and its extraction from text. It is taking place in the context of the Clinical e-Science Framework (CLEF) project [3], which is developing information extraction (IE) tools to allow querying of medical records. Both IE and querying require domain knowledge, whether encoded explicitly or implicitly. In IE, domain knowledge is required to resolve co-references between textual entities, such as those in Example 1. In querying, domain knowledge is required to expand and constrain user expressions. For example, the query in Example 2 should retrieve sarcomas in the pelvis, but not in limbs.

(2) Retrieve patients on Gemcitabine with advanced sarcomas in the trunk of the body.

The part-whole relation is critical to domain knowledge in biomedicine: the structure and function of biological organisms are organised along partitive axes. The relation is modelled in several medical knowledge resources (Rogers and Rector, 2000),

[1] Oxford English Dictionary, Second Edition, 1989.

[2] Although it is generally held that *partOf* is not just a single simple relation, this will not be considered here.

[3] http://www.clef-user.com/

Proceedings of the ACL Student Research Workshop, pages 49–54,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

but they are incomplete, costly to maintain, and unsuitable for language engineering. This paper looks at simple lexico-syntactic techniques for learning meronyms. Section 2 considers background and related work; Section 3 introduces an algorithm for relation extraction, and its implementation in the PartEx system; Section 4 considers materials and methods used for experiments with PartEx. The experiments are reported in Section 5, followed by conclusions and suggestions for future work.

2 Related Work

Early work on knowledge extraction from electronic dictionaries used lexico-syntactic patterns to build relational records from definitions. This included some work on *partOf* (Evens, 1988). Lexical relation extraction has, however, concentrated on hyponym extraction. A widely cited method is that of Hearst (1992), who argues that specific lexical relations are expressed in well-known intra-sentential lexico-syntactic patterns. Hearst successfully extracted hyponym relations, but had little success with meronymy, finding that meronymic contexts are ambiguous (for example, `cat’s paw` and `cat’s dinner`). Morin (1999) reported a semi-automatic implementation of Hearst’s algorithm. Recent work has applied lexical relation extraction to ontology learning (Maedche and Staab, 2004).

Berland and Charniak (1999) report what they believed to be the first work finding part-whole relations from unlabelled corpora. The method used is similar to that of Hearst, but includes metrics for ranking proposed part-whole relations. They report 55% accuracy for the top 50 ranked relations, using only the two best extraction patterns.

Girju (2003) reports a relation discovery algorithm based on Hearst. Girju contends that the ambiguity of part-whole patterns means that more information is needed to distinguish meronymic from non-meronymic contexts. She developed an algorithm to learn semantic constraints for this differentiation, achieving 83% precision and 98% recall with a small set of manually selected patterns. Others have looked specifically at meronymy in anaphora resolution (e.g. Poesio et al (2002)).

The algorithm presented here learns relations directly between semantically typed multiword terms, and itself contributes to term recognition. Learning is automatic, with neither manual selection of best patterns, nor expert validation of patterns. In these respects, it differs from earlier work. Hearst and others learn relations between either noun phrases or single words, while Morin (1999) discusses how hypernyms learnt between single words can be projected onto multi-word terms. Earlier algorithms include manual selection of initial or “best” patterns. The experiments differ from others in that they are restricted to a well defined domain, anatomy, and use existing domain knowledge resources.

```
Input:                              Output:

  • A lexicon                         • New relations
  • Relations between                 • New terms
    terms                             • Context patterns
  • Corpus from which
    to learn

Steps:

  1. Using input resources
       (a) Label terms
       (b) Label relations

  2. For a fixed number of iterations or until no
     new relations are learned
       (a) Identify contexts that contain both
           participants in a relation
       (b) Create patterns describing contexts
       (c) Generalise the patterns
       (d) Use generalised patterns to identify new
           relation instances
       (e) Label new terms
       (f) Label new relations
```

Figure 1: PartEx algorithm for relation discovery

3 Algorithm

Input to the algorithm consists of existing lexical and relational resources, such as terminologies and ontologies. These are used to label text with training relations. The context of these relations are found automatically, and patterns created to describe these contexts. These patterns are generalised and used to discover new relations, which are fed back iteratively into the algorithm. The algorithm is given in Figure 1. An example iteration is shown in Figure 2.

3.1 Discovering New Terms

Step 2e in Figure 1 labels new terms, which may be discovered as a by-product of identifying new rela-

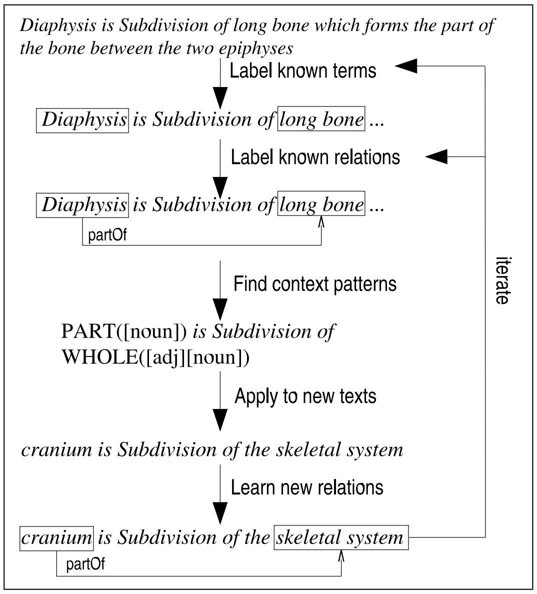

Diaphysis is Subdivision of long bone which forms the part of the bone between the two epiphyses
Label known terms
Diaphysis is Subdivision of long bone ...
Label known relations
Diaphysis is Subdivision of long bone ...
partOf
Find context patterns
iterate
PART([noun]) is Subdivision of WHOLE([adj][noun])
Apply to new texts
cranium is Subdivision of the skeletal system
Learn new relations
cranium is Subdivision of the skeletal system
partOf

Figure 2: PartEx relation discovery between terms, patterns represented by tokens and parts of speech.

tion instances. This is possible because there is a distinction between the lexical item used to find the pattern context (Step 2a), and the pattern element against which new relations are matched (Step 2d). For example, a pattern could be found from the context (`term` *relation* `term`), and expressed as (`noun` *relation* `adjective noun`). When applied to the text to learn new relation instances, sequences of tokens taking part in this relation will be found, and may be inferred to be terms for the next iteration.

3.2 Implementation: PartEx

Implementation was independent of any specific relation, but configured, as the PartEx system, to discover *partOf*. Relations were usually learned between terms, although this was varied in some experiments. The algorithm was implemented using the GATE NLP framework (Cunningham et al., 2002) and texts preprocessed using the tokeniser, sentence splitter, and part-of-speech (POS) tagger provided with GATE. In training, terms were labelled using MMTx, which uses lexical variant generation to map noun phrases to candidate terms and concepts attested in a terminology database. Final candidate selection is based on linguistic matching metrics, and concept resolution on filtering ambiguity from the MMTx source terminologies (Aronson, 2001). Training relations were labelled from an existing meronymy. Simple contexts of up to five tokens between the participants in the relation were identified using JAPE, a regular expression language integrated into GATE. For some experiments, relations were considered between noun phrases, labelled using LT CHUNK (Mikheev and Finch, 1997). GATE wrappers for MMTx, LT CHUNK, and other PartEx modules are freely available [4].

Patterns describing contexts were expressed as shallow lexico-syntactic patterns in JAPE, and a JAPE transducer used to find new relations. A typical pattern consisted of a sequence of parts of speech and words. Pattern generalisation was minimal, removing only those patterns that were either identical to another pattern, or that had more specific lexico-syntactic elements of another pattern. To simplify pattern creation for the experiments reported here, patterns only used context between the relation participants, and did not use regular expression quantifiers. New terms found during relation discovery were labelled using a finite state machine created with the Termino compiler (Harkema et al., 2004).

4 Materials and Method

Lexical and relational resources were provided by the Unified Medical Language System (UMLS), a collection of medical terminologies [5]. Term lookup in the training phase was carried out using MMTx. Experiments made particular use of The University of Washington Digital Anatomist Foundational Model (UWDA), a knowledge base of anatomy included in UMLS. Relation labelling in the training phase used a meronymy derived by computing the transitive closure of that provided with the UWDA.

The UWDA gives definitions for some terms, as headless phrases that do not include the term being defined. A corpus was constructed from these, for learning and evaluation. This corpus used the first 300 UWDA terms with a definition, that had a UMLS semantic type of “Body Part”. These terms included synonyms and orthographic variants given the same definition. Complete definitions were constructed by prepending terms to definitions with the copula “is”. An example is shown in Figure 2.

[4] http://www.dcs.shef.ac.uk/∼angus

[5] Version 2003AC, http://www.nlm.nih.gov/research/umls/

Experiments were carried out using cross validation over ten random unseen folds, with 71 unique meronyms across all ten folds. Definitions were pre-processed by tokenising, sentence splitting, POS tagging and term labelling. Evaluation was carried out by comparison of relations learned in the held back fold, to those in an artificially generated gold standard (described below). Evaluation was type based, rather than instance based: unique relation instances in the gold standard were compared with unique relation instances found by PartEx, i.e. identical relation instances found within the same fold were treated as a single type. Evaluation therefore measures domain knowledge discovery.

Gold standard relations were generated using the same context window as for Step 2a of the algorithm. Pairs of terms from each context were checked automatically for a relation in UWDA, and this added to the gold standard. This evaluation strategy is not ideal. First, the presence of a part and a whole in a context does not mean that they are being meronymically related (for example, “found in the hand and finger”). The number of spurious meronyms in the gold standard has not yet been ascertained. Second, a true relation in the text may not appear in a limited resource such as the UWDA (although this can be overcome through a failure analysis, as described in Section 4.1). Although a better gold standard would be based on expert mark up of the text, the one used serves to give quick feedback with minimal cost. Standard evaluation metrics were used. The accuracy of initial term and relation labelling were not evaluated, as these are identical in both gold standard creation and in experiments.

4.1 Failure Analysis

For some experiments, a failure analysis was carried out on missing and spurious relations. The reasons for failure were hypothesised by examining the sentence in which the relation occurred, the pattern that led to its discovery, and the source of the pattern.

Some spurious relations appeared to be correct, even though they were not in the gold standard. This is because the gold standard is based on a resource which itself has limits. One of the aims of the work is to supplement such resources: the algorithm *should* find correct relations that are not in the resource. Proper evaluation of these relations requires care, and methodologies are currently being investigated. A quick measure of their contribution was, however, found by applying a simple methodology, based on the source texts being definitional, authoritative, and describing relations in unambiguous language. The methodology adjusts the number of spurious relations, and calculates a *corrected precision*. By leaving the number of actual relations unchanged, corrected precision still reflects the proportion of discovered relations that were correct relative to the gold standard, but also reflects the number of correct relations not in the gold standard. The methodology followed the steps in Figure 3.

```
1. Examine the context of the relation.

2. If the text gives a clear statement of
   meronomy, the relation is not spurious.

3. If the text is clearly not a statement of
   meronomy, the relation is spurious.

4. If the text is ambiguous, refer to a second
   authoritative resource[6].  If this gives a
   clear statement of meronomy, the relation is
   not spurious.

5. If none of these apply, the relation is
   spurious.

6. Calculate corrected precision from the new
   number of spurious relations.
```

Figure 3: Calculating corrected precision.

5 Experimental Results

Table 3 shows the results of running PartEx in various configurations, and evaluating over the same ten folds. The first configuration, labelled BASE, used PartEx as described in Section 3.2, to give a recall of 0.80 and precision of 0.25. A failure analysis for this configuration is given in Table 2. It shows that the largest contribution to spurious relations (i.e. to lack of precision), was due to relations discovered by some pattern that is ambiguous for meronymy (category PATTERN). For example, the pattern “`[noun]` *and* `[noun]`” finds the incorrect meronym “`median` *partOf* `lateral`” from the text “median and lateral glossoepiglottic folds”. The algorithm learned the pattern from a correct meronym, and applying it in the next iteration, learned spurious relations, compounding the error.

[6] In this case, Clinically Oriented Anatomy. K. Moore and A. Dalley. 4th Edition. 1999. Lippincott Williams and Wilkins.

Category	Description	Count	%
SPECIFIC	There are one or more variant patterns that come close to matching this relation, but none specific to it.	10	50%
DISCARD	Patterns that could have picked these up were discarded, as they were also generating spurious patterns.	7	35%
SCARCE	The context is unique in the corpus, and so a pattern could not be learnt without generalisation.	3	15%
COMPOUND	The relation is within a compound noun. These are not recognised by the discovery algorithm.	1	5%
COMPLEX	Complex context, which is beyond the simple current “part token* whole” context.	1	5%

Table 1: Failure analysis of 20 missing relations over ten folds, using PartEx configuration FILT.

Category	Description	BASE		FILT	
		Count	%	Count	%
PATTERN	The pattern used to discover the relation does not encode partonomy in this case (Patterns involving: *is* 33 (69%); *and* 10 (21%); *or* 3 (6%); other 2 (4%)).	48	43%	0	0%
CORRECT	Although not in the gold standard, the relation is clearly correct, either from an unambiguous statement of fact in the text from which it was mined, or by reference to a standard anatomy textbook.	30	27%	33	49%
DEEP	The relation is within a deeper structure than the surface patterns considered. The algorithm has found an incorrect relation that relates to this deep structure. For example, the text “limen nasi is subdivision of surface of viscerocranial mucosa” leads to (`limen nasi` *partOf* `surface`).	12	11%	14	21%
FRAGMENT:DEEP	A combination of the FRAGMENT and DEEP categories. For example, given the text “nucleus of nerve is subdivision of neural tree”, it has learnt that (`subdivision` *partOf* `neural`).	10	9%	4	6%
FRAGMENT	The relation is a fragment of one in the text. For example, “plica salpingopalatine is subdivision of viscerocranial mucosa” leads to (`plica salpingopalatine` *partOf* `viscerocranial`).	9	8%	12	18%
OTHER	Other reason.	4	4%	3	5%

Table 2: Failure analysis of spurious part-whole relations found by PartEx, for configuration BASE (over half the spurious relations across ten folds) and configuration FILT (all spurious relations in ten folds). In each case, a small number of relations are in two categories.

	Possible	Actual	Missing	Spurious	P	R
BASE	71	56	15	168	0.25	**0.80**
FILT	71	51	20	67	0.43	0.73
CORR	71	51	20	34	**0.58**	0.73
ITR1	71	45	26	66	0.39	0.62
ITR2	71	51	20	67	0.43	0.73
TERM	71	51	20	213	0.20	0.74
TOK	30	26	4	266	0.09	0.88
NP	32	27	5	393	0.07	0.81
POS	71	21	50	749	0.03	0.32

Table 3: Evaluation of PartEx. Total number of relations, mean precision (P) and mean recall (R) for various configurations, as discussed in the text.

The bulk of the spurious results of this type were learnt from patterns using the tokens *and, is,* and *or*.

This problem needs a principled solution, perhaps based on pruning patterns against a held-out portion of training data, or by learning ambiguous patterns from a large general corpus. Such a solution is being developed. In order to mimic it for the purpose of these experiments, a filter was built to remove patterns derived from problematic contexts. Table 3 shows the results of this change, as configuration FILT: precision rose to 0.43, and recall dropped. All other experiments reported used this filter.

A failure analysis of missing relations from configuration FILT is shown in Table 1. The drop in recall is explained by PartEx filtering ambiguous patterns. The biggest contribution to lack of recall was over-specific patterns (for example, the pattern “`[term]` *is part of* `[term]`“ would not identify the meronym in “finger is a part of the hand”. Generalisation of patterns is essential to improve recall. Improvements could also be made with more sophisticated context, and by examining compounds.

A failure analysis of spurious relations for configuration FILT is shown in Table 2. The biggest impact on precision was made by relations that could be considered correct, as discussed in Section 4.1. A corrected precision of 0.58 was calculated, shown as configuration CORR in Table 3. Two other factors affecting precision can be deduced from Table 2. First, some relations were encoded in deeper linguistic structures than those considered (category DEEP). Improvements could be made to precision by considering these deeper structures. Second, some spurious relations were found between fragments of terms, due to failure of term recognition.

The algorithm used by PartEx is iterative, the implementation completing in two iterations. Configurations ITR1 and ITR2 in Table 3 show that both recall and precision increase as learning progresses.

Four other experiments were run, to assess the impact of term recognition. Results are shown in Table 3. Configuration TERM continued to label terms in the training phase, but did not label new terms found during iteration (as discussed in Section 3.1).

TOK and NP used no term recognition, instead finding relations between tokens and noun phrases respectively (the gold standard being amended to reflect the new task). POS omitted part-of-speech tags from patterns. In all cases, there was a large increase in spurious results, impacting precision. Term recognition seemed to provide a constraint in relation discovery, although the nature of this is unclear.

6 Conclusions

The PartEx system is capable of fully automated learning of meronyms between semantically typed terms, from the experimental corpus. With simulated pattern pruning, it achieves a recall of 0.73 and a precision of 0.58. In contrast to earlier work, these results were achieved without manual labelling of the corpus, and without direct manual selection of high performance patterns. Although the cost of this automation is lower results than the earlier work, failure analyses provide insights into the algorithm and scope for its further improvement.

Current work includes: automated pattern pruning, extending pattern context and generalisation; incorporating deeper analyses of the text, such as semantic labelling (c.f. Girju (2003)) and the use of dependency structures; investigating the rôle of term recognition in relation discovery; measures for evaluating new relation discovery; extraction of putative sub-relations of meronymy. Work to scale the algorithm to larger corpora is also under way, in recognition of the fact that the corpus used was small, highly regularised, and unusually rich in meronyms.

Acknowledgements

This work was supported by a UK Medical Research Council studentship. The author thanks his supervisor Robert Gaizauskas for useful discussions, and the reviewers for their comments.

References

A. Aronson. 2001. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program. In *Proceedings of the 2001 American Medical Informatics Association Annual Symposium*, pages 17–21, Bethesda, MD.

M. Berland and E. Charniak. 1999. Finding Parts in Very Large Corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 57–64, College Park, MD.

D. Cruse. 2000. *Meaning in Language: An Introduction to Semantics and Pragmatics*. Oxford University Press.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, pages 168–175, Philadelphia, PA.

M. Evens, editor. 1988. *Relational Models of the Lexicon: Representing Knowledge in Semantic Networks*. Cambridge University Press.

R. Girju, A. Badulescu, and D. Moldovan. 2003. Learning Semantic Constraints for the Automatic Discovery of Part-Whole Relations. In *Proceedings of the Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Conference*, Edmonton, Canada.

H. Harkema, R. Gaizauskas, M. Hepple, N. Davis, Y. Guo, A. Roberts, and I. Roberts. 2004. A Large-Scale Resource for Storing and Recognizing Technical Terminology. In *Proceedings of 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

M. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the Fourteenth International Conference on Computational Linguistics*, pages 539–545, Nantes, France.

A. Maedche and S. Staab. 2004. Ontology Learning. In *Handbook on Ontologies*, pages 173–190. Springer.

A. Mikheev and S. Finch. 1997. A Workbench for Finding Structure in Texts. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 372–379, Washington D.C.

E. Morin and C. Jacquemin. 1999. Projecting Corpus-based Semantic Links on a Thesaurus. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 389–396, College Park, MD.

M. Poesio, T. Ishikawa, S. Schulte im Walde, and R. Vieira. 2002. Acquiring Lexical Knowledge for Anaphora Resolution. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, Las Palmas, Canary Islands.

J. Rogers and A. Rector. 2000. GALEN's Model of Parts and Wholes: Experience and Comparisons. In *Proceedings of the 2000 American Medical Informatics Association Annual Symposium*, pages 714–718, Philadelphia, PA.

Using Readers to Identify Lexical Cohesive Structures in Texts

Beata Beigman Klebanov
School of Computer Science and Engineering
The Hebrew University of Jerusalem
Jerusalem, 91904, Israel
beata@cs.huji.ac.il

Abstract

This paper describes a reader-based experiment on lexical cohesion, detailing the task given to readers and the analysis of the experimental data. We conclude with discussion of the usefulness of the data in future research on lexical cohesion.

1 Introduction

The quest for finding what it is that makes an ordered list of linguistic forms into a text that is fluently readable by people dates back at least to Halliday and Hasan's (1976) seminal work on textual cohesion. They identified a number of cohesive constructions: repetition (using the same words, or via repeated reference, substitution and ellipsis), conjunction and lexical cohesion.

Some of those structures - for example, cohesion achieved through repeated reference - have been subjected to reader based tests, often while trying to produce gold standard data for testing computational models, a task requiring sufficient inter-annotator agreement (Hirschman et al., 1998; Mitkov et al., 2000; Poesio and Vieira, 1998).

Experimental investigation of lexical cohesion is an emerging enterprise (Morris and Hirst, 2005) to which the current study contributes. We present our version of the question to the reader to which lexical cohesion patterns are an answer (section 2), describe an experiment on 22 readers using this question (section 3), and analyze the experimental data (section 4).

2 From Lexical Cohesion to Anchoring

Cohesive ties between items in a text draw on the resources of a language to build up the text's unity (Halliday and Hasan, 1976). Lexical cohesive ties draw on the lexicon, i.e. word meanings.

Sometimes the relation between the members of a tie is easy to identify, like near-synonymy (*disease/illness*), complementarity (*boy/girl*), whole-to-part (*box/lid*), but the bulk of lexical cohesive texture is created by relations that are difficult to classify (Morris and Hirst, 2004). Halliday and Hasan (1976) exemplify those with pairs like *dig/garden*, *ill/doctor*, *laugh/joke*, which are reminiscent of the idea of scripts (Schank and Abelson, 1977) or schemata (Rumelhart, 1984): certain things are expected in certain situations, the paradigm example being menu, tables, waiters and food in a restaurant.

However, texts sometimes start with descriptions of situations where many possible scripts could apply. Consider a text starting with *Mother died today.*[1] What are the generated expectations? A description of an accident that led to the death, or of a long illness? A story about what happened to the rest of the family afterwards? Or emotional reaction of the speaker - like the sense of loneliness in the world? Or something more ”technical” - about the funeral, or the will? Or something about the mother's last wish and its fulfillment? Many directions are easily thinkable at this point.

We suggest that rather than generating predictions, scripts/schemata could provide a basis for abduction. Once any ”normal” direction is ac-

[1] the opening sentence of A. Camus' *The Stranger*

Proceedings of the ACL Student Research Workshop, pages 55–60,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

tually taken up by the following text, there is a connection back to whatever makes this a normal direction, according to the reader's commonsense knowledge (possibly coached in terms of scripts or schemata). Thus, had the text developed the illness line, one would have known that it can be best explained-by/blamed-upon/abduced-to the previously mentioned lethal outcome. We say in this case that *illness* is **anchored** by *died*, and mark it *illness*→ *died*; we aim to elicit such anchoring relations from the readers.

3 Experimental Design

We chose 10 texts for the experiment: 3 news articles, 4 items of journalistic writing, and 3 fiction pieces. All news and one fiction story were taken in full; others were cut at a meaningful break to stay within 1000 word limit. The texts were in English - original language for all but two texts.

Our subjects were 22 students at the Hebrew University of Jerusalem, Israel; 19 undergraduates and 3 graduates, all aged 21-29 years, studying various subjects - Engineering, Cognitive Science, Biology, History, Linguistics, Psychology, etc. Three of the participants named English their mother tongue; the rest claimed very high proficiency in English. People were paid for participation.

All participants were first asked to read the guidelines that contained an extensive example of an annotation done by us on a 4-paragraph text (a small extract is shown in table 1), and short paragraphs highlighting various issues, like the possibility of multiple anchors per item (see table 1) and of multiword anchors (*Scientific* or *American* alone do not anchor *editor*, but taken together they do).

In addition, the guidelines stressed the importance of separation between general and personal knowledge, and between general and instantial relations. For the latter case, an example was given of a story about children who went out in a boat with their father who was an experienced sailor, with an explanation that whereas *father*→*children* and *sailor*→*boat* are based on general commonsense knowedge, the connection between *sailor* and *father* is not something general but is created in the particular case because the two descriptions apply to the same person; people were asked not to mark such relations.

Afterwards, the participants performed a trial annotation on a short news story, after which meetings in small groups were held for them to bring up any questions and comments[2].

<table>
<tr><td colspan="2">The Federal Aviation Administration underestimated
the number of aircraft flying over the Pantex Weapons Plant
outside Amarillo, Texas, where much of the nation's surplus
plutonium is stored, according to computerized studies
under way by the Energy Department.</td></tr>
<tr><td>the
federal
aviation
administration→federal
underestimated
number→underestimated
of
aircraft→aviation
flying→{aircraft aviation}
over→flying
pantex
weapons
plant
outside
amarillo
texas→federal</td><td>where→{amarillo texas outside}
much
nation→federal
surplus
plutonium→weapons
is
stored→surplus
according
to
computerized
studies→underestimated
under
way
by
energy→plutonium
department→administration</td></tr>
</table>

Table 1: Example Annotation from the Guidelines (extract). $x \rightarrow \{\ c\ d\ \}$ means each of c and d is an anchor for x.

The experiment then started. For each of the 10 texts, each person was given the text to read, and a separate wordlist on which to write down annotations. The wordlist contained words from the text, in their appearance order, excluding verbatim and inflectional repetitions[3]. People were instructed to read the text first, and then go through the wordlist and ask themselves, for every item on the list, which previously mentioned items help the easy accommodation of this concept into the evolving story, if indeed it is easily accommodated, based on the commonsense knowledge as it is perceived by the annotator. People were encouraged to use a dictionary if they were not sure about some nuance of meaning.

Wordlist length per text ranged from 175 to 339 items; annotation of one text took a person 70 min-

[2] The guidelines and all the correspondence with the participants is archived and can be provided upon request.

[3] The exclusion was done mainly to keep the lists to reasonable length while including as many newly mentioned items as possible. We conjectured that repetitions are usually anchored by the previous mention; this assumption is a simplification, since sometimes the same form is used in a somewhat different sense and may get anchored separately from the previous use of this form. This issue needs further experimental investigation.

utes on average (each annotator was timed on two texts; every text was timed for 2-4 annotators).

4 Analysis of Experimental Data

Most of the existing research in computational linguistics that uses human annotators is within the framework of classification, where an annotator decides, for every test item, on an appropriate tag out of the pre-specified set of tags (Poesio and Vieira, 1998; Webber and Byron, 2004; Hearst, 1997; Marcus et al., 1993).

Although our task is not that of classification, we start from a classification sub-task, and use agreement figures to guide subsequent analysis. We use the by now standard κ statistic (Di Eugenio and Glass, 2004; Carletta, 1996; Marcu et al., 1999; Webber and Byron, 2004) to quantify the degree of above-chance agreement between multiple annotators, and the α statistic for analysis of sources of unreliability (Krippendorff, 1980). The formulas for the two statistics are given in appendix A.

4.1 Classification Sub-Task

Classifying items into anchored/unanchored can be viewed as a sub-task of our experiment: before writing any particular item as an anchor, the annotator asked himself whether the concept at hand is easy to accommodate at all. Getting reliable data on this task is therefore a pre-condition for asking any questions about the anchors. Agreement on this task averages $\kappa = 0.45$ for the 10 texts. These reliability figures do not reach the $\kappa = 0.67$ area which is the accepted threshold for deciding that annotators were working under similar enough internalized theories[4] of the phenomenon; however, the figures are high enough to suggest considerable overlaps.

Seeking more detailed insight into the degree of similarity of the annotators' ideas of the task, we follow the procedure described in (Krippendorff, 1980) to find outliers. We calculate the category-by-category co-markup matrix β for all annotators[5]; then for all but one annotators, and by subtraction find the portion that is due to this one annotator. We then regard the data as two-annotator data (one vs. everybody else), and calculate agreement coefficients. We rank annotators (1 to 22) according to the degree of agreement with the rest, separately for each text, and average over the texts to obtain the *conformity rank* of an annotator. The lower the rank, the less compliant the annotator.

Annotators' conformity ranks cluster into 3 groups described in table 2. The two members of group A are consistent outliers - their average rank for the 10 texts is below 2. The second group (B) is, on average, in the bottom half of the annotators with respect to agreement with the common, whereas members of group C display relatively high conformity.

Gr	Size	Ranks	Agr. within group (κ)
A	2	1.7 - 1.9	0.55
B	9	5.8 - 10.4	0.41
C	11	13.6 - 18.3	0.54

Table 2: Groups of annotators, by conformity ranks.

It is possible that annotators in groups A, B and C have alternative interpretations of the guidelines, but our idea of the ”common” (and thus the conformity ranks) is dominated by the largest group, C. Within-group agreement rates shown in table 2 suggest that two annotators in group A do indeed have an alternative understanding of the task, being much better correlated between each other than with the rest.

The figures for the other two groups could support two scenarios: (1) each group settled on a different theory of the phenomenon, where group C is in better agreement on its version that group B on its own; (2) people in groups B and C have basically the same theory, but members of C are more systematic in carrying it through. It is crucial for our analysis to tell those apart - in the case of multiple stable interpretations it is difficult to talk about **the** anchoring phenomenon; in the core-periphery case, there is hope to identify the core emerging from 20 out of 22 annotations.

Let us call the set of majority opinions on a list of items an *interpretation* of the group, and let us call the average majority percentage *consistency*. Thus, if all decisions of a 9 member group were almost unanimous, the consistency of the group is 8/9 = 89%, whereas if every time there was a one vote

[4] whatever annotators think the phenomenon is after having read the guidelines

[5] See formula 7 in appendix A.

edge to the winning decision, the consistency was 5/9=56%. The more consistent the interpretation given by a group, the higher its agreement coefficient.

If groups B and C have different interpretations, adding a person *p* from group C to group B would usually not improve the consistency of the target group (B), since *p* is likely to represent majority opinion of a group with a different interpretation.

On the other hand, if the two groups settled on basically the same interpretation, the difference in ranks reflects difference in consistency. Then moving *p* from C to B would usually improve the consistency in B, since, coming from a more consistent group, *p*'s agreement with the interpretation is expected to be better than that of an average member of group B, so the addition strengthens the majority opinion in B[6].

We performed this analysis on groups A and C with respect to group B. Adding members of group A to group B improved the agreement in group B only for 1 out of the 10 texts. Thus, the relationship between the two groups seems to be that of different interpretations. Adding members of group C to group B resulted in improvement in agreement in at least 7 out of 10 texts for every added member. Thus, the difference between groups B and C is that of consistency, not of interpretation; we may now search for the well-agreed-upon core of this interpretation. We exclude members of group A from subsequent analysis; the remaining group of 20 annotators exhibits an average agreement of $\kappa = 0.48$ on anchored/unanchored classification.

4.2 Finding the Common Core

The next step is finding a reliably classified subset of the data. We start with the most agreed upon items - those classified as anchored or non-anchored by all the 20 people, then by 19, 18, etc., testing, for every such inclusion, that the chances of taking in instances of chance agreement are small enough. This means performing a statistical hypothesis test: with how much confidence can we reject the hypothesis that certain agreement level[7] is due to chance. Confidence level of $p < 0.01$ is achieved including items marked by at least 13 out of 20 people and items unanimously left unmarked.[8]

The next step is identifying trustworthy anchors for the reliably anchored items. We calculated *average anchor strength* for every text: the number of people who wrote the same anchor for a given item, averaged on all reliably anchored items in a text. Average anchor strength ranges between 5 and 7 in different texts. Taking only strong anchors (anchors of at least the average strength), we retain about 25% of all anchors assigned to anchored items in the reliable subset. In total, there are 1261 pairs of reliably anchored items with their strong anchors, between 54 and 205 per text.

Strength cut-off is a heuristic procedure; some of those anchors were marked by as few as 6 or 7 out of 20 people, so it is not clear whether they can be trusted as embodiments of the core of the anchoring phenomenon in the analyzed texts. Consequently, an anchor validation procedure is needed.

4.3 Validating the Common Core

We observe that although people were asked to mark all anchors for every item they thought was anchored, they actually produced only 1.86 anchors per anchored item. Thus, people were most concerned with finding **an** anchor, i.e. making sure that something they think is easily accommodatable is given at least one preceding item to blame for that; they were less diligent in marking up all such items. This is also understandable processing-wise; after a scrupulous read of the text, coming up with one or two anchors can be done from memory, only occasionally going back to the text; putting down all anchors would require systematic scanning of the previous stretch of text for every item on the list; the latter task is hardly doable in 70 minutes.

[6] Experiments with synthetic data confirm this analysis: with 20 annotations split into 2 sets of sizes 9 and 11, it is possible to get an overall agreement of about $\kappa = 0.40$ either with 75% and 90% consistency on the same interpretation, or with 90% and 95% consistency on two interpretations with induced (i.e. non-random) overlap of just 20%.

[7] A random variable ranging between 0 and 20 says how many "random" people marked an item as anchored. We model "random" versions of annotators by taking the proportions p_i of items marked as anchored by annotator i in the whole of the dataset, and assuming that for every word, the person was tossing a coin with P(heads) = p_i, independently for every word.

[8] Confidence level of $p < 0.03$ allows augmenting the set of reliably unanchored items with those marked by 1 or 2 people, retaining the same cutoff for anchoredness. This cut covers more than 60% of the data, and contains 1504 items, 538 of which are anchored.

Having in mind the difficulty of producing an exhaustive list of anchors for every item, we conducted a follow-up experiment to see whether people would accept anchors when those are presented to them, as opposed to generating ones. We used 6 out of the 10 texts and 17 out of 20 annotators for the follow-up experiment. Each person did 3 text, each texts received 7-9 annotations of this kind.

For each text, the reader was presented with the same list of words as in the first part, only now each word was accompanied by a list of anchors. For each item, every anchor generated by at least one person was included; the order of the anchors had no correspondence with the number of people who generated it. A small number of items also received a random anchor – a randomly chosen word from the preceding part of the wordlist. The task was crossing over anchors that the person does not agree with.

Ideally, i.e. if lack of markup is merely a difference in attention but not in judgment, all non-random anchors should be accepted. To see the distance of the actual results from this scenario, we calculate the total mass of votes as number of anchored-anchor pairs times number of people, and check how many are accept votes. For all non-random pairs, 62% were accept votes; for the core annotations (pairs of reliably anchored items with strong anchors) 94% were accept votes, texts ranging between 90% and 96%; for pairs with a random anchor, only 15% were accept votes. Thus, agreement based analysis of anchor generation data allowed us to identify a highly valid portion of the annotations.

5 Conclusion

This paper presented a reader-based experiment on finding lexical cohesive patterns in texts. As it often happens with tasks related to semantics/pragmatics (Poesio and Vieira, 1998; Morris and Hirst, 2005), the inter-reader agreement levels did not reach the accepted reliability thresholds. We showed, however, that statistical analysis of the data, in conjunction with a subsequent validation experiment, allow identification of a reliably annotated core of the phenomenon.

The core data may now be used in various ways. First, it can seed psycholinguistic experimentation of lexical cohesion: are anchored items processed quicker than unanchored ones? When asked to recall the content of a text, would people remember prolific anchors of this text? Such experiments will further our understanding of the nature of text-reader interaction and help improve applications like text generation and summarization.

Second, it can serve as a minimal test data for computational models of lexical cohesion: any good model should at least get the core part right. Much of the existing applied research on lexical cohesion uses WordNet-based (Miller, 1990) lexical chains to identify the cohesive texture for a larger text processing application (Barzilay and Elhadad, 1997; Stokes et al., 2004; Moldovan and Novischi, 2002; Al-Halimi and Kazman, 1998). We can now subject these putative chains to a direct test; in fact, this is the immediate future research direction.

In addition, analysis techniques discussed in the paper – separating interpretation disagreement from difference in consistency, using statistical hypothesis testing to find reliable parts of the annotations and validating them experimentally – may be applied to data resulting from other kinds of exploratory experiments to gain insights about the phenomena at hand.

Acknowledgment

I would like to thank Prof. Eli Shamir for guidance and numerous discussions.

References

Reem Al-Halimi and Rick Kazman. 1998. Temporal indexing through lexical chaining. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 333–351. MIT Press, Cambridge, MA.

Regina Barzilay and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL Intelligent Scalable Text Summarization Workshop*, pages 86–90.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254.

Barbara Di Eugenio and Michael Glass. 2004. The kappa statistic: a second look. *Computational Linguistics*, 30(1):95–101.

M.A.K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman Group Ltd.

Marti Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Lynette Hirschman, Patricia Robinson, John D. Burger, and Marc Vilain. 1998. Automating coreference: The role of annotated training data. *CoRR*, cmp-lg/9803001.

Klaus Krippendorff. 1980. *Content Analysis*. Sage Publications.

Daniel Marcu, Estibaliz Amorrortu, and Magdalena Romera. 1999. Experiments in constructing a corpus of discourse trees. In *Proceedings of ACL'99 Workshop on Standards and Tools for Discourse Tagging*, pages 48–57.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313 – 330.

G. Miller. 1990. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–312.

Ruslan Mitkov, Richard Evans, Constantin Orasan, Catalina Barbu, Lisa Jones, and Violeta Sotirova. 2000. Coreference and anaphora: developing annotating tools, annotated resources and annotation strategies. In *Proceedings of the Discourse Anaphora and Anaphora Resolution Colloquium (DAARC'2000)*, pages 49–58.

Dan Moldovan and Adrian Novischi. 2002. Lexical chains for question answering. In *Proceedings of COLING 2002*.

Jane Morris and Graeme Hirst. 2004. Non-classical lexical semantic relations. In *Proceedings of HLT-NAACL Workshop on Computational Lexical Semantics*.

Jane Morris and Graeme Hirst. 2005. The subjectivity of lexical cohesion in text. In James C. Chanahan, Yan Qu, and Janyce Wiebe, editors, *Computing attitude and affect in text*. Springer, Dodrecht, The Netherlands.

Massimo Poesio and Renata Vieira. 1998. A corpus-based investigation of definite description use. *Computational Linguistics*, 24(2):183–216.

David E. Rumelhart. 1984. Understanding understanding. In J. Flood, editor, *Understanding Reading Comprehension*, pages 1–20. Delaware: International Reading Association.

Roger Schank and Robert Abelson. 1977. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Hillsdale, NJ: Lawrence Erlbaum.

Sidney Siegel and John N. Castellan. 1988. *Nonparametric statistics for the behavioral sciences*. McGraw Hill, Boston, MA.

Nicola Stokes, Joe Carthy, and Alan F. Smeaton. 2004. Select: A lexical cohesion based news story segmentation system. *Journal of AI Communications*, 17(1):3–12.

Bonny Webber and Donna Byron, editors. 2004. *Proceedings of the ACL-2004 Workshop on Discourse Annotation*, Barcelona, Spain, July.

A Measures of Agreement

Let N be the number of items to be classified; m - the number of categories to classify into; k - the number of raters; n_{ij} is the number of annotators who assigned the i-th item to j-th category. We use Siegel and Castellan's (1988) version of κ; although it assumes similar distributions of categories across coders in that it uses the average to estimate the expected agreement (see equation 2), the current experiment employs 22 coders, so averaging is a much better justified enterprise than in studies with very few coders (2-4), typical in discourse annotation work (Di Eugenio and Glass, 2004). The calculation of the α statistic follows (Krippendorff, 1980).

The κ Statistic

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)} \tag{1}$$

$$P(E) = \sum_{j=1}^{m} p_j^2, \quad p_j = \frac{\sum_{i=1}^{N} n_{ij}}{Nk} \tag{2}$$

$$P(A) = \frac{1}{Nk(k-1)} \sum_{i=1}^{N} \sum_{j=1}^{m} n_{ij}(n_{ij} - 1) \tag{3}$$

The α Statistic

$$\alpha = 1 - \frac{D_{obs}}{D_{exp}} \tag{4}$$

$$D_{obs} = \frac{1}{\zeta} \sum_{j=1}^{m} \sum_{i=1}^{m} \beta_{ji}(1 - \Delta_{ji}) \tag{5}$$

$$D_{exp} = \frac{1}{\zeta(\zeta - k + 1)} \sum_{j=1}^{m} \sum_{i=1}^{m} \gamma_j \gamma_i (1 - \Delta_{ji}) \tag{6}$$

$$\beta_{ji} = \sum_{l=1}^{N} n_{lj}(n_{li} - \Delta_{ji}), \quad \zeta = mk(k-1) \tag{7}$$

$$\Delta_{ji} = \left\{ \begin{array}{ll} 0 & j \neq i \\ 1 & j = i \end{array} \right. , \quad \gamma_i = \sum_{j=1}^{m} \beta_{ji} \tag{8}$$

Towards an Optimal Lexicalization in a Natural-Sounding Portable Natural Language Generator for Dialog Systems

Inge M. R. De Bleecker
Department of Linguistics
The University of Texas at Austin
Austin, TX 78712, USA
imrdb@mail.utexas.edu

Abstract

In contrast to the latest progress in speech recognition, the state-of-the-art in natural language generation for spoken language dialog systems is lagging behind. The core dialog managers are now more sophisticated; and natural-sounding and flexible output is expected, but not achieved with current simple techniques such as template-based systems. Portability of systems across subject domains and languages is another increasingly important requirement in dialog systems. This paper presents an outline of LEGEND, a system that is both portable and generates natural-sounding output. This goal is achieved through the novel use of existing lexical resources such as FrameNet and WordNet.

1 Introduction

Most of the natural language generation (NLG) components in current dialog systems are implemented through the use of simple techniques such as a library of hand-crafted and pre-recorded utterances, or a template-based system where the templates contain slots in which different values can be inserted. These techniques are unmanageable if the dialog system aims to provide variable, natural-sounding output, because the number of pre-recorded strings or different templates becomes very large (Theune, 2003). These techniques also make it difficult to port the system into another subject domain or language.

In order to be widely successful, natural language generation components of future dialog systems need to provide natural-sounding output while being relatively easy to port. This can be achieved by developing more sophisticated techniques based on concepts from deep linguistically-based NLG and text generation, and through the use of existing resources that facilitate both the natural-sounding and the portability requirement.

We might wonder what exactly it means for a computer to generate ‘natural-sounding’ output. Computer-generated natural-sounding output should not mimic the output a human would construct, because spontaneous human dialog tends to be teeming with disfluencies, interruptions, syntactically incorrect and incomplete sentences among others (Zue, 1997). Furthermore, Oberlander (1998) points out that humans do not always take the most efficient route in their reasoning and communication. These observations lead us to define natural-sounding computer-generated output to consist of utterances that are free of disfluencies and interruptions, and where complete and syntactically correct sentences convey the meaning in a concise yet clear manner.

Secondly we can define the portability requirement to include both domain and language independence. Domain-independence suggests that the system must be easily portable between different domains, while language-independence requires that the system must be able to accommodate a new natural language without any changes to the core components.

Section 2 of this paper explains some prerequisites, such as the NLG pipeline architecture our system is based on, and the FrameNet and WordNet resources. Next an overview of the system ar-

Proceedings of the ACL Student Research Workshop, pages 61–66,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

chitecture and implementation, as well as an in-depth analysis of the lexicalization component are presented. Section 3 presents related work. Section 4 outlines a preliminary conclusion and lists some outstanding issues.

2 System Architecture

2.1 Three-Stage Pipeline Architecture

Our natural language generator architecture follows the three-stage pipeline architecture, as described in Reiter & Dale (2000). In this architecture, the generation component of a text generation system consists of the following subcomponents:

- The **document planner** determines what the actual content of the output will be on an abstract level and decides how pieces of content should be grouped together.
- The **microplanner** includes lexicalization, aggregation, and referring expression generation tasks.
- The **surface realizer** takes the information constructed by the microplanner and generates a syntactically correct sentence in a natural language.

2.2 Lexical Resources

The use of FrameNet and WordNet in our system is critical to its success. The FrameNet database (Baker et al., 1998) is a machine-readable lexicographic database which can be found at http://framenet.icsi.berkeley.edu/. It is based on the principles of Frame Semantics (Fillmore, 1985). The following quote explains the idea behind Frame Semantics: "The central idea of Frame Semantics is that word meanings must be described in relation to *semantic frames* – schematic representations of the conceptual structures and patterns of beliefs, practices, institutions, images, etc. that provide a foundation for meaningful interaction in a given speech community." (Fillmore et al., 2003, p. 235). In FrameNet, lexical units are grouped in frames; frame hierarchy information is provided for each frame, in combination with a list of semantically annotated corpus sentences and syntactic valence patterns. WordNet is a lexical database that uses conceptual-semantic and lexical relations in order to group lexical items and link them to other groups (Fellbaum, 1998).

2.3 System Overview

Our system, called LEGEND (LExicalization in natural language GENeration for Dialog systems) adapts the pipeline architecture presented in section 2.1 by replacing the document planner with the dialog manager. This makes it more suitable for use in dialog systems, since the dialog manager decides on the actual content of the output in dialog systems. Figure 1 below shows an overview of our system architecture.

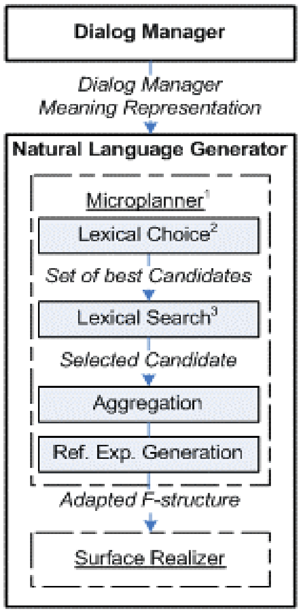


Figure 1. System Architecture

As figure 1 shows, the dialog manager provides the generator with a dialog manager meaning representation (DM MR), which contains the content information for the answer.

Our research focuses on the lexicalization subcomponent of the microplanner (number 1 in figure 1). Lexicalization is further divided into two processes: lexical choice and lexical search. Based on the DM MR, the lexical choice process (number 2 in figure 1) constructs a set of all potential output candidates. Section 2.5 describes the lexical choice process in detail. Lexical search (number 3 in figure 1) consists of the decision algorithm that decides which one of the set of possible candidates is most appropriate in any situation. Lexical search is also responsible for packaging up the most appropriate candidate information in an adapted F-structure, which is subsequently processed through aggregation and referring expression generation, and finally sent to the surface realizer. Section 2.6 describes the details of the lexical search process.

2.4 Implementation Details

Given time and resource constraints, our implementation will consist of a prototype (written in Python) of the lexical choice and lexical search processes only of the microplanner. We take a DM MR as our input. Aggregation and referring expression generation requirements are hard-coded for each example; algorithm development, identification and implementation for these modules is beyond the scope of this research.

Our system uses the LFG-based XLE system's generator component as a surface realizer. For more information, refer to Shemtov (1997) and Kaplan & Wedekind (2000).

2.5 Lexical Choice

The task of the lexical choice process is to take the meaning representation presented by the dialog manager (refer to figure 1), and to construct a set of output candidates. We will illustrate this by taking a simple example through the entire dialog system. The example question and answer are deliberately kept simple in order to focus on the workings of the system, rather than the specifics of the example.

Assume this is a dialog system that helps the consumer in buying camping equipment. The user says to the dialog system: "Where can I buy a tent?" The speech recognizer recognizes the utterance, and feeds this information to the parser. The semantic parser parses the input and builds the meaning representation shown in figure 2. The main event (main verb) is identified as the lexical item *buy*. The parser looks up this lexical item in FrameNet, and identifies it as belonging to the *commerce_buy* frame. This frame is defined in FrameNet as: "… describing a basic commercial transaction involving a buyer and a seller exchanging money and goods, taking the perspective of the buyer." (http://framenet.icsi.berkeley.edu/). All other elements in the meaning representation are extracted from the input utterance.

Event: buy
Frame: commerce_buy
Query: location
Agent: 1st pers sing
Patient: tent

Figure 2. Parser Meaning Representation

This meaning representation is then sent to the dialog manager. The dialog manager consults the domain model for help in the query resolution, and subsequently composes a meaning representation consisting of the answer to the user's question (figure 3). For our example, the domain model presents the query resolution as "Camping World", the name of a (fictitious) store selling tents. The DM MR also shows that the *Agent* and the *Patient* have been identified by their frame element names. This DM MR serves as the input to the microplanner, where the first task is that of lexical choice.

Event: buy
Frame: commerce_buy
Query Resolution: place "Camping World"
Agent: buyer (1st p.s. => 2nd p.s.)
Object: goods ("tent")

Figure 3. Dialog Mgr Meaning Representation

In order to construct the set of output candidates, the lexical choice process mines the FrameNet and WordNet databases in order to find acceptable generation possibilities. This is done in several steps:

- In step 1, lexicalization variations of the main *Event* within the same frame are identified.
- Step 2 consists of the investigation of lexical variation in the frames that are one link away in the hierarchy, namely the frame the current frame inherits from, and the subframes, if any exist.
- Step 3 is concerned with special relations within FrameNet, such as the 'use'-relation The lexical variation within these frames is investigated.

We return to our example in figure 3 to clarify these 3 steps.

In step 1, appropriate lexical variation within the same frame is identified. This is done by listing all lexical units of same syntactic category as the original word. The following verbs are lexical units in *commerce_buy*: *buy, lease, purchase, rent.* These verbs are not necessarily synonyms or near-synonyms of each other, but do belong to the same frame. In order to determine which of these lexical items are synonyms or near-synonyms, we turn to WordNet, and look at the entry for *buy*. The only lexical item that is also listed in one of the senses of *buy* is *purchase*. We thus conclude that *buy* and *purchase* are both good verb candidates.

Step 2 investigates the lexical items in the frames that are one link away from the *commerce_buy* frame. *Commerce_buy* inherits from *getting*, and has no subframes. The lexical items of the *getting* frame are listed. The lexical items of the *getting* frame are: *acquire, gain, get, obtain, secure.* For each entry, WordNet is consulted as a first pruning mechanism. This results in the following:

- Acquire: get
- Gain: acquire, win
- Get: acquire
- Obtain: get, find, receive, incur
- Secure: no items on the list

How exactly lexical choice determines that *get* and *acquire* are possible candidates, while the others are not (because they aren't suitable in the context in which we use them) is as of yet an open issue. It is also an open issue whether WordNet is the most appropriate resource to use for this goal; we must consider other options, such as Thesaurus, etc…

In step 3 we investigate the other relations that FrameNet presents. To date, we have only investigated the 'use relation'. Other relations available are the inchoative and causative relations. At this point, it is not entirely clear how those relations will prove to be of any value to our task. The *commerce_buy* frame uses *commerce_goods_transfer*, which is also used by *commerce_sell.* We find our frame elements *goods* and *buyer* in the *commerce_sell* frame as well. Lexical choice concludes that the use of the lexical items in this frame might be valuable and repeats step 1 on these lexical items.

After all 3 steps are completed, we assume our set of output candidates to be complete. The set of output candidates is presented to the lexical search process, whose task it is to choose the most appropriate candidate. For the example we have been using throughout this section, the set of output candidates is as follows:

- You can buy a tent at Camping World.
- You can purchase a tent at Camping World.
- You can get a tent at Camping World.
- You can acquire a tent at Camping World.
- Camping World sells tents.

As mentioned at the beginning of this section, this example is very simple. For this reason, one can definitely argue that the first 4 output possibilities could be constructed in much simpler ways than the method used here, e.g. by simply taking the question and making it an affirmative sentence through a simple rule. However, it should be pointed out that the last possibility on the list would not be covered by this simple method. While user studies would need to provide backup for this assumption, we feel that possibility 5 is a very good example of natural-sounding output, and thus proves our method to be valuable, even for simple examples.

2.6 Lexical Search

The set of output candidates for the example above contains 5 possibilities. The main task of the lexical search process is to choose the most optimal candidate, thus the most natural-sounding candidate (or at least one of the most natural-sounding candidates, if more than one candidate fits that criterion). There are a number of directions we can take for this implementation.

One option is to implement a rule-based system. Every output candidate is matched against the rules, and the most appropriate one comes out at the top. Problems with rule-based systems are well-known: they must be handcrafted, which is very time-consuming, constructing the rule base such that the desired rules fire in the desired circumstances is somewhat of a "black" art, and of course a rule base is highly domain-dependent. Extending and maintaining it is also a laborious effort.

Next we can look at a corpus-based technique. One suggestion is to construct a language model of the corpus data, and use this model to statistically determine the most suitable candidate. Langkilde (2000) uses this approach. However, the main problem here is that one needs a large corpus in the domain of the application. Rambow (2001) agrees that most often, no suitable corpora are available for dialog system development.

Another possibility is to use machine learning to train the microplanner. Walker et al. (2002) use this approach in the SPOT sentence planner. Their ranker's main purpose is to choose between different aggregation possibilities. The authors suggest that many generation problems can successfully be treated as ranking problems. The advantage of this approach is that no domain-dependent hand-crafted rules need to be constructed, and no existence of a corpus is needed.

Our current research idea is somewhat related to option two. A relatively small domain-independent corpus of spoken dialogue is semi-automatically labeled with frames and semantic roles. For each frame, all the occurrences in the corpus are ordered according to their frequency for each separate valence pattern. This model is then used as a comparator for all output candidates, and the most optimal one (most frequent one) will be selected. This approach is currently not implemented; further work needs to determine the viability of the approach.

Independent of the method used to find the most suitable candidate, the output must be packaged up to be sent to the surface realizer. The XLE system expects a fairly detailed syntactic description of the utterance's argument structure. We construct this through the use of FrameNet and its valence pattern information. In returning to our example, let's assume the selected candidate is "Camping World sells tents." Its meaning representation is as follows:

Event: sell
Frame: commerce_sell
Seller: Camping World
Goods: tents

Figure 4. "Camping World sells tents."

FrameNet provides an overview of the frame elements a given frame requires ("core elements") and those that are optional ("peripheral elements"). For the *commerce_sell* frame, the two core elements are *Goods* and *Seller*. It also provides an overview of the valence patterns that were found in the annotated sentences for this frame. FrameNet does not include frequency information for each annotation. We thus need to pick a valence pattern at random. One way of doing this is to find a pattern that includes all (both) frame elements in our utterance, and then use the (non-statistical) frequency information. Figure 5 shows that, for our example above, this results in:

FE_Seller *sell* FE_goods
With the following syntactic pattern:
NP.Ext *sell* NP.Obj

No. Annotated	Patterns	
	Goods	Seller
3	--	NP.Ext
2	NP.Comp	NP.Ext
27	NP	--
4	NP.Ext	PP[by].Comp
27	NP.Obj	NP.Ext

Figure 5. Valence Patterns "commerce_sell"

Thus our output to the surface realizer indicates that the seller frame element fills the subject role and consists of an NP, while the goods frame element fills the object role and consists of an NP. Given this syntactic pattern information that we gather from FrameNet, we are able to construct an F-structure that is suitable as the input to the surface realizer.

3 Related Work

To date, only a limited amount of research has dealt with deep linguistically-based natural language generation for dialog systems. Theune (2003) presents an extensive overview of different NLG methods and systems. A number of stochastic-based generation efforts have been undertaken in recent years. These generators generally consist of an architecture similar to ours, in which first a set of possible candidates is constructed, followed by a decision process to choose the most appropriate output. Some examples are the Nitrogen system (Langkilde and Knight, 1998) and the SPoT trainable sentence planner (Walker et al., 2002).

4 Outlook and Future Work

We propose a novel approach to lexicalization in NLG in order to generate natural-sounding speech in a portable environment. The use of existing lexical resources allows a system to be more portable across subject domains and languages, as long as those resources are available for the targeted domains and languages. FrameNet in particular allows us to generate multiple possibilities of natural-sounding output while WordNet helps in a first step to prune this set. FrameNet is further applied on an existing corpus to help with the final decision on choosing the most optimal candidate among the presented possibilities. The valence pattern information in FrameNet helps constructing the detailed syntactic pattern required by the surface realizer.

A number of issues need further consideration, including the following:

- lexical choice: investigation of semantic distances (step 2 of algorithm), use of WordNet and/or other resources for first-step pruning.
- lexical search: develop initial research ideas further and implement
- a user study to assess whether the goals of natural-sounding output and portability have successfully been fulfilled.

Furthermore, for this generator to be used in a real-life environment, the entire dialog system must be developed; for our research purposes, we have left out the construction of a semantic parser, the dialog manager, and an appropriate domain model. We have also not focused on the development of the aggregation and referring expression generation subtasks in the microplanner.

References

Baker, Collin F. and Charles J. Fillmore and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the COLING-ACL,* Montreal, Canada.

Dale, Robert and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science* 18:233-263.

Fellbaum, Christiane. 1998. A Semantic Network of English: The Mother of All WordNets. In *Computers and the Humanities,* Kluwer, The Netherlands, 32: 209-220.

Fillmore, Charles J. and Christopher R. Johnson and Miriam R.L. Petruck. 2003. Background to FrameNet. In *International Journal of Lexicography.* Vol. 16 No. 3. 2003. Oxford University Press. Oxford, UK.

Fillmore, Charles J. 1985. Frames and the semantics of understanding. In *Quaderni di Semantica*, Vol. 6.2: 222-254.

Oberlander, Jon. 1998. Do the Right Thing… but Expect the Unexpected. *Computational Linguistics.* Volume 24, Number 3. September 1998, pp. 501-507. The MIT Press, Cambridge, MA.

Shemtov, Hadar. 1997. Ambiguity Management in Natural Language Generation, PhD Thesis, Stanford.

Kaplan, R. M. and J. Wedekind. 2000. LFG generation produces context-free languages. In *Proceedings of COLING-2000,* Saarbruecken, pp. 297-302.

Langkilde, Irene. 2000. Forest-based Statistical Sentence Generation. In Proceedings of the *North American Meeting of the Association for Computational Linguistics (NAACL),* 2000.

Langkilde, Irene and Kevin Knight. 1998. Generation that Exploits Corpus-Based Statistical Knowledge. In *Proceedings of Coling-ACL 1998.* Montréal, Canada.

Rambow, Owen, 2001. Corpus-based Methods in Natural Language Generation: Friend or Foe? Invited talk at the *European Workshop for Natural Language Generation*, Toulouse, France.

Reiter, Ehud and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press. Cambridge, UK.

Theune, Mariët. 2000. From data to speech: language generation in context. Ph.D. thesis, Eindhoven University of Technology.

Theune, Mariët. 2003. Natural Language Generation for Dialogue: System Survey. University of Twente. Twente, the Netherlands.

Walker, Marilyn and Owen Rambow and Monica Rogati. 2002. Training a Sentence Planner for Spoken Dialogue Using Boosting. *Computer Speech and Language, Special Issue on Spoken Language Generation*, July 2002.

Zue, Victor. 1997. Conversational Interfaces: Advances and Challenges. Keynote in *Proceedings of Eurospeech 1997.* Rhodes, Greece.

Phrase Linguistic Classification and Generalization for Improving Statistical Machine Translation

Adrià de Gispert
TALP Research Center
Universitat Politècnica de Catalunya (UPC)
Barcelona
agispert@gps.tsc.upc.es

Abstract

In this paper a method to incorporate linguistic information regarding single-word and compound verbs is proposed, as a first step towards an SMT model based on linguistically-classified phrases. By substituting these verb structures by the base form of the head verb, we achieve a better statistical word alignment performance, and are able to better estimate the translation model and generalize to unseen verb forms during translation. Preliminary experiments for the English - Spanish language pair are performed, and future research lines are detailed.

1 Introduction

Since its revival in the beginning of the 1990s, statistical machine translation (SMT) has shown promising results in several evaluation campaigns. From original word-based models, results were further improved by the appearance of phrase-based translation models.

However, many SMT systems still ignore any morphological analysis and work at the surface level of word forms. For highly-inflected languages, such as German or Spanish (or any language of the Romance family) this poses severe limitations both in training from parallel corpora, as well as in producing a correct translation of an input sentence.

This lack of linguistic knowledge in SMT forces the translation model to learn different translation probability distributions for all inflected forms of nouns, adjectives or verbs (’vengo’, ’vienes’, ’viene’, etc.), and this suffers from usual data sparseness. Despite the recent efforts in the community to provide models with this kind of information (see Section 6 for details on related previous work), results are yet to be encouraging.

In this paper we address the incorporation of morphological and shallow syntactic information regarding verbs and compound verbs, as a first step towards an SMT model based on linguistically-classified phrases. With the use of POS-tags and lemmas, we detect verb structures (with or without personal pronoun, single-word or compound with auxiliaries) and substitute them by the base form[1] of the head verb. This leads to an improved statistical word alignment performance, and has the advantages of improving the translation model and generalizing to unseen verb forms, during translation. Experiments for the English - Spanish language pair are performed.

The organization of the paper is as follows. Section 2 describes the rationale of this classification strategy, discussing the advantages and difficulties of such an approach. Section 3 gives details of the implementation for verbs and compound verbs, whereas section 4 shows the experimental setting used to evaluate the quality of the alignments. Section 5 explains the current point of our research, as well as both our most-immediate to-do tasks and our medium and long-term experimentation lines. Finally, sections 6 and 7 discuss related works that can be found in literature and conclude, respectively.

[1] The terms ’base form’ or ’lemma’ will be used equivalently in this text.

Proceedings of the ACL Student Research Workshop, pages 67–72,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

2 Morphosyntactic classification of translation units

State-of-the-art SMT systems use a log-linear combination of models to decide the best-scoring target sentence given a source sentence. Among these models, the basic ones are a translation model $Pr(e|f)$ and a target language model $Pr(e)$, which can be complemented by reordering models (if the language pairs presents very long alignments in training), word penalty to avoid favoring short sentences, class-based target-language models, etc (Och and Ney, 2004).

The translation model is based on phrases; we have a table of the probabilities of translating a certain source phrase $\tilde{f}_j$ into a certain target phrase $\tilde{e}_k$. Several strategies to compute these probabilities have been proposed (Zens et al., 2004; Crego et al., 2004), but none of them takes into account the fact that, when it comes to translation, many different inflected forms of words share the same translation. Furthermore, they try to model the probability of translating certain phrases that contain just auxiliary words that are not directly relevant in translation, but play a secondary role. These words are a consequence of the syntax of each language, and should be dealt with accordingly.

For examples, consider the probability of translating 'in the' into a phrase in Spanish, which does not make much sense in isolation (without knowing the following meaning-bearing noun), or the modal verb 'will', when Spanish future verb forms are written without any auxiliary.

Given these two problems, we propose a classification scheme based on the base form of the phrase head, which is explained next.

2.1 Translation with classified phrases

Assuming we translate from f to e, and defining $\tilde{e}_i$, $\tilde{f}_j$ a certain source phrase and a target phrases (sequences of contiguous words), the phrase translation model $Pr(\tilde{e}_i|\tilde{f}_j)$ can be decomposed as:

$$\sum_T Pr(\tilde{e}_i|T,\tilde{f}_j)Pr(\tilde{E}_i|\tilde{F}_j,\tilde{f}_j)Pr(\tilde{F}_j,\tilde{f}_j) \qquad (1)$$

where $\tilde{E}_i$, $\tilde{F}_j$ are the generalized classes of the source and target phrases, respectively, and $T = (\tilde{E}_i,\tilde{F}_j)$ is the pair of source and target classes used, which we call Tuple. In our current implementation, we consider a classification of phrases that is:

- *Linguistic*, ie. based on linguistic knowledge
- *Unambiguous*, ie. given a source phrase there is only one class (if any)
- *Incomplete*, ie. not all phrases are classified, but only the ones we are interested in
- *Monolingual*, ie. it runs for every language independently

The second condition implies $Pr(\tilde{F}|\tilde{f}) = 1$, leading to the following expression:

$$Pr(\tilde{e}_i|\tilde{f}_j) = Pr(\tilde{E}_i|\tilde{F}_j)Pr(\tilde{e}_i|T,\tilde{f}_j) \qquad (2)$$

where we have just two terms, namely a standard phrase translation model based on the classified parallel data, and an instance model assigning a probability to each target instance given the source class and the source instance. The latter helps us choose among target words in combination with the language model.

2.2 Advantages

This strategy has three advantages:

Better alignment. By reducing the number of words to be considered during first word alignment (auxiliary words in the classes disappear and no inflected forms used), we lessen the data sparseness problem and can obtain a better word alignment. In a secondary step, one can learn word alignment relationships inside aligned classes by realigning them as a separate corpus, if that is desired.

Improvement of translation probabilities. By considering many different phrases as different instances of a single phrase class, we reduce the size of our phrase-based (now class-based) translation model and increase the number of occurrences of each unit, producing a model $Pr(\tilde{E}|\tilde{F})$ with less perplexity.

Generalizing power. Phrases not occurring in the training data can still be classified into a class, and therefore be assigned a probability in the translation model. The new difficulty that rises is how to produce the target phrase from the target class and the source phrase, if this was not seen in training.

2.3 Difficulties

Two main difficulties[2] are associated with this strategy, which will hopefully lead to improved translation performance if tackled conveniently.

Instance probability. On the one hand, when a phrase of the test sentence is classified to a class, and then translated, how do we produce the instance of the target class given the tuple T and the source instance? This problem is mathematically expressed by the need to model the term of the $Pr(\tilde{e}_i|T, \tilde{f}_j)$ in Equation 2.

At the moment, we learn this model from relative frequency across all tuples that share the same source phrase, dividing the times we see the pair $(\tilde{f}_j, \tilde{e}_i)$ in the training by the times we see $\tilde{f}_j$.

Unseen instances. To produce a target instance $\tilde{f}$ given the tuple T and an unseen $\tilde{e}$, our idea is to combine both the information of verb forms seen in training *and* off-the-shelf knowledge for generation. A translation memory can be built with all the seen pairs of instances with their inflectional affixes separated from base forms.

For example, suppose we translate from English to Spanish and see the tuple T=(V[go],V[ir]) in training, with the following instances:

I will go PRP(1S) will VB	iré VB 1S F
you will go PRP(2S) will VB	irás VB 2S F
you will go PRP(2S) will VB	vas VB 2S P

where the second row is the analyzed form in terms of person (1S: 1st singular, 2S: 2nd singular and so on) and tense (VB: infinitive and P: present, F: future). From these we can build a generalized rule independent of the person ’ PRP(X) will VB ’ that would enable us to translate ’we will go’ to two different alternatives (present and future form):

we will go	VB 1P F
we will go	VB 1P P

These alternatives can be weighted according to the times we have seen each case in training. An unambiguous form generator produces the forms ’iremos’ and ’vamos’ for the two Spanish translations.

3 Classifying Verb Forms

As mentioned above, our first and basic implementation deals with verbs, which are classified unambiguously before alignment in training and before translating a test.

3.1 Rules used

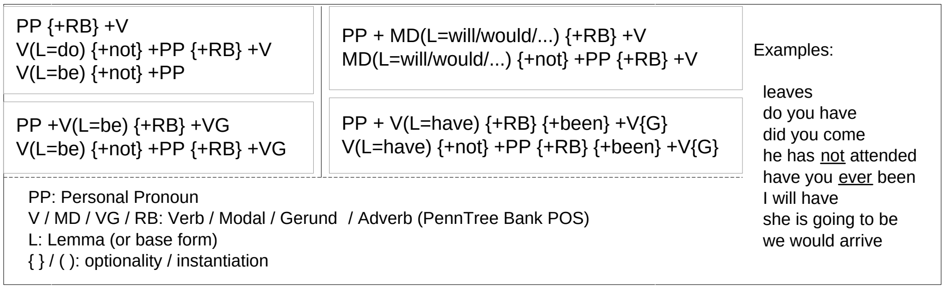
We perform a knowledge-based detection of verbs using deterministic automata that implement a few simple rules based on word forms, POS-tags and word lemmas, and map the resulting expression to the lemma of the head verb (see Figure 1 for some rules and examples of detected verbs). This is done both in the English and the Spanish side, and before word alignment.

Note that we detect verbs containing adverbs and negations (underlined in Figure 1), which are ordered before the verb to improve word alignment with Spanish, but once aligned they are reordered back to their original position *inside* the detected verb, representing the real instance of this verb.

4 Experiments

In this section we present experiments with the Spanish-English parallel corpus developed in the framework of the LC-STAR project. This corpus consists of transcriptions of spontaneously spoken dialogues in the tourist information, appointment scheduling and travel planning domain. Therefore, sentences often lack correct syntactic structure. Preprocessing includes:

[2] A third difficulty is the classification task itself, but we take it for granted that this is performed by an independent system based on other knowledge sources, and therefore out of scope here.



PP {+RB} +V
V(L=do) {+not} +PP {+RB} +V
V(L=be) {+not} +PP

PP +V(L=be) {+RB} +VG
V(L=be) {+not} +PP {+RB} +VG

PP + MD(L=will/would/...) {+RB} +V
MD(L=will/would/...) {+not} +PP {+RB} +V

PP + V(L=have) {+RB} {+been} +V{G}
V(L=have) {+not} +PP {+RB} {+been} +V{G}

PP: Personal Pronoun
V / MD / VG / RB: Verb / Modal / Gerund / Adverb (PennTree Bank POS)
L: Lemma (or base form)
{ } / ( ): optionality / instantiation

Examples:

leaves
do you have
did you come
he has not attended
have you ever been
I will have
she is going to be
we would arrive

Figure 1: *Some verb phrase detection rules and detected forms in English.*

- Normalization of contracted forms for English (ie. wouldn't = would not, we've = we have)
- English POS-tagging using freely-available *TnT* tagger (Brants, 2000), and lemmatization using *wnmorph*, included in the WordNet package (Miller et al., 1991).
- Spanish POS-tagging using *FreeLing* analysis tool (Carreras et al., 2004). This software also generates a lemma or base form for each input word.

4.1 Parallel corpus statistics

Table 1 shows the statistics of the data used, where each column shows number of sentences, number of words, vocabulary, and mean length of a sentence, respectively.

	sent.	words	vocab.	Lmean
Train set				
English	29998	419113	5940	14.0
Spanish		388788	9791	13.0
Test set				
English	500	7412	963	14.8
Spanish		6899	1280	13.8

Table 1: *LC-Star English-Spanish Parallel corpus.*

There are 116 unseen words in the Spanish test set (1.7% of all words), and 48 unseen words in the English set (0.7% of all words), an expected big difference given the much more inflectional nature of the Spanish language.

4.2 Verb Phrase Detection/Classification

Table 2 shows the number of detected verbs using the detection rules presented in section 3.1, and the number of different lemmas they map to. For the test set, the percentage of unseen verb forms and lemmas are also shown.

	verbs	unseen	lemmas	unseen
Train set				
English	56419		768	
Spanish	54460		911	
Test set				
English	1076	5.2%	146	4.7%
Spanish	1061	5.6%	171	4.7%

Table 2: *Detected verb forms in corpus.*

In average, detected English verbs contain 1.81 words, whereas Spanish verbs contain 1.08 words. This is explained by the fact that we are including the personal pronouns in English and modals for future, conditionals and other verb tenses.

4.3 Word alignment results

In order to assess the quality of the word alignment, we randomly selected from the training corpus *350 sentences*, and a manual gold standard alignment has been done with the criterion of Sure and Possible links, in order to compute Alignment Error Rate (AER) as described in (Och and Ney, 2000) and widely used in literature, together with appropriately redefined Recall and Precision measures. Mathematically, they can be expressed thus:

$$recall = \frac{|A \cap S|}{|S|}, \quad precision = \frac{|A \cap P|}{|A|}$$

$$AER = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|}$$

where A is the hypothesis alignment and S is the set of Sure links in the gold standard reference, and P includes the set of Possible *and* Sure links in the gold standard reference.

We have aligned our data using GIZA++ (Och, 2003) from English to Spanish and vice versa (performing 5 iterations of model IBM1 and HMM, and 3 iterations of models IBM3 and IBM4), and have evaluated two symmetrization strategies, namely the union and the intersection, the union always rating the best. Table 3 compares the result when aligning words (current baseline), and when aligning classified verb phrases. In this case, after the alignment we substitute the class for the original verb form and each new word gets the same links the class had. Of course, adverbs and negations are kept apart from the verb and have separate links.

	Recall	Precision	AER
baseline	74.14	86.31	20.07
with class. verbs	76.45	89.06	**17.37**

Table 3: *Results in statistical alignment.*

Results show a significant improvement in AER, which proves that verbal inflected forms and auxiliaries do harm alignment performance in absence of the proposed classification.

4.4 Translation results

We have integrated our classification strategy in an SMT system which implements:

- $Pr(\tilde{e}_i|\tilde{f}_k)$ as a tuples language model (Ngram), as done in (Crego et al., 2004)
- $Pr(e)$ as a standard Ngram language model using SRILM toolkit (Stolcke, 2002)

Parameters have been optimised for BLEU score in a 350 sentences development set. Three references are available for both development and test sets. Table 4 presents a comparison of English to Spanish translation results of the baseline system and the configuration with classification (without dealing with unseen instances). Results are promising, as we achieve a significant mWER error reduction, while still leaving about 5.6 % of the verb forms in the test without translation. Therefore, we expect a further improvement with the treatment of unseen instances.

	mWER	BLEU
baseline	23.16	0.671
with class. verbs	22.22	0.686

Table 4: *Results in English to Spanish translation.*

5 Ongoing and future research

Ongoing research is mainly focused on developing an appropriate generalization technique for unseen instances and evaluating its impact in translation quality.

Later, we expect to run experiments with a much bigger parallel corpus such as the European Parliament corpus, in order to evaluate the improvement due to morphological information for different sizes of the training data. Advanced methods to compute $Pr(\tilde{e}_i|T, \tilde{f}_j)$ should also be tested (based on source and target contextual features).

The next step will be to extend the approach to other potential classes such as:

- Nouns and adjectives. A straightforward strategy would classify all nouns and adjectives to their base form, reducing sparseness.
- Simple Noun phrases. Noun phrases with or without article (determiner), and with or without preposition, could also be classified to the base form of the head noun, leading to a further reduction of the data sparseness, in a subsequent stage. In this case, expressions like `at night`, `the night`, `nights` or `during the night` would all be mapped to the class ’night’.
- Temporal and numeric expressions. As they are usually tackled in a preprocessing stage in current SMT systems, we did not deal with them here.

More on a long-term basis, ambiguous linguistic classification could also be allowed and included in the translation model. For this, incorporating statistical classification tools (chunkers, shallow parsers, phrase detectors, etc.) should be considered, and evaluated against the current implementation.

6 Related Work

The approach to deal with inflected forms presented in (Ueffing and Ney, 2003) is similar in that it also tackles verbs in an English – Spanish task. However, whereas the authors join personal pronouns and auxiliaries to form extended English units and do not transform the Spanish side, leading to an increased English vocabulary, our proposal aims at reducing both vocabularies by mapping all different verb forms to the base form of the head verb.

An improvement in translation using IBM model 1 in an Arabic – English task can be found in (Lee, 2004). From a processed Arabic text with all prefixes and suffixes separated, the author determines which of them should be linked back to the word and which should not. However, no mapping to base forms is performed, and plurals are still different words than singulars.

In (Nießen and Ney, 2004) hierarchical lexicon models including base form and POS information for translation from German into English are introduced, among other morphology-based data transformations. Finally, the same pair of languages is used in (Corston-Oliver and Gamon, 2004), where the inflectional normalization leads to improvements in the perplexity of IBM translation models and reduces alignment errors. However, compound verbs are not mentioned.

7 Conclusion

A proposal of linguistically classifying translation phrases to improve statistical machine translation performance has been presented. This classification allows for a better translation modeling and a generalization to unseen forms. A preliminary implementation detecting verbs in an English – Spanish task has been presented. Experiments show a significant improvement in word alignment, and in preliminary translation results. Ongoing and future research lines are discussed.

References

T. Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proc. of the Sixth Applied Natural Language Processing (ANLP-2000)*, Seattle, WA.

X. Carreras, I. Chao, L. Padr´o, and M. Padr´o. 2004. Freeling: An open-source suite of language analyzers. *4th Int. Conf. on Language Resources and Evaluation, LREC'04*, May.

S. Corston-Oliver and M. Gamon. 2004. Normalizing german and english inflectional morphology to improve statistical word alignment. *Proc. of the 6th Conf. of the Association for Machine Translation in the Americas*, pages 48–57, October.

J.M. Crego, J. Mariño, and A. de Gispert. 2004. Finite-state-based and phrase-based statistical machine translation. *Proc. of the 8th Int. Conf. on Spoken Language Processing, ICSLP'04*, pages 37–40, October.

Y.S. Lee. 2004. Morphological analysis for statistical machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 57–60, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, K. Miller, and R. Tengi. 1991. Five papers on wordnet. *Special Issue of International Journal of Lexicography*, 3(4):235–312.

S. Nießen and H. Ney. 2004. Statistical machine translation with scarce resources using morpho-syntactic information. *Computational Linguistics*, 30(2):181–204, June.

F.J. Och and H. Ney. 2000. Improved statistical alignment models. *38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, October.

F.J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449, December.

F.J. Och. 2003. Giza++ software. http://www-i6.informatik.rwth-aachen.de/˜och/ software/giza++.html.

A. Stolcke. 2002. Srilm - an extensible language modeling toolkit. *Proc. of the 7th Int. Conf. on Spoken Language Processing, ICSLP'02*, September.

N. Ueffing and H. Ney. 2003. Using pos information for smt into morphologically rich languages. *10th Conf. of the European Chapter of the Association for Computational Linguistics*, pages 347–354, April.

R. Zens, F.J. Och, and H. Ney. 2004. Improvements in phrase-based statistical machine translation. *Proc. of the Human Language Technology Conference, HLT-NAACL'2004*, pages 257–264, May.

Automatic Induction of a CCG Grammar for Turkish

Ruken Çakıcı
School of Informatics
Institute for Communicating and Collaborative Systems
University of Edinburgh
2 Buccleuch Place, Edinburgh EH8 9LW
United Kingdom
r.cakici@sms.ed.ac.uk

Abstract

This paper presents the results of automatically inducing a Combinatory Categorial Grammar (CCG) lexicon from a Turkish dependency treebank. The fact that Turkish is an agglutinating free word-order language presents a challenge for language theories. We explored possible ways to obtain a compact lexicon, consistent with CCG principles, from a treebank which is an order of magnitude smaller than Penn WSJ.

1 Introduction

Turkish is an agglutinating language, a single word can be a sentence with tense, modality, polarity, and voice. It has free word-order, subject to discourse restrictions. All these properties make it a challenge to language theories like CCG (Steedman (2000)).

Several studies have been made into building a CCG for Turkish (Bozşahin, 2002; Hoffman, 1995). Bozşahin builds a morphemic lexicon to model the phrasal scope of the morphemes which cannot be acquired with classical lexemic approach. He handles scrambling with type raising and composition. Hoffman proposes a generalisation of CCG (Multiset-CCG) for argument scrambling. She underspecifies the directionality, which results in an undesirable increase in the generative power of the grammar. However, Baldridge (2002) gives a more restrictive form of free order CCG. Both Hoffman and Baldridge ignore morphology and treat the inflected forms as different words.

The rest of this section contains an overview of the underlying formalism (1.1). This is followed by a review of the relevant work (1.2). In Section 2, the properties of the data are explained. Section 3 then gives a brief sketch of the algorithm used to induce a CCG lexicon, with some examples of how certain phenomena in Turkish are handled. As is likely to be the case for most languages for the foreseeable future, the Turkish treebank is quite small (less than 60K words). A major emphasis in the project is on generalising the induced lexicon to improve coverage. Results and future work are discussed in the last two sections.

1.1 Combinatory Categorial Grammar

Combinatory Categorial Grammar (Ades and Steedman, 1982; Steedman, 2000) is an extension to the classical Categorial Grammar (CG) of Ajdukiewicz (1935) and Bar-Hillel (1953). CG, and extensions to it, are lexicalist approaches which deny the need for movement or deletion rules in syntax. Transparent composition of syntactic structures and semantic interpretations, and flexible constituency make CCG a preferred formalism for long-range dependencies and non-constituent coordination in many languages e.g. English, Turkish, Japanese, Irish, Dutch, Tagalog (Steedman, 2000; Baldridge, 2002).

The categories in categorial grammars can be atomic, or functions which specify the directionality of their arguments. A lexical item in a CG can be represented as the triplet: $\phi := \sigma\colon \lambda$ where ϕ is the phonological form, σ is its syntactic type, and λ its semantic type. Some examples are:

Proceedings of the ACL Student Research Workshop, pages 73–78,
Ann Arbor, Michigan, June 2005.

(1) a. $book := N\colon \mathsf{book}$
b. $oku := (S\backslash NP)\backslash NP\colon \lambda x.\lambda y.\mathsf{read}\,xy$

In classical CG, there are two kinds of application rules, which are presented below:

(2) Forward Application (>):
$$X/Y\colon f \quad Y\colon a \quad \Rightarrow \quad X\colon fa$$
Backward Application (<):
$$Y\colon a \quad X\backslash Y\colon f \quad \Rightarrow \quad X\colon fa$$

In addition to functional application rules, CCG has combinatory operators for composition (**B**), type raising (**T**), and substitution (**S**).[1] These operators increase the expressiveness to mildly context-sensitive while preserving the transparency of syntax and semantics during derivations, in contrast to the classical CG, which is context-free (Bar-Hillel et al., 1964).

(3) Forward Composition (>**B**):
$$X/Y\colon f \quad Y/Z\colon g \quad \Rightarrow \quad X/Z\colon \lambda x.f(gx)$$
Backward Composition (<**B**):
$$Y\backslash Z\colon g \quad X\backslash Y\colon f \quad \Rightarrow \quad X\backslash Z\colon \lambda x.f(gx)$$

(4) Forward Type Raising (>**T**):
$$X\colon a \quad \Rightarrow \quad T/(T\backslash X)\colon \lambda f.f[a]$$
Backward Type Raising (<**T**):
$$X\colon a \quad \Rightarrow \quad T\backslash(T/X)\colon \lambda f.f[a]$$

Composition and type raising are used to handle syntactic coordination and extraction in languages by providing a means to construct constituents that are not accepted as constituents in other theories.

1.2 Relevant Work

Julia Hockenmaier's robust CCG parser builds a CCG lexicon for English that is then used by a statistical model using the Penn Treebank as data (Hockenmaier, 2003). She extracts the lexical categories by translating the treebank trees to CCG derivation trees. As a result, the leaf nodes have CCG categories of the lexical entities. Head-complement distinction is not transparent in the Penn Treebank so Hockenmaier uses an algorithm to find the heads (Collins, 1999). There are some inherent advantages to our use of a dependency treebank that only represents surface dependencies. For example, the head is always known, because dependency links are from dependant to head. However, some problems are caused by that fact that only surface dependencies are included. These are discussed in Section 3.5.

2 Data

The METU-Sabancı Treebank is a subcorpus of the METU Turkish Corpus (Atalay et al., 2003; Oflazer et al., 2003). The samples in the corpus are taken from 3 daily newspapers, 87 journal issues and 201 books. The treebank has 5635 sentences.There are a total of 53993 tokens. The average sentence length is about 8 words. However, a Turkish word may correspond to several English words, since the morphological information which exists in the treebank represents additional information including part-of-speech, modality, tense, person, case, etc. The list of the syntactic relations used to model the dependency relations are the following.

1.Subject 2. Object 3.Modifier
4.Possessor 5.Classifier 6.Determiner
7.Adjunct 8.Coordination 9.Relativiser
10.Particles 11.S.Modifier 12.Intensifier
13. Vocative 14. Collocation 15. Sentence
16.ETOL

ETOL is used for constructions very similar to phrasal verbs in English. "Collocation" is used for the idiomatic usages and word sequences with certain patterns. Punctuation marks do not play a role in the dependency structure unless they participate in a relation, such as the use of comma in coordination. The label "Sentence" links the head of the sentence to the punctuation mark or a conjunct in case of coordination. So the head of the sentence is always known, which is helpful in case of scrambling. Figure 1 shows how (5) is represented in the treebank.

(5) Kapının kenarındaki duvara dayanıp bize baktı bir an.
(He) looked at us leaning on the wall next to the door, for a moment.

The dependencies in Turkish treebank are surface dependencies. Phenomena such as traces and pro-drop are not modelled in the treebank. A word

[1] Substitution and others will not be mentioned here. Interested reader should refer to Steedman (2000).

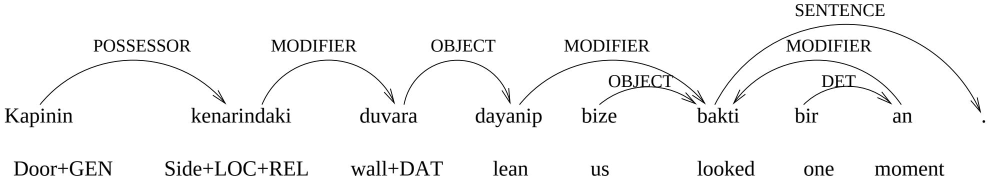


Figure 1: The graphical representation of the dependencies

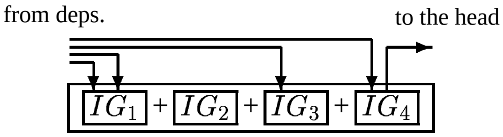


Figure 2: The structure of a word

can be dependent on only one word but words can have more than one dependants. The fact that the dependencies are from the head of one constituent to the head of another (Figure 2) makes it easier to recover the constituency information, compared to some other treebanks e.g. the Penn Treebank where no clue is given regarding the head of the constituents.

Two principles of CCG, Head Categorial Uniqueness and Lexical Head Government, mean both extracted and in situ arguments depend on the same category. This means that long-range dependencies must be recovered and added to the trees to be used in the lexicon induction process to avoid wrong predicate argument structures (Section 3.5).

3 Algorithm

The lexicon induction procedure is recursive on the arguments of the head of the main clause. It is called for every sentence and gives a list of the words with categories. This procedure is called in a loop to account for all sentential conjuncts in case of coordination (Figure 3).

Long-range dependencies, which are crucial for natural language understanding, are not modelled in the Turkish data. Hockenmaier handles them by making use of traces in the Penn Treebank (Hockenmaier, 2003)[sec 3.9]. Since Turkish data do not have traces, this information needs to be recovered from morphological and syntactic clues. There are no relative pronouns in Turkish. Subject and object extraction, control and many other phenomena are marked by morphological processes on the subordinate verb. However, the relative morphemes behave in a similar manner to relative pronouns in English (Çakıcı, 2002). This provides the basis for a heuristic method for recovering long range dependencies in extractions of this type, described in Section 3.5.

```
recursiveFunction(index i, Sentence s)
headcat = findheadscat(i)
//base case
if myrel is “MODIFIER”
    handleMod(headcat)
elseif “COORDINATION”
    handleCoor(headcat)
elseif “OBJECT”
    cat = NP
elseif “SUBJECT”
    cat = NP[nom]
elseif “SENTENCE”
    cat = S
.
.
if hasObject(i)
    combCat(cat,“NP”)
if hasSubject(i)
    combCat(cat,“NP[nom]”)
//recursive case
    forall arguments in arglist
        recursiveFunction(argument,s);
```

Figure 3: The lexicon induction algorithm

3.1 Pro-drop

The subject of a sentence and the genitive pronoun in possessive constructions can drop if there are morphological cues on the verb or the possessee. There is no pro-drop information in the treebank, which is consistent with the surface dependency approach. A *[nom]* (for nominative case) feature is added to the NPs by us to remove the ambiguity for verb categories. All sentences must have a nominative subject.[2] Thus, a verb with a category *S\NP* is assumed to be transitive. This information will be useful in generalising the lexicon during future work (Section 5).

	original	pro-drop
transitive	(S\NP[nom])\NP	S\NP
intransitive	S\NP[nom]	S

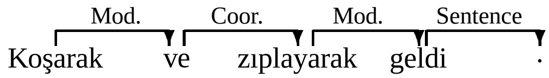
3.2 Adjuncts

Adjuncts can be given CCG categories like *S/S* when they modify sentence heads. However, adjuncts can modify other adjuncts, too. In this case we may end up with categories like (6), and even more complex ones. CCG's composition rule (3) means that as long as adjuncts are adjacent they can all have *S/S* categories, and they will compose to a single *S/S* at the end without compromising the semantics. This method eliminates many gigantic adjunct categories with sparse counts from the lexicon, following (Hockenmaier, 2003).

(6) daha:= *(((S/S)/(S/S))/((S/S)/(S/S)))/*
(((S/S)/(S/S))/((S/S)/(S/S)))
'more'

3.3 Coordination

The treebank annotation for a typical coordination example is shown in (7). The constituent which is directly dependent on the head of the sentence, "zıplayarak" in this case, takes its category according to the algorithm. Then, conjunctive operator is given the category *(X\X)/X* where *X* is the category of "zıplayarak" (or whatever the category of the last conjunct is), and the first conjunct takes the same category as *X*. The information in the treebank is not enough to distinguish sentential coordination and VP coordination. There are about 800 sentences of this type. We decided to leave them out to be annotated appropriately in the future.

(7)


He came running and jumping.

3.4 NPs

Object heads are given *NP* categories. Subject heads are given *NP[nom]*. The category for a modifier of a subject NP is *NP[nom]/NP[nom]* and the modifier for an object NP is *NP/NP* since NPs are almost always head-final.

3.5 Subordination and Relativisation

The treebank does not have traces or null elements. There is no explicit evidence of extraction in the treebank; for example, the heads of the relative clauses are represented as modifiers. In order to have the same category type for all occurences of a verb to satisfy the Principle of Head Categorial Uniqueness, heuristics to detect subordination and extraction play an important role.

(8) Kitabı okuyan adam uyudu.
Book+ACC read+PRESPART man slept.
The man who read the book slept

These heuristics consist of morphological information like existence of a "PRESPART" morpheme in (8), and part-of-speech of the word. However, there is still a problem in cases like (9a) and (9b). Since case information is lost in Turkish extractions, surface dependencies are not enough to differentiate between an adjunct extraction (9a) and an object extraction (9b). A *T.LOCATIVE.ADJUNCT* dependency link is added from "araba" to "uyuduğum" to emphasize that the predicate is intransitive and it may have a locative adjunct. Similarly, a *T.OBJECT* link is added from "kitap" to "okuduğum". Similar labels were added to the treebank manually for approximately 800 sentences.

(9) a. Uyuduğum araba yandı.
Sleep+PASTPART car burn+PAST.
The car I slept in burned.

b. Okuduğum kitap yandı.
Read+PASTPART book burn+PAST.
The book I read burned.

The relativised verb in (9b) is given a transitive verb category with pro-drop, *(S\NP)*, instead of *(NP/NP)\NP*, as the Principle of Head Categorial Uniqueness requires. However, to complete the process we need the relative pronoun equivalent in Turkish,*-dHk+AGR*. A lexical entry with

[2] This includes the passive sentences in the treebank

category *(NP/NP)\(S\NP)* is created and added to the lexicon to give the categories in (10) following Bozşahin (2002).[3]

(10)	Oku	-duğum	kitap	yandı.
	S\NP	(NP/NP)\(S\NP)	NP	S\NP

4 Results

The output is a file with all the words and their CCG categories. The frequency information is also included so that it can be used in probabilistic parsing.

The most frequent words and their most frequent categories are given in Figure 4. The fact that the 8th most frequent word is the non-function word “dedi”(*said*) reveals the nature of the sources of the data —mostly newspapers and novels.

In Figure 5 the most frequent category types are shown. The distribution reflects the real usage of the language (some interesting categories are explained in the last column of the table). There are 518 distinct category types in total at the moment and 198 of them occur only once, but this is due to the fact that the treebank is relatively small (and there are quite a number of annotation mistakes in the version we are using).

In comparison with the English treebank lexicon (1224 types with around 417 occuring only once (Hockenmaier, 2003)) this probably is not a complete inventory of category types. It may be that dependency relations are too few to make the correct category assignment automatically. For instance, all adjectives and adverbs are marked as “MODIFIER”. Figure 6 shows that even after 4500 sentences the curve for most frequent categories has not converged. The data set is too small to give convergence and category types are still being added as unseen words appear. Hockenmaier (2003) shows that the curve for categories with frequencies greater than 5 starts to converge only after 10K sentences in the Penn Treebank.[4]

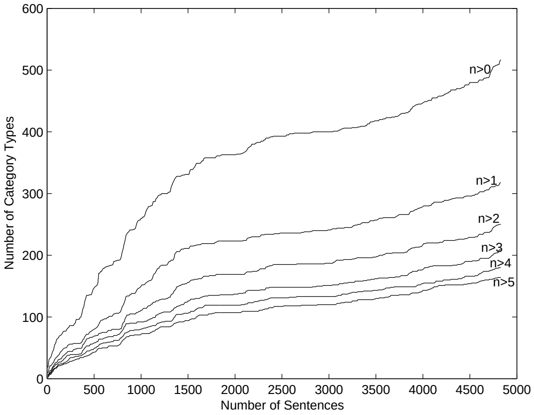


Figure 6: The growth of category types

5 Future Work

The lexicon is going to be trained and tested with a version of the statistical parser written by Hockenmaier (2003). There may be some alterations to the parser, since we will have to use different features to the ones that she used, such as morphological information.

Since the treebank is considerably small compared to the Penn WSJ treebank, generalisation of the lexicon and smoothing techniques will play a crucial role. Considering that there are many small-scale treebanks being developed for “understudied” languages, it is important to explore ways to boost the performances of statistical parsers from small amounts of human labeled data.

Generalisation of this lexicon using the formalism in Baldridge (2002) would result in a more compact lexicon, since a single entry would be enough for several word order permutations. We also expect that the more effective use of morphological information will give better results in terms of parsing performance. We are also considering the use of unlabelled data to learn word-category pairs.

References

A.E. Ades and Mark Steedman. 1982. On the order of words. *Linguistics and Philosophy*, 4:517–558.

Kazimierz Ajdukiewicz. 1935. Die syntaktische konnexitat. In *Polish Logic, ed. Storrs McCall, Oxford University Press*, pages 207–231.

[3] Current version of the treebank has empty “MORPH” fields. Therefore, we are using dummy tokens for relative morphemes at the moment.

[4] The slight increase after 3800 sentences may be because the data are not uniform. Relatively longer sentences from a history article start after short sentences from a novel.

token	**eng.**	**freq.**	**pos**	**most freq. cat**	**fwc***
,	Comma	2286	Conj	(NP/NP)\NP	159
bir	a	816	Det	NP/NP	373
-yAn	who	554	Rel. morph.	(NP/NP)\(S\NP)	554
ve	and	372	Conj	(NP/NP)\NP	100
de	too	335	Int	NP[nom]\NP[nom]	116
bu	this	279	Det	NP/NP	110
da	too	268	Int	NP[nom]\NP[nom]	86
dedi	said	188	Verb	S\NP	87
-DHk+AGR	which	163	Rel. morph.	(NP/NP)\(S\NP)	163
Bu	This	159	Det	NP/NP	38
gibi	like	148	Postp	(S/S)\NP	21
o	that	141	Det	NP/NP	37

***fwc** Frequency of the word occuring with the given category

Figure 4: The lexicon statistics

cattype	frequency	rank	type
NP	5384	1	noun phrase
NP/NP	3292	2	adjective,determiner, etc
NP[nom]	3264	3	subject NP
S/S	3212	4	sentential adjunct
S\NP	1883	5	transitive verb with pro-drop
S	1346	6	sentence
S\NP[nom]	1320	7	intransitive verb
(S\NP[nom])\NP	827	9	transitive verb

Figure 5: The most frequent category types

Nart B. Atalay, Kemal Oflazer, and Bilge Say. 2003. The annotation process in the Turkish Treebank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora*, Budapest, Hungary.

Jason M. Baldridge. 2002. *Lexically Specified Derivation Control in Combinatory Categorial Grammar*. Ph.D. thesis, University of Edinburgh.

Yehoshua Bar-Hillel, C. Gaifman, and E. Shamir. 1964. On categorial and phrase structure grammars. In *Language and Information ed. Bar-Hillel, Addison-Wesley*, pages 99–115.

Yehoshua Bar-Hillel. 1953. A quasi-arithmetic description for syntactic description. *Language*, 29:47–58.

Cem Bozşahin. 2002. The combinatory morphemic lexicon. *Computational Linguistics*, 28(2):145–186.

Ruken Çakıcı. 2002. A computational interface for syntax and morphemic lexicons. Master's thesis, Middle East Technical University.

Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Julia Hockenmaier. 2003. *Data Models for statistical parsing with Combinatory Categorial Grammar*. Ph.D. thesis, University of Edinburgh.

Beryl Hoffman. 1995. *The Computational Analysis of the Syntax and Interpretation of ”Free” Word Order in Turkish*. Ph.D. thesis, University of Pennsylvania.

Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gokhan Tür. 2003. Building a turkish treebank. In Abeille Anne, editor, *Treebanks: Building and Using Parsed Corpora*, pages 261–277. Kluwer, Dordrecht.

Mark Steedman. 2000. *The Syntactic Process*. The MIT Press, Cambridge, Massachusetts.

Dialogue Act Tagging for Instant Messaging Chat Sessions

Edward Ivanovic
Department of Computer Science and Software Engineering
University of Melbourne
Victoria 3010, Australia
edwardi@csse.unimelb.edu.au

Abstract

Instant Messaging chat sessions are real-time text-based conversations which can be analyzed using dialogue-act models. We describe a statistical approach for modelling and detecting dialogue acts in Instant Messaging dialogue. This involved the collection of a small set of task-based dialogues and annotating them with a revised tag set. We then dealt with segmentation and synchronisation issues which do not arise in spoken dialogue. The model we developed combines naive Bayes and dialogue-act n-grams to obtain better than 80% accuracy in our tagging experiment.

1 Introduction

Instant Messaging (IM) dialogue has received relatively little attention in discourse modelling. The novelty and popularity of IM dialogue and the significant differences between written and spoken English warrant specific research on IM dialogue. We show that IM dialogue has some unique problems and attributes not found in transcribed spoken dialogue, which has been the focus of most work in discourse modelling. The present study addresses the problems presented by these differences when modelling dialogue acts in IM dialogue.

Stolcke et al. (2000) point out that the use of dialogue acts is a useful first level of analysis for describing discourse structure. Dialogue acts are based on the illocutionary force of an utterance from speech act theory, and represent acts such as assertions and declarations (Austin, 1962; Searle, 1979). This theory has been extended in dialogue acts to model the conversational functions that utterances can perform. Dialogue acts have been used to benefit tasks such as machine translation (Tanaka and Yokoo, 1999) and the automatic detection of dialogue games (Levin et al., 1999). This deeper level of discourse understanding may help replace or assist a support representative using IM dialogue by suggesting responses that are more sophisticated and realistic to a human dialogue participant.

The unique problems and attributes exhibited by IM dialogue prohibit existing dialogue act classification methods from being applied directly. We present solutions to some of these problems along with methods to obtain high accuracy in automated dialogue act classification. A statistical discourse model is trained and then used to classify dialogue acts based on the observed words in an utterance. The training data are online conversations between two people: a customer and a shopping assistant, which we collected and manually annotated. Table 1 shows a sample of the type of dialogue and discourse structure used in this study.

We begin by considering the preliminary issues that arise in IM dialogue, why they are problematic when modelling dialogue acts, and present their solutions in §2. With the preliminary problems solved, we investigate the dialogue act labelling task with a description of our data in §3. The remainder of the paper describes our experiment involving the training of a naive Bayes model combined with a n-gram discourse model (§4). The results of this model and evaluation statistics are presented in §5. §6 contains a discussion of the approach we used including its strengths, areas of improvement, and issues for future research followed by the conclusion in §7.

Proceedings of the ACL Student Research Workshop, pages 79–84,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

Turn	Msg	Sec	Speaker	Message
5	8	18	Customer	[i was talking to mike and my browser crashed]$^{U_8:\textsc{Statement}}$ - [can you transfer me to him again?]$^{U_9:\textsc{Yes-No-Question}}$
5	9	7	Customer	[he found a gift i wanted]$^{U_{10}:\textsc{Statement}}$
6	10	35	Sally	[I will try my best to help you find the gift,]$^{U_{11}:\textsc{Statement}}$ [please let me know the request]$^{U_{12}:\textsc{Request}}$
6	11	9	Sally	[Mike is not available at this point of time]$^{U_{13}:\textsc{Statement}}$
7	12	1	Customer	[but mike already found it]$^{U_{14}:\textsc{Statement}}$ [isn't he there?]$^{U_{15}:\textsc{Yes-No-Question}}$
8	13	8	Customer	[it was a remote control car]$^{U_{16}:\textsc{Statement}}$
9	14	2	Sally	[Mike is not available right now.]$^{U_{17}:\textsc{No-Answer}}$ [I am here to assist you.]$^{U_{18}:\textsc{Statement}}$
10	15	28	Sally	[Sure Customer,]$^{U_{19}:\textsc{Response-Ack}}$ [I will search for the remote control car.]$^{U_{20}:\textsc{Statement}}$

Table 1: An example of unsynchronised messages occurring when a user prematurely assumes a turn is finished. Here, message ("Msg") 12 is actually in response to 10, not 11 since turn 6 was sent as 2 messages: 10 and 11. We use the seconds elapsed ("Sec") since the previous message as part of a method to re-synchronise messages. Utterance boundaries and their respective dialogue acts are denoted by U_n.

2 Issues in Instant Messaging Dialogue

There are several differences between IM and transcribed spoken dialogue. The dialogue act classifier described in this paper is dependent on preprocessing tasks to resolve the issues discussed in this section.

Sequences of words in textual dialogue are grouped into three levels. The first level is a Turn, consisting of at least one Message, which consists of at least one Utterance, defined as follows:

Turn: Dialogue participants normally take turns writing.

Message: A message is defined as a group of words that are sent from one dialogue participant to the other as a single unit. A single turn can span multiple messages, which sometimes leads to accidental interruptions as discussed in §2.2.

Utterance: This is the shortest unit we deal with and can be thought of as one complete semantic unit—something that has a meaning. This can be a complete sentence or as short as an emoticon (e.g. ":-)" to smile).

Several lines from one of the dialogues in our corpus are shown as an example denoted with Turn, Message, and Utterance boundaries in Table 1.

2.1 Utterance Segmentation

Because dialogue acts work at the utterance level and users send messages which may contain more than one utterance, we first need to segment the messages by detecting utterance boundaries. Messages in our data were manually labelled with one or more dialogue act depending on the number of utterances each message contained. Labelling in this fashion had the effect of also segmenting messages into utterances based on the dialogue act boundaries.

2.2 Synchronising Messages in IM Dialogue

The end of a turn is not always obvious in typed dialogue. Users often divide turns into multiple messages, usually at clause or utterance boundaries, which can result in the end of a message being mistaken as the end of that turn. This ambiguity can lead to accidental turn interruptions which cause messages to become unsynchronised. In these cases each participant tends to respond to an earlier message than the immediately previous one, making the conversation seem somewhat incoherent when read as a transcript. An example of such a case is shown in Table 1 in which Customer replied to message 10 with message 12 while Sally was still completing turn 6 with message 11. If the resulting discourse is read sequentially it would seem that the customer ignored the information provided in message 11. The time between messages shows that only 1 second elapsed between messages 11 and 12, so message 12 must in fact be in response to message 10.

Message M_i is defined to be *dependent* on message M_d if the user wrote M_i having already seen and presumably considered M_d. The importance of unsynchronised messages is that they result in the dialogue acts also being out of order, which is problematic when using bigram or higher-order n-gram language models. Therefore, messages are re-synchronised as described in §3.2 before training and classification.

3 The Dialogue Act Labelling Task

The domain being modelled is the online shopping assistance provided as part of the MSN Shopping site. People are employed to provide live assistance via an IM medium to potential customers who need help in finding items for purchase. Several dialogues were collected using this service, which were then manually labelled with dialogue acts and used to train our statistical models.

There were 3 aims of this task: 1) to obtain a realistic corpus; 2) to define a suitable set of dialogue act tags; and 3) to manually label the corpus using the dialogue act tag set, which is then used for training the statistical models for automatic dialogue act classification.

3.1 Tag Set

We chose 12 tags by manually labelling the dialogue corpus using tags that seemed appropriate from the 42 tags used by Stolcke et al. (2000) based on the Dialog Act Markup in Several Layers (DAMSL) tag set (Core and Allen, 1997). Some tags, such as UNINTERPRETABLE and SELF-TALK, were eliminated as they are not relevant for typed dialogue. Tags that were difficult to distinguish, given the types of utterances in our corpus, were collapsed into one tag. For example, NO ANSWERS, REJECT, and NEGATIVE NON-NO ANSWERS are all represented by NO-ANSWER in our tag set.

The Kappa statistic was used to compare inter-annotator agreement normalised for chance (Siegel and Castellan, 1988). Labelling was carried out by three computational linguistics graduate students with 89% agreement resulting in a Kappa statistic of 0.87, which is a satisfactory indication that our corpus can be labelled with high reliability using our tag set (Carletta, 1996).

A complete list of the 12 dialogue acts we used is shown in Table 2 along with examples and the frequency of each dialogue act in our corpus.

Tag	Example	%
STATEMENT	I am sending you the page now	36.0
THANKING	Thank you for contacting us	14.7
YES-NO-QUESTION	Did you receive the page?	13.9
RESPONSE-ACK	Sure	7.2
REQUEST	Please let me know how I can assist	5.9
OPEN-QUESTION	how do I use the international version?	5.3
YES-ANSWER	yes, yeah	5.1
CONVENTIONAL-CLOSING	Bye Bye	2.9
NO-ANSWER	no, nope	2.5
CONVENTIONAL-OPENING	Hello Customer	2.3
EXPRESSIVE	haha, :-), grr	2.3
DOWNPLAYER	my pleasure	1.9

Table 2: The 12 dialogue act labels with examples and frequencies given as percentages of the total number of utterances in our corpus.

3.2 Re-synchronising Messages

The typing rate is used to determine message dependencies. We calculate the typing rate by $\frac{time(M_i)-time(M_d)}{length(M_i)}$, which is the elapsed time between two messages divided by the number of characters in M_i. The dependent message M_d may be the immediately preceding message such that $d = i - 1$ or any earlier message where $0 < d < i$ with the first message being M_1. This algorithm is shown in Algorithm 1.

Algorithm 1 Calculate message dependency for message i

```
$d \leftarrow i$
**repeat**
    $d \leftarrow d - 1$
    $typing\_rate \leftarrow \frac{time(M_i)-time(M_d)}{length(M_i)}$
**until** $typing\_rate < typing\_threshold$ or $d = 1$ or $speaker(M_i) = speaker(M_d)$
```

The $typing\_threshold$ in Algorithm 1 was calculated by taking the 90th percentile of all observed typing rates from approximately 300 messages that had their dependent messages manually labelled resulting in a value of 5 characters per second. We found that 20% of our messages were unsynchronised, giving a baseline accuracy of automatically detecting message dependencies of 80% assuming that $M_d = M_{i-1}$. Using the method described, we achieved a correct dependency detection accuracy of 94.2%.

4 Training on Speech Acts

Our goal is to perform automatic dialogue act classification of the current utterance given any previous utterances and their tags. Given all available evidence E about a dialogue, the goal is to find the dialogue act sequence U with the highest posterior probability $P(U|E)$ given that evidence. To achieve this goal, we implemented a naive Bayes classifier using bag-of-words feature representation such that the most probable dialogue act $\hat{d}$ given a bag-of-words input vector $\bar{v}$ is taken to be:

$$\hat{d} = \operatorname*{argmax}_{d \in D} \frac{P(\bar{v}|d)P(d)}{P(\bar{v})} \quad (1)$$

$$P(\bar{v}|d) \approx \prod_{j=1}^{n} P(v_j|d) \quad (2)$$

$$\hat{d} = \operatorname*{argmax}_{d \in D} P(d) \prod_{j=1}^{n} P(v_j|d) \quad (3)$$

where v_j is the jth element in $\bar{v}$, D denotes the set of all dialogue acts and $P(\bar{v})$ is constant for all $d \in D$.

The use of $P(d)$ in Equation 3 assumes that dialogue acts are independent of one another. However, we intuitively know that if someone asks a Yes-No-Question then the response is more likely to be a Yes-Answer rather than, say, Conventional-Closing. This intuition is reflected in the bigram transition probabilities obtained from our corpus.[1]

To capture this dialogue act relationship we trained standard n-gram models of dialogue act history with add-one smoothing for the calculation of $P(v_j|d)$. The bigram model uses the posterior probability $P(d|H)$ rather than the prior probability $P(d)$ in Equation 3, where H is the n-gram context vector containing the previous dialogue act or previous 2 dialogue acts in the case of the trigram model.

Model	Min	Max	Mean	Hit %	Px
Baseline	—	—	36.0%	—	—
Likelihood	72.3%	90.5%	80.1%	—	—
Unigram	74.7%	90.5%	80.6%	100	7.7
Bigram	75.0%	92.4%	81.6%	97	4.7
Trigram	69.5%	94.1%	80.9%	88	3.3

Table 3: Mean accuracy of labelling utterances with dialogue acts using n-gram models. Shown with hit-rate results and perplexities ("Px")

5 Experimental Results

Evaluation of the results was conducted via 9-fold cross-validation across the 9 dialogues in our corpus using 8 dialogues for training and 1 for testing. Table 3 shows the results of running the experiment with various models replacing the prior probability, $P(d)$, in Equation 3. The Min, Max, and Mean columns are obtained from the cross-validation technique used for evaluation. The baseline used for this task was to assign the most frequently observed dialogue act to each utterance, namely, Statement.

Omitting $P(d)$ from Equation 3 such that only the likelihood (Equation 2) of the naive Bayes formula is used resulted in a mean accuracy of 80.1%. The high accuracy obtained with only the likelihood reflects the high dependency between dialogue acts and the actual words used in utterances. This dependency is represented well by the bag-of-words approach. Using $P(d)$ to arrive at Equation 3 yields a slight increase in accuracy to 80.6%.

The bigram model obtains the best result with 81.6% accuracy. This result is due to more accurate predictions with $P(d|H)$. The trigram model produced a slightly lower accuracy rate, partly due to a lack of training data and to dialogue act adjacency pairs not being dependent on dialogue acts further removed as discussed in §4.

In order to gauge the effectiveness of the bigram and trigram models in view of the small amount of training data, hit-rate statistics were collected during testing. These statistics, presented in Table 3, show the percentage of conditions that existed in the various models. Conditions that did not exist were not counted in the accuracy measure during evaluation.

The perplexities (Cover and Thomas, 1991) for the various n-gram models we used are shown in

[1] Due to space constraints, the dialogue act transition table has been omitted from this paper and is made available at http://www.cs.mu.oz.au/∼edwardi/papers/da_transitions.html

Table 3. The biggest improvement, indicated by a decreased perplexity, comes when moving from the unigram to bigram models as expected. However, the large difference between the bigram and trigram models is somewhat unexpected given the theory of adjacency pairs. This may be a result of insufficient training data as would be suggested by the lower trigram hit rate.

6 Discussion and Future Research

As indicated by the Kappa statistics in §3.1, labelling utterances with dialogue acts can sometimes be a subjective task. Moreover, there are many possible tag sets to choose from. These two factors make it difficult to accurately compare various tagging methods and is one reason why Kappa statistics and perplexity measures are useful. The work presented in this paper shows that using even the relatively simple bag-of-words approach with a naive Bayes classifier can produce very good results.

One important area not tackled by this experiment was that of utterance boundary detection. Multiple utterances are often sent in one message, sometimes in one sentence, and each utterance must be tagged. Approximately 40% of the messages in our corpus have more than one utterance per message. Utterances were manually marked in this experiment as the study was focussed only on dialogue act classification given a sequence of utterances. It is rare, however, to be given text that is already segmented into utterances, so some work will be required to accomplish this segmentation before automated dialogue act tagging can commence. Therefore, utterance boundary detection is an important area for further research.

The methods used to detect dialogue acts presented here do not take into account sentential structure. The sentences in (1) would thus be treated equally with the bag-of-words approach.

(1) a. john has been to london
b. has john been to london

Without the punctuation (as is often the case with informal typed dialogue) the bag-of-words approach will not differentiate the sentences, whereas if we look at the ordering of even the first two words we can see that "john has ..." is likely to be a STATEMENT whereas "has john ..." would be a question. It would be interesting to research other types of features such as phrase structure or even looking at the order of the first x words and the parts of speech of an utterance to determine its dialogue act.

Aspects of dialogue macrogame theory (DMT) (Mann, 2002) may help to increase tagging accuracy. In DMT, sets of utterances are grouped together to form a *game*. Games may be nested as in the following example:

A: May I know the price range please?
B: In which currency?
A: $US please
B: 200–300

Here, B has nested a clarification question which was required before providing the price range. The bigram model presented in this paper will incorrectly capture this interaction as the sequence YES-NO-QUESTION, OPEN-QUESTION, STATEMENT, STATEMENT, whereas DMT would be able to extract the nested question resulting in the correct pairs of question and answer sequences.

Although other studies have attempted to automatically tag utterances with dialogue acts (Stolcke et al., 2000; Jurafsky et al., 1997; Kita et al., 1996) it is difficult to fairly compare results because the data was significantly different (transcribed spoken dialogue versus typed dialogue) and the dialogue acts were also different ranging from a set of 9 (Kita et al., 1996) to 42 (Stolcke et al., 2000). It may be possible to use a standard set of dialogue acts for a particular domain, but inventing a set that could be used for all domains seems unlikely. This is primarily due to differing needs in various applications. A superset of dialogue acts that covers all domains would necessarily be a large number of tags (at least the 42 identified by Stolcke et al. (2000)) with many tags not being appropriate for other domains.

The best result from our dialogue act classifier was obtained using a bigram discourse model resulting in an average tagging accuracy of 81.6% (see Table 3). Although this is higher than the results from 13 recent studies presented by Stolcke et al. (2000) with accuracy ranging from $\approx 40\%$ to 81.2%, the tasks, data, and tag sets used were all quite different, so any comparison should be used as only a guideline.

7 Conclusion

In this paper, we have highlighted some unique characteristics in IM dialogue that are not found in transcribed spoken dialogue or other forms of written dialogue such as e-mail; namely, utterance segmentation and message synchronisation. We showed the problem of unsynchronised messages can be readily solved using a simple technique utilising the typing-rate and time stamps of messages. We described a method for high-accuracy dialogue act classification, which is an essential part for a deeper understanding of dialogue. In our experiments, the bigram model performed with the highest tagging accuracy which indicates that dialogue acts often occur as adjacency pairs. We also saw that the high tagging accuracy results obtained by the likelihood from the naive Bayes model indicated the high correlation between the actual words and dialogue acts. The Kappa statistics we calculated indicate that our tag set can be used reliably for annotation tasks.

The increasing popularity of IM and automated agent-based support services is ripe with new challenges for research and development. For example, IM provides the ability for an automated agent to ask clarification questions. Appropriate dialogue modelling will enable the automated agent to reliably distinguish questions from statements. More generally, the rapidly expanding scope of online support services provides the impetus for IM dialogue systems and discourse models to be developed further. Our findings have demonstrated the potential for dialogue modelling for IM chat sessions, and opens the way for a comprehensive investigation of this new application area.

Acknowledgments

We thank Steven Bird, Timothy Baldwin, Trevor Cohn, and the anonymous reviewers for their helpful and constructive comments on this paper. We also thank Vanessa Smith, Patrick Ye, and Jeremy Nicholson for annotating the data.

References

John L. Austin. 1962. *How to do Things with Words*. Clarendon Press, Oxford.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254.

Mark Core and James Allen. 1997. Coding dialogs with the DAMSL annotation scheme. *Working Notes of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35.

Thomas M. Cover and Joy A. Thomas. 1991. *Elements of Information Theory*. Wiley, New York.

Daniel Jurafsky, Rebecca Bates, Noah Coccaro, Rachel Martin, Marie Meteer, Klaus Ries, Elizabeth Shriberg, Andreas Stolcke, Paul Taylor, and Carol Van Ess-Dykema. 1997. Automatic detection of discourse structure for speech recognition and understanding. *Proceedings of the 1997 IEEE Workshop on Speech Recognition and Understanding*, pages 88–95.

Kenji Kita, Yoshikazu Fukui, Masaaki Nagata, and Tsuyoshi Morimoto. 1996. Automatic acquisition of probabilistic dialogue models. *Proceedings of the Fourth International Conference on Spoken Language*, 1:196–199.

Lori Levin, Klaus Ries, Ann Thyme-Gobbel, and Alon Lavie. 1999. Tagging of speech acts and dialogue games in spanish call home. *Towards Standards and Tools for Discourse Tagging (Proceedings of the ACL Workshop at ACL'99)*, pages 42–47.

William Mann. 2002. Dialogue macrogame theory. *Proceedings of the 3rd SIGdial Workshop on Discourse and Dialogue*, pages 129–141.

John R. Searle. 1979. *Expression and Meaning: Studies in the Theory of Speech Acts*. Cambridge University Press, Cambridge, UK.

Sidney Siegel and N. John Castellan, Jr. 1988. *Nonparametric statistics for the behavioral sciences*. McGraw-Hill, second edition.

Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373.

Hideki Tanaka and Akio Yokoo. 1999. An efficient statistical speech act type tagging system for speech translation systems. In *Proceedings of the 37th conference on Association for Computational Linguistics*, pages 381–388. Association for Computational Linguistics.

Learning Strategies for Open-Domain Natural Language Question Answering

Eugene Grois
Department of Computer Science
University of Illinois, Urbana-Champaign
Urbana, Illinois
e-grois@uiuc.edu

Abstract

This work presents a model for learning inference procedures for story comprehension through inductive generalization and reinforcement learning, based on classified examples. The learned inference procedures (or strategies) are represented as of sequences of transformation rules. The approach is compared to three prior systems, and experimental results are presented demonstrating the efficacy of the model.

1 Introduction

This paper presents an approach to automatically learning strategies for natural language question answering from examples composed of textual sources, questions, and answers. Our approach is focused on one specific type of text-based question answering known as *story comprehension*. Most TREC-style QA systems are designed to extract an answer from a document contained in a fairly large general collection (Voorhees, 2003). They tend to follow a generic architecture, such as the one suggested by (Hirschman and Gaizauskas, 2001), that includes components for document pre-processing and analysis, candidate passage selection, answer extraction, and response generation. Story comprehension requires a similar approach, but involves answering questions from a single narrative document. An important challenge in text-based question answering in general is posed by the syntactic and semantic variability of question and answer forms, which makes it difficult to establish a match between the question and answer candidate. This problem is particularly acute in the case of story comprehension due to the rarity of information restatement in the single document.

Several recent systems have specifically addressed the task of story comprehension. The Deep Read reading comprehension system (Hirschman et al., 1999) uses a statistical bag-of-words approach, matching the question with the lexically most similar sentence in the story. Quarc (Riloff and Thelen, 2000) utilizes manually generated rules that selects a sentence deemed to contain the answer based on a combination of syntactic similarity and semantic correspondence (i.e., semantic categories of nouns). The Brown University statistical language processing class project systems (Charniak et al., 2000) combine the use of manually generated rules with statistical techniques such as bag-of-words and bag-of-verb matching, as well as deeper semantic analysis of nouns. As a rule, these three systems are effective at identifying the sentence containing the correct answer as long as the answer is explicit and contained entirely in that sentence. They find it difficult, however, to deal with semantic alternations of even moderate complexity. They also do not address situations where answers are split across multiple sentences, or those requiring complex inference.

Our framework, called QABLe (Question-Answering Behavior Learner), draws on prior work in learning action and problem-solving strategies (Tadepalli and Natarajan, 1996; Khardon, 1999). We represent textual sources as sets of features in a sparse domain, and treat the QA task as behavior in a stochastic, partially observable world. QA strategies are learned as sequences of transformation rules capable of deriving certain types of answers from particular text-question combinations. The transformation rules are generated by instantiating primitive domain operators in specific feature contexts. A process of reinforcement learning (Kaebling et al., 1996) is used to select and promote effective transformation rules. We rely on recent work in attribute-efficient relational learning (Khardon et al., 1999; Cumby and Roth, 2000; Even-Zohar and Roth, 2000) to acquire natural representations of the underlying domain features. These

Proceedings of the ACL Student Research Workshop, pages 85–90,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

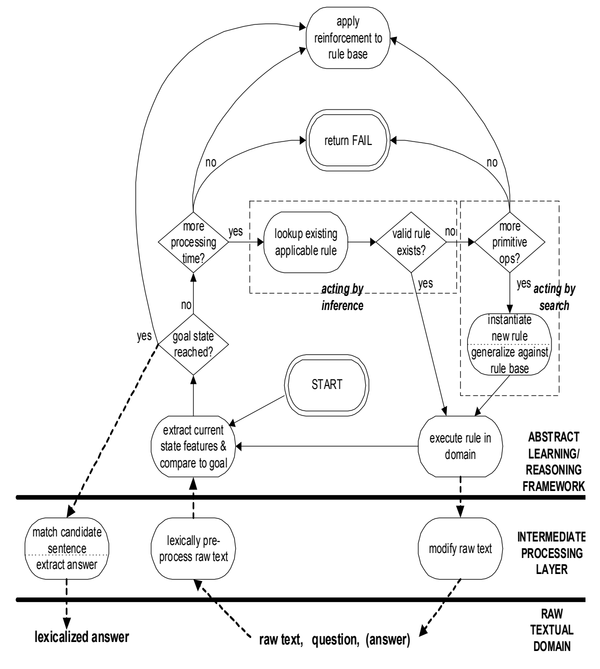


Figure 1. The QABLe architecture for question answering.

representations are learned in the course of interacting with the domain, and encode the features at the levels of abstraction that are found to be conducive to successful behavior. This selection effect is achieved through a combination of inductive generalization and reinforcement learning elements.

The rest of this paper is organized as follows. Section 2 presents the details of the QABLe framework. In section 3 we describe preliminary experimental results which indicate promise for our approach. In section 4 we summarize and draw conclusions.

2 QABLe – Learning to Answer Questions

2.1 Overview

Figure 1 shows a diagram of the QABLe framework. The bottom-most layer is the natural language textual domain. It represents raw textual sources, questions, and answers. The intermediate layer consists of processing modules that translate between the raw textual domain and the top-most layer, an abstract representation used to reason and learn.

This framework is used both for learning to answer questions and for the actual QA task. While learning, the system is provided with a set of training instances, each consisting of a textual narrative, a question, and a corresponding answer. During the performance phase, only the narrative and question are given.

At the lexical level, an answer to a question is generated by applying a series of *transformation rules* to the text of the narrative. These transformation rules augment the original text with one or more additional sentences, such that one of these explicitly contains the answer, *and* matches the form of the question.

On the abstract level, this is essentially a process of searching for a path through problem space that transforms the world state, as described by the textual source and question, into a world state containing an appropriate answer. This process is made efficient by learning answer-generation strategies. These strategies store procedural knowledge regarding the way in which answers are derived from text, and suggest appropriate transformation rules at each step in the answer-generation process. Strategies (and the procedural knowledge stored therein) are acquired by explaining (or deducing) correct answers from training examples. The framework's ability to answer questions is tested only with respect to the kinds of documents it has seen during training, the kinds of questions it has practiced answering, and its interface to the world (domain sensors and operators).

In the next two sections we discuss lexical pre-processing, and the representation of features and relations over them in the QABLe framework. In section 2.4 we look at the structure of transformation rules and describe how they are instantiated. In section 2.5, we build on this information and describe details of how strategies are learned and utilized to generate answers. In section 2.6 we explain how candidate answers are matched to the question, and extracted.

2.2 Lexical Pre-Processing

Several levels of syntactic and semantic processing are required in order to generate structures that facilitate higher order analysis. We currently use MontyTagger 1.2, an off-the-shelf POS tagger based on (Brill, 1995) for POS tagging. At the next tier, we utilize a Named Entity (NE) tagger for proper nouns a semantic category classifier for nouns and noun phrases, and a co-reference resolver (that is limited to pronominal anaphora). Our taxonomy of semantic categories is derived from the list of unique beginners for WordNet nouns (Fellbaum, 1998). We also have a parallel stage that identifies phrase types. Table 1 gives a list of phrase types currently in use, together with the categories of questions each phrase type can answer. In the near future, we plan to utilize a link parser to boost phrase-type tagging accuracy. For questions, we have a classifier that identifies the semantic category of information requested by the question. Currently, this taxonomy is identical to that of semantic categories. However, in the future, it may be expanded to accommodate a wider range of queries. A separate module reformulates questions into statement form for later matching with answer-containing phrases.

2.3 Representing the QA Domain

In this section we explain how features are extracted from raw textual input and tags which are generated by pre-processing modules.

A sentence is represented as a sequence of words $\langle w_1, w_2, \ldots, w_n \rangle$, where *word*($w_i$, word) binds a particular word to its position in the sentence. The k^{th} sentence in a passage is given a unique designation s_k. Several simple functions capture the syntax of the sentence. The sentence *Main* (e.g., main verb) is the controlling element of the sentence, and is recognized by *main*(w_m, s_k). Parts of speech are recognized by the function *pos*, as in *pos*(w_i, NN) and *pos*(w_i, VBD). The relative syntactic ordering of words is captured by the function w_j=*before*(w_i). It can be applied recursively, as w_k = *before*(w_j) = *before*(*before*(w_i)) to generate the entire sentence starting with an arbitrary word, usually the sentence *Main*. *before*() may also be applied as a predicate, such as *before*(w_i, w_j). Thus for each word w_i in the sentence, *inSentence*(w_i, s_i) $\Rightarrow$ *main*(w_m, s_k) $\wedge$ (*before*(w_i, w_m) $\vee$ *before*(w_m, w_i)). A consecutive sequence of words is a *phrase entity* or simply *entity*. It is given the designation e_x and declared by a binding function, such as *entity*(e_x, *NE*) for a named entity, and *entity*(e_x, *NP*) for a syntactic group of type *noun phrase*. Each phrase entity is identified by its *head*, as *head*(w_h, e_x), and we say that the phrase head controls the entity. A phrase entity is defined as *head*(w_h, e_x) $\wedge$ *inPhrase*(w_i, e_x) $\wedge \ldots \wedge$ *inPhrase*(w_j, e_x).

We also wish to represent higher-order relations such as functional roles and semantic categories. Functional dependency between pairs of words is encoded as, for example, *subj*(w_i, w_j) and *aux*(w_j, w_k). Functional groups are represented just like phrase entities. Each is assigned a designation r_x, declared for example, as *func_role*(r_x, *SUBJ*), and defined in terms of its head and members (which may be individual words or composite entities). Semantic categories are similarly defined over the set of words and syntactic phrase entities – for example, *sem_cat*(c_x, *PERSON*) $\wedge$ *head*(w_h, c_x) $\wedge$ *pos*(w_i, NNP) $\wedge$ *word*(w_h, "John").

Semantically, sentences are treated as events defined by their verbs. A multi-sentential passage is represented by tying the member sentences together with relations over their verbs. We declare two such relations – *seq* and *cause*. The *seq* relation between two sentences, *seq*(s_i, s_j) $\Rightarrow$ *prior*(*main*(s_i), *main*(s_j)), is defined as the sequential ordering in time of the corresponding events. The *cause* relation *cause*(s_i, s_j) $\Rightarrow$ *cdep*(*main*(s_i), *main*(s_j)) is defined such that the second event is causally dependent on the first.

2.4 Primitive Operators and Transformation Rules

The system, in general, starts out with no procedural knowledge of the domain (*i.e.*, no transformation rules). However, it is equipped with 9 primitive operators that define basic actions in the domain. Primitive operators are existentially quantified. They have no activation condition, but only an *existence condition* – the minimal binding condition for the operator to be applicable in a given state. A primitive operator has the form $C^E \rightarrow \hat{A}$, where C^E is the existence condition and $\hat{A}$ is an action implemented in the domain. An example primitive operator is

primitive-op-1 : $\exists\, w_x, w_y \rightarrow$ add-word-after-word(w_y, w_x)

Other primitive operators delete words or manipulate entire phrases. Note that primitive operators act directly on the syntax of the domain. In particular, they manipulate words and phrases. A primitive operator bound to a state in the domain constitutes a transformation rule. The procedure

Phrase Type	Comments
SUBJ	"Who" and nominal "What" questions
VERB	event "What" questions
DIR-OBJ	"Who" and nominal "What" questions
INDIR-OBJ	"Who" and nominal "What" questions
ELAB-SUBJ	descriptive "What" questions (eg. what kind)
ELAB-VERB-TIME	
ELAB-VERB-PLACE	
ELAB-VERB-MANNER	
ELAB-VERB-CAUSE	"Why" question
ELAB-VERB-INTENTION	"Why" as well as "What for" question
ELAB-VERB-OTHER	smooth handling of undefined verb phrase types
ELAB-DIR-OBJ	descriptive "What" questions (eg. what kind)
ELAB-INDIR-OBJ	descriptive "What" questions (eg. what kind)
VERB-COMPL	WHERE/WHEN/HOW questions concerning state or status

Table 1. Phrase types used by QABLe framework.

for instantiating transformation rules using primitive operators is given in Figure 2. The result of this procedure is a universally quantified rule having the form $C \wedge G^R \rightarrow A$. A may represent either the name of an action in the world or an internal predicate. C represents the necessary condition for rule activation in the form of a conjunction over the *relevant* attributes of the world state. G^R represents the expected effect of the action. For example, $x_1 \wedge \bar{x}_2 \wedge g_2 \rightarrow$ turn_on_x2 indicates that when x_1 is on and x_2 is off, this operator is expected to turn x_2 on.

An instantiated rule is assigned a *rank* composed of:

- priority rating
- level of experience with rule
- confidence in current parameter bindings

The first component, priority rating, is an inductively acquired measure of the rule's performance on previous instances. The second component modulates the priority rating with respects to a frequency of use measure. The third component captures any uncertainty inherent in the underlying features serving as parameters to the rule.

Each time a new rule is added to the rule base, an attempt is made to combine it with similar existing rules to produce more general rules having a wider relevance and applicability.

Given a rule $c_a \wedge c_b \wedge g_x^R \wedge g_y^R \rightarrow A_1$ covering a set of example instances E_1 and another rule $c_b \wedge c_c \wedge g_y^R \wedge g_z^R \rightarrow A_2$ covering a set of examples E_2, we add a more general rule $c_b \wedge g_y^R \rightarrow A_3$ to the strategy. The new rule A_3 is consistent with E_1 and E_2. In addition it will bind to any state where the literal c_b is active. Therefore the hypothesis represented by the triggering condition is likely an overgeneralization of the target concept. This means that rule A_3 may bind in some states erroneously. However, since all rules that can bind in a state compete to fire in that state, if there is a better rule, then A_3 will be preempted and will not fire.

2.5 Generating Answers

Returning to Figure 1, we note that at the abstract level the process of answer generation begins with the extraction of features active in the current state. These features represent low-level textual attributes and the relations over them described in section 2.3.

Immediately upon reading the current state, the system checks to see if this is a *goal state*. A goal state is a state who's corresponding textual domain representation contains an explicit answer in the right form to match the questions. In the abstract representation, we say that in this state all of the goal constraints are satisfied.

If the current state is indeed a goal state, no further inference is required. The inference process terminates and the actual answer is identified by the matching technique described in section 2.6 and extracted.

If the current state is not a goal state and more processing time is available, QABLe passes the state to the Inference Engine (IE). This module stores strategies in the form of decision lists of rules. For a given state, each strategy may recommend at most one rule to execute. For each strategy this is the first rule in its decision list to fire. The IE selects the rule among these with the highest relative rank, and recommends it as the next transformation rule to be applied to the current state.

If a valid rule exists it is executed in the domain. This modifies the concrete textual layer. At this point, the pre-processing and feature extraction stages are invoked, a new current state is produced, and the inference cycle begins anew.

If a valid rule cannot be recommend by the IE, QABLe passes the current state to the Search Engine (SE). The SE uses the current state and its set of primitive operators to instantiate a new rule, as described in section 2.4. This rule is then executed in the domain, and another iteration of the process begins.

If no more primitive operators remain to be applied to the current state, the SE cannot instantiate a new rule. At this point, search for the goal state cannot proceed, processing terminates, and QABLe returns failure.

Instantiate Rule
Given:

- set of primitive operators
- current state specification
- goal specification

1. select primitive operator to instantiate
2. bind active state variables & goal spec to existentially quantified condition variables
3. execute action in domain
4. update expected effect of new rule according to change in state variable values

Figure 2. Procedure for instantiating transformation rules using primitive operators.

System	who	what	when	where	why	Overall
Deep Read	48%	38%	37%	39%	21%	36%
Quarc	41%	28%	55%	47%	28%	40%
Brown	57%	32%	32%	50%	22%	41%
QABLe-N/L	48%	35%	52%	43%	28%	41%
QABLe-L	56%	41%	56%	45%	35%	47%
QABLe-L+	59%	43%	56%	46%	36%	48%

Table 2. Comparison of QA accuracy by question type.

System	# rules learned	# rules on solution path	average # rules per correct answer
QABLe-L	3,463	426	3.02
QABLe-L+	16,681	411	2.85

Table 3. Analysis of transformation rule learning and use.

When the system is in the training phase and the SE instantiates a new rule, that rule is generalized against the existing rule base. This procedure attempts to create more general rules that can be applied to unseen example instances.

Once the inference/search process terminates (successfully or not), a reinforcement learning algorithm is applied to the entire rule search-inference tree. Specifically, rules on the solution path receive positive reward, and rules that fired, but are not on the solution path receive negative reinforcement.

2.6 Candidate Answer Matching and Extraction

As discussed in the previous section, when a goal state is generated in the abstract representation, this corresponds to a textual domain representation that contains an explicit answer in the right form to match the questions. Such a candidate answer may be present in the original text, or may be generated by the inference/search process. In either case, the answer-containing sentence must be found, and the actual answer extracted. This is accomplished by the Answer Matching and Extraction procedure.

The first step in this procedure is to reformulate the question into a statement form. This results in a sentence containing an empty slot for the information being queried. Recall further that QABLe's pre-processing stage analyzes text with respect to various syntactic and semantic types. In addition to supporting abstract feature generation, these tags can be used to analyze text on a lexical level. The goal now is to find a sentence whose syntactic and semantic analysis matches that of the reformulated question's as closely as possible.

3 Experimental Evaluation

3.1 Experimental Setup

We evaluate our approach to open-domain natural language question answering on the Remedia corpus. This is a collection of 115 children's stories provided by Remedia Publications for reading comprehension. The comprehension of each story is tested by answering five *who*, *what*, *where*, and *why* questions.

The Remedia Corpus was initially used to evaluate the Deep Read reading comprehension system, and later also other systems, including Quarc and the Brown University statistical language processing class project.

The corpus includes two answer keys. The first answer key contains annotations indicating the story sentence that is lexically closest to the answer found in the published answer key (AutSent). The second answer key contains sentences that a human judged to best answer each question (HumSent). Examination of the two keys shows the latter to be more reliable. We trained and tested using the HumSent answers. We also compare our results to the HumSent results of prior systems. In the Remedia corpus, approximately 10% of the questions lack an answer. Following prior work, only questions with annotated answers were considered.

We divided the Remedia corpus into a set of 55 tests used for development, and 60 tests used to evaluate our model, employing the same partition scheme as followed by the prior work mentioned above. With five questions being supplied with each test, this breakdown provided 275 example instances for training, and 300 example instances to test with. However, due to the heavy reliance of our model on learning, many more training examples were necessary. We widened the training set by adding story-question-answer sets obtained from several online sources. With the extended corpus, QABLe was trained on 262 stories with 3-5 questions each, corresponding to 1000 example instances.

3.2 Discussion of Results

Table 2 compares the performance of different versions of QABLe with those reported by the three systems described above. We wish to discern the particular contribution of transformation rule learning in the QABLe model, as well as the value of expanding the training set. Thus, the QABLe-N/L results indicate the accuracy of answers returned by the QA matching and extraction algorithm described in section 2.6 only. This algorithm is similar to prior answer extraction techniques, and provides a baseline for our experiments. The QABLe-L results include answers returned by the full QABLe framework, including the utilization of learned transformation rules, but trained only on the limited training portion of the Remedia corpus. The QABLe-L+ results are for the version trained on the expanded training set.

As expected, the accuracy of QABLe-N/L is comparable to those of the earlier systems. The Remedia-only training set version, QABLe-L, shows an improvement over both the baseline QABLe, and most of the prior system results. This is due to its expanded ability to deal with semantic alternations in the narrative by finding and learning transformation rules that reformulate the alternations into a lexical form matching that of the question.

The results of QABLe-L+, trained on the expanded training set, are for the most part noticeably better than those of QABLe-L. This is because training on more example instances leads to wider domain coverage through the acquisition of more transformation rules. Table 3 gives a break-down of rule learning and use for the two learning versions of QABLe. The first column is the total number of rules learned by each system version. The second column is the number of rules that ended up being successfully used in generating an answer. The third column gives the average number of rules each system needed to answer an answer (where a correct answer was generated). Note that QABLe-L+ used fewer rules on average to generate more correct answers than QABLe-L. This is because QABLe-L+ had more opportunities to refine its policy controlling rule firing through reinforcement and generalization.

Note that the learning versions of QABLe do significantly better than the QABLe-N/L and all the prior systems on *why*-type questions. This is because many of these questions require an inference step, or the combination of information spanning multiple sentences. QABLe-L and QABLe-L+ are able to successfully learn transformation rules to deal with a subset of these cases.

4 Conclusion

This paper present an approach to automatically learn strategies for natural language questions answering from examples composed of textual sources, questions, and corresponding answers. The strategies thus acquired are composed of ranked lists transformation rules that when applied to an initial state consisting of an unseen text and question, can derive the required answer. The model was shown to outperform three prior systems on a standard story comprehension corpus.

References

E. Brill. Transformation-based error driven learning and natural language processing: A case study in part of speech tagging. In *Computational Linguistics*, 21(4):543-565, 1995.

Charniak, Y. Altun, R. de Salvo Braz, B. Garrett, M. Kosmala, T. Moscovich, L. Pang, C. Pyo, Y. Sun, W. Wy, Z. Yang, S. Zeller, and L. Zorn. Reading comprehension programs in a statistical-language-processing class. *ANLP/NAACL-00*, 2000.

C. Cumby and D. Roth. Relational representations that facilitate learning. *KR-00*, pp. 425-434, 2000.

Y. Even-Zohar and D. Roth. A classification approach to word prediction. *NAACL-00*, pp. 124-131, 2000.

C. Fellbaum (ed.) *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

L. Hirschman and R. Gaizauskas. Natural language question answering: The view from here. *Natural Language Engineering*, 7(4):275-300, 2001.

L. Hirschman, M. Light, and J. Burger. Deep Read: A reading comprehension system. *ACL-99*, 1999.

L. P. Kaebling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *J. Artif. Intel. Research*, 4:237-285, 1996.

R. Khardon, D. Roth, and L. G. Valiant. Relational learning for nlp using linear threshold elements, *IJCAI-99*, 1999.

R. Khardon. Learning to take action. *Machine Learning* 35(1), 1999.

E. Riloff and M. Thelen. A rule-based question answering system for reading comprehension tests. *ANLP/NAACL-2000*, 2000.

P. Tadepalli and B. Natarajan. A formal framework for speedup learning from problems and solutions. *J. Artif. Intel. Research*, 4:445-475, 1996.

E. M. Voorhees Overview of the TREC 2003 question answering track. *TREC-12*, 2003.

Dependency-Based Statistical Machine Translation

Heidi J. Fox
Brown Laboratory for Linguistic Information Processing
Brown University, Box 1910, Providence, RI 02912
hjf@cs.brown.edu

Abstract

We present a Czech-English statistical machine translation system which performs tree-to-tree translation of dependency structures. The only bilingual resource required is a sentence-aligned parallel corpus. All other resources are monolingual. We also refer to an evaluation method and plan to compare our system's output with a benchmark system.

1 Introduction

The goal of statistical machine translation (SMT) is to develop mathematical models of the translation process whose parameters can be automatically estimated from a parallel corpus. Given a string of foreign words **F**, we seek to find the English string **E** which is a "correct" translation of the foreign string. The first work on SMT done at IBM (Brown et al., 1990; Brown et al., 1992; Brown et al., 1993; Berger et al., 1994), used a noisy-channel model, resulting in what Brown et al. (1993) call "the Fundamental Equation of Machine Translation":

$$\hat{\mathbf{E}} = \overset{\text{argmax}}{\mathbf{E}} P(\mathbf{E})P(\mathbf{F} \mid \mathbf{E}) \qquad (1)$$

In this equation we see that the translation problem is factored into two subproblems. $P(\mathbf{E})$ is the *language model* and $P(\mathbf{F} \mid \mathbf{E})$ is the *translation model*. The work described here focuses on developing improvements to the translation model.

While the IBM work was groundbreaking, it was also deficient in several ways. Their model translates words in isolation, and the component which accounts for word order differences between languages is based on linear position in the sentence. Conspicuously absent is all but the most elementary use of syntactic information. Several researchers have subsequently formulated models which incorporate the intuition that syntactically close constituents tend to stay close across languages. Below are descriptions of some of these different methods of integrating syntax.

- Stochastic Inversion Transduction Grammars (Wu and Wong, 1998): This formalism uses a grammar for English and from it derives a possible grammar for the foreign language. This derivation includes adding productions where the order of the RHS is reversed from the ordering of the English.

- Syntax-based Statistical Translation (Yamada and Knight, 2001): This model extends the above by allowing all possible permutations of the RHS of the English rules.

- Statistical Phrase-based Translation (Koehn et al., 2003): Here "phrase-based" means "subsequence-based", as there is no guarantee that the phrases learned by the model will have any relation to what we would think of as syntactic phrases.

- Dependency-based Translation (Čmejrek et al., 2003): This model assumes a dependency parser for the foreign language. The syntactic structure and labels are preserved during translation. Transfer is purely lexical. A generator builds an English sentence out of the structure, labels, and translated words.

Proceedings of the ACL Student Research Workshop, pages 91–96,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

2 System Overview

The basic framework of our system is quite similar to that of Čmejrek et al. (2003) (we reuse many of their ancillary modules). The difference is in how we use the dependency structures. Čmejrek et al. only translate the lexical items. The dependency structure and any features on the nodes are preserved and all other processing is left to the generator. In addition to lexical translation, our system models structural changes and changes to feature values, for although dependency structures are fairly well preserved across languages (Fox, 2002), there are certainly many instances where the structure must be modified.

While the entire translation system is too large to discuss in detail here, I will provide brief descriptions of ancillary components. References are provided, where available, for those who want more information.

2.1 Corpus Preparation

Our parallel Czech-English corpus is comprised of Wall Street Journal articles from 1989. The English data is from the University of Pennsylvania Treebank (Marcus et al., 1993; Marcus et al., 1994). The Czech translations of these articles are provided as part of the Prague Dependency Treebank (PDT) (Böhmová et al., 2001). In order to learn the parameters for our model, we must first create aligned dependency structures for the sentence pairs in our corpus. This process begins with the building of dependency structures.

Since Czech is a highly inflected language, morphological tagging is extremely helpful for downstream processing. We generate the tags using the system described in (Hajič and Hladká, 1998). The tagged sentences are parsed by the Charniak parser, this time trained on Czech data from the PDT. The resulting phrase structures are converted to tectogrammatical dependency structures via the procedure documented in (Böhmová, 2001). Under this formalism, function words are deleted and any information contained in them is preserved in features attached to the remaining nodes. Finally, functors (such as agent or patient) are automatically assigned to nodes in the tree (Žabokrtský et al., 2002).

On the English side, the process is simpler. We parse with the Charniak parser (Charniak, 2000) and convert the resulting phrase-structure trees to a function-argument formalism, which, like the tectogrammatic formalism, removes function words. This conversion is accomplished via deterministic application of approximately 20 rules.

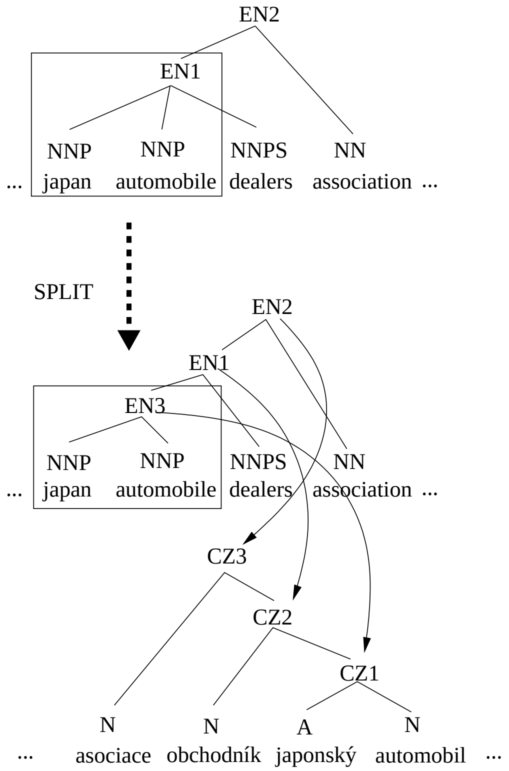


Figure 1: Left SPLIT Example

2.2 Aligning the Dependency Structures

We now generate the alignments between the pairs of dependency structures we have created. We begin by producing word alignments with a model very similar to that of IBM Model 4 (Brown et al., 1993). We keep fifty possible alignments and require that each word has at least two possible alignments. We then align phrases based on the alignments of the words in each phrase span. If there is no satisfactory alignment, then we allow for structural mutations. The probabilities for these mutations are refined via another round of alignment. The structural mutations allowed are described below. Examples are shown in phrase-structure format rather than dependency format for ease of explanation.

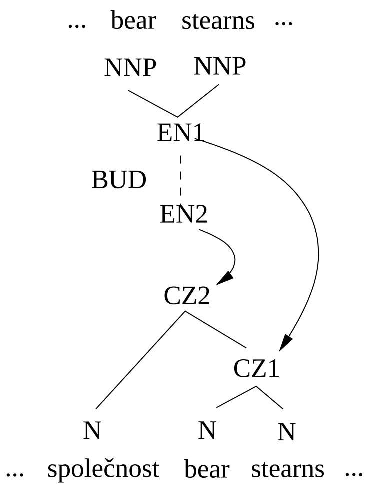


Figure 2: BUD Example

- KEEP: No change. This is the default.

- SPLIT: One English phrase aligns with two Czech phrases and splitting the English phrase results in a better alignment. There are three types of split (left, right, middle) whose probabilities are also estimated. In the original structure of Figure 1, English node EN1 would align with Czech nodes CZ1 and CZ2. Splitting the English by adding child node EN3 results in a better alignment.

- BUD: This adds a unary level in the English tree in the case when one English node aligns to two Czech nodes, but one of the Czech nodes is the parent of the other. In Figure 2, the Czech has one extra word "společnost" ("company") compared with the English. English node EN1 would normally align to both CZ1 and CZ2. Adding a unary node EN2 to the English results in a better alignment.

- ERASE: There is no corresponding Czech node for the English one. In Figure 3, the English has two nodes, EN1 and EN2, which have no corresponding Czech nodes. Erasing them brings the Czech and English structures into alignment.

- PHRASE-TO-WORD: An entire English phrase aligns with one Czech word. This operates similarly to ERASE.

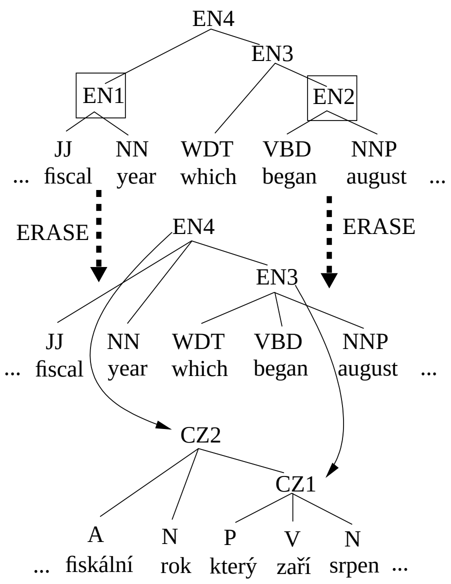

EN4
EN3
EN1
EN2
JJ NN WDT VBD NNP
... fiscal year which began august ...
ERASE
ERASE
EN4
EN3
JJ NN WDT VBD NNP
... fiscal year which began august ...
CZ2
CZ1
A N P V N
... fiskální rok který září srpen ...

Figure 3: ERASE Example

3 Translation Model

Given $\mathcal{E}$, the parse of the English string, our translation model can be formalized as $P(F \mid \mathcal{E})$. Let $\mathcal{E}_1 \ldots \mathcal{E}_n$ be the nodes in the English parse, $\mathcal{F}$ be a parse of the Czech string, and $\mathcal{F}_1 \ldots \mathcal{F}_m$ be the nodes in the Czech parse. Then,

$$P(F \mid \mathcal{E}) = \sum_{\mathcal{F} for F} P(\mathcal{F}_1 \ldots \mathcal{F}_m \mid \mathcal{E}_1 \ldots \mathcal{E}_n) \quad (2)$$

We initially make several strong independence assumptions which we hope to eventually weaken. The first is that each Czech parse node is generated independently of every other one. Further, we specify that each English parse node generates exactly one (possibly NULL) Czech parse node.

$$P(\mathcal{F} \mid \mathcal{E}) = \prod_{\mathcal{F}_i \in \mathcal{F}} P(\mathcal{F}_i \mid \mathcal{E}_1 \ldots \mathcal{E}_n) = \prod_{i=1}^{n} P(\mathcal{F}_i \mid \mathcal{E}_i) \quad (3)$$

An English parse node $\mathcal{E}_i$ contains the following information:

- An English word: e_i

- A part of speech: t_i^e

- A vector of n features (e.g. negation or tense): $< \phi_i^e[1], \ldots, \phi_i^e[n] >$

- A list of dependent nodes

In order to produce a Czech parse node $\mathcal{F}_i$, we must generate the following:

Lemma f_i: We generate the Czech lemma f_i dependent only on the English word e_i.

Part of Speech t_i^f: We generate Czech part of speech t_i^f dependent on the part of speech of the Czech parent $t_{par(i)}^f$ and the corresponding English part of speech t_i^e.

Features $< \phi_i^f[1], \ldots, \phi_i^f[n] >$: There are several features (see Table 1) associated with each parse node. Of these, all except IND are typical morphological and analytical features. IND (indicator) is a loosely-specified feature comprised of functors, where assigned, and other words or small phrases (often prepositions) which are attached to the node and indicate something about the node's function in the sentence. (e.g. an IND of "at" could indicate a locative function). We generate each Czech feature $\phi_i^f[j]$ dependent only on its corresponding English feature $\phi_i^e[j]$.

Head Position h_i: When an English word is aligned to the head of a Czech phrase, the English word is typically also the head of its respective phrase. But, this is not always the case, so we model the probability that the English head will be aligned to either the Czech head or to one of its children. To simplify, we set the probability for each particular child being the head to be uniform in the number of children. The head position is generated independent of the rest of the sentence.

Structural Mutation m_i: Dependency structures are fairly well preserved across languages, but there are cases when the structures need to be modified. Section 2.2 contains descriptions of the different structural changes which we model. The mutation type is generated independent of the rest of the sentence.

Feature	Description
NEG	Negation
STY	Style (e.g. statement, question)
QUO	Is node part of a quoted expression?
MD	Modal verb associated with node
TEN	Tense (past, present, future)
MOOD	Mood (infinitive, perfect, progressive)
CONJ	Is node part of a conjoined expression?
IND	Indicator

Table 1: Features

3.1 Model with Independence Assumptions

With all of the independence assumptions described above, the translation model becomes:

$$P(\mathcal{F}_i \mid \mathcal{E}_i) = P(f_i \mid e_i) P(t_i^f \mid t_i^e, t_{par(i)}^f) \times P(h_i) P(m_i) \prod_{j=1}^{n} P(\phi_i^f[j] \mid \phi_i^e[j]) \quad (4)$$

4 Training

The Czech and English data are preprocessed (see Section 2.1) and the resulting dependency structures are aligned (see Section 2.2). We estimate the model parameters from this aligned data by maximum likelihood estimation. In addition, we gather the inverse probabilities $P(E \mid F)$ for use in the figure of merit which guides the decoder's search.

5 Decoding

Given a Czech sentence to translate, we first process it as described in Section 2.1. The resulting dependency structure is the input to the decoder. The decoder itself is a best-first decoder whose priority queue holds partially-constructed English nodes.

For our figure of merit to guide the search, we use the probability $P(E \mid F)$. We normalize this using the *perplexity* (2^H) to compensate for the different number of possible values for the features $\phi[j]$. Given two different features whose values have the same probability, the figure of merit for the feature with the greater uncertainty will be boosted. This prevents features with few possible values from monopolizing the search at the expense of the other features. Thus, for feature $\phi_i^e[j]$, the figure of merit is

$$FOM = P(\phi_i^e[j] \mid \phi_i^f[j]) \times 2^{H(\Phi_i^e[j] \mid \phi_i^f[j])} \quad (5)$$

Since our goal is to build a forest of partial translations, we translate each Czech dependency node independently of the others. (As more conditioning factors are added in the future, we will instead translate small subtrees rather than single nodes.) Each translated node $\mathcal{E}_i$ is constructed incrementally in the following order:

1. Choose the head position h_i
2. Generate the part of speech t_i^e
3. For $j = 1..n$, generate $\phi_i^e[j]$
4. Choose a structural mutation m_i

English nodes continue to be generated until either the queue or some other stopping condition is reached (e.g. having a certain number of possible translations for each Czech node). After stopping, we are left with a forest of English dependency nodes or subtrees.

6 Language Model

We use a syntax-based language model which was originally developed for use in speech recognition (Charniak, 2001) and later adapted to work with a syntax-based machine translation system (Charniak et al., 2001). This language model requires a forest of partial phrase structures as input. Therefore, the format of the output of the decoder must be changed. This is the inverse transformation of the one performed during corpus preparation. We accomplish this with a statistical tree transformation model whose parameters are estimated during the corpus preparation phase.

7 Evaluation

We propose to evaluate system performance with version 0.9 of the NIST automated scorer (NIST, 2002), which is a modification of the BLEU system (Papineni et al., 2001). BLEU calculates a score based on a weighted sum of the counts of matching n-grams, along with a penalty for a significant difference in length between the system output and the reference translation closest in length. Experiments have shown a high degree of correlation between BLEU score and the translation quality judgments of humans. The most interesting difference in the NIST scorer is that it weights n-grams based on a notion of informativeness. Details of the scorer can be found in their paper.

For our experiments, we propose to use the data from the PDT, which has already been segmented into training, held out (or development test), and evaluation sets. As a baseline, we will run the GIZA++ implementation of IBM's Model 4 translation algorithm under the same training conditions as our own system (Al-Onaizan et al., 1999; Och and Ney, 2000; Germann et al., 2001).

8 Future Work

Our first priority is to complete the final pieces so that we have an end-to-end system to experiment with. Once we are able to evaluate our system output, our first priority will be to analyze the system errors and adjust the model accordingly. We recognize that our independence assumptions are generally too strong, and improving them is a hight priority. Adding more conditioning factors should improve the quality of the decoder output as well as reducing the amount of probability mass lost on structures which are not well formed. With this will come sparse data issues, so it will also be important for us to incorporate smoothing into the model.

There are many interesting subproblems which deserve attention and we hope to examine at least a couple of these in the near future. Among these are discontinuous constituents, head switching, phrasal translation, English word stemming, and improved modeling of structural changes.

Acknowledgments

This work was supported in part by NSF grant IGERT-9870676. We would like to thank Jan Hajič, Martin Čmejrek, Jan Cuřín for all of their assistance.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical machine translation: Final report, JHU workshop 1999. www.clsp.jhu.edu/ws99/projects/mt/final_report/mt-final-report.ps.

Adam L. Berger, Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, John R. Gillett, John D. Lafferty, Robert L. Mercer, Harry Printz, and Luboš Ureš. 1994. The Candide system for machine translation. In *Proceedings of the ARPA Human Language Technology Workshop*.

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2001. The Prague Dependency Treebank: Three-level annotation scenario. In Anne Abeillé, editor, *Treebanks: Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers.

Alena Böhmová. 2001. Automatic procedures in tectogrammatical tagging. *The Prague Bulletin of Mathematical Linguistics*, 76.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Peter F. Brown, Stephen A. Della Petra, Vincent J. Della Pietra, John D. Lafferty, and Robert L. Mercer. 1992. Analysis, statistical transfer, and synthesis in machine translation. In *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 83–100.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, June.

Eugene Charniak, Kevin Knight, and Kenji Yamada. 2001. Syntax-based language models for statistical machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France, July.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 116–123, Toulouse, France, July.

Martin Čmejrek, Jan Cuřín, and Jiří Havelka. 2003. Czech-English Dependency-based Machine Translation. In *EACL 2003 Proceedings of the Conference*, pages 83–90, April 12–17, 2003.

Heidi Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, July.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*.

Jan Hajič and Barbora Hladká. 1998. Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset. In *Proceedings of COLING-ACL Conference*, pages 483–490, Montreal, Canada.

Philip Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, Edmonton, Canada, May.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 13(2):313–330, June.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 114–119.

NIST. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. www.nist.gov/speech/tests/mt/doc/ngram-study.pdf.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: A method for automatic evaluation of machine translation. Technical report, IBM.

Dekai Wu and Hongsing Wong. 1998. Machine translation with a stochastic grammatical channel. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 1408–1414.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*.

Zdeněk Žabokrtský, Petr Sgall, and Sašo Džeroski. 2002. Machine learning approach to automatic functor assignment in the Prague Dependency Treebank. In *Proceedings of LREC 2002 (Third International Conference on Language Resources and Evaluation)*, volume V, pages 1513–1520, Las Palmas de Gran Canaria, Spain.

Minimalist Parsing of Subjects Displaced from Embedded Clauses in Free Word Order Languages

Asad B. Sayeed
Department of Computer Science
University of Maryland at College Park
A. V. Williams Building
MD 20742 USA
asayeed@mbl.ca

Abstract

In Sayeed and Szpakowicz (2004), we proposed a parser inspired by some aspects of the Minimalist Program. This incremental parser was designed specifically to handle discontinuous constituency phenomena for NPs in Latin. We take a look at the application of this parser to a specific kind of apparent island violation in Latin involving the extraction of constituents, including subjects, from tensed embedded clauses. We make use of ideas about the left periphery from Rizzi (1997) to modify our parser in order to handle apparently violated subject islands and similar phenomena.

1 Introduction

In Sayeed and Szpakowicz (2004), we started by describing the difficulty of parsing sentences in languages with discontinuous constituency in a syntactically robust and cognitively realistic manner. We made the assumption that semantic links between the words of a sentence are made as soon as they arrive; we noted that this constrains the kinds of formalisms and algorithms that could be used to parse human sentences. In the spirit of the Minimalist Programme, we would like to produce the most economical parsing process, where, potentially controversially, we characterize economy as computational complexity. Discontinuity of phrases (usually noun phrases) in e.g. Latin provides a specific set of challenges in the development of a robust syntactic analysis; for instance, in the process of building parse trees, nouns must often be committed to positions in particular structures prior to the arrival of adjectives in an incremental parsing environment.

Inspired by work such as Stabler (2001), we proposed a formalism and algorithm[1] that used feature set unification rather than feature cancellation, which Stabler uses to implement basic Minimalist operations such as MOVE and MERGE. We demonstrated the workings of the algorithm given simple declarative sentences—in other words, within a single, simple clause. What we wish to do now is demonstrate that our algorithm parses Latin sentences with embedded clauses, and in particular those with constituents displaced beyond the boundaries of embedded clauses where this displacement does not appear to be legitimate wh-movements; these are, in a sense, another form of discontinuity. In doing this, we hope to show that our formalism works for a wider subset of the Latin language, and that we have reduced the problem of developing a grammar to one of choosing the correct features.

2 Background

Noun phrases in Latin can become discontinuous within clauses. For instance, it is possible to place a noun before a verb and an adjective that agrees with the noun after the verb. However, for the most part, the noun phrase components stay within CP. Nevertheless, Kessler (1995) noted several instances where, possibly for intonational effect, Latin prose writers extracted items into matrix clauses from embedded clauses and clauses embedded within those embedded clauses. For example,

(1) Tametsi *tu*
Although you-NOM-SG
scio *quam*
know-IND-PRES-1SG how

[1] For the purpose of clarification, our algorithm can be found at http://www.umiacs.umd.edu/∼asayeed/discont.pdf

Proceedings of the ACL Student Research Workshop, pages 97–102,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

sis	*curiosus*
are-SUBJ-PRES-2SG	interested-NOM-SG

‘Although I know how interested you are’ (Caelius at Cicero, Fam 8.1.1)

In this and other cases provided by Kessler, a word is extracted from an embedded clause and moved to the beginning of the matrix clause. (The italicized words consist of the extracted element and the clause from which it was extracted.) Note in particular that 1 involves the dislocation of the subject from a tensed embedded clause, something that would ordinarily be a well-known island violation (Haegeman, 1994).

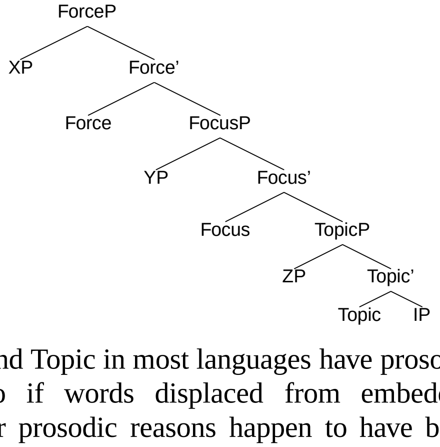
According to Kessler, this situation is rare enough that many contemporary accounts of Latin syntax neglect discussion of this kind of device. It is likely that Cicero occasionally wrote this way for prosodic reasons; however, there is no reason why prosody should not have syntactic consequences, and we attempt to account for the parsing of such sentences in this document.

It is interesting to note how in these examples, the displaced element moves somewhere near to the beginning of the outer clause. Rizzi (1997) suggests a structure for this “left periphery” based on observations from Italian:

(2) . . . Force . . . (Focus) . . . (Topic) . . .

Within Rizzi’s GB-based framework, this is suggested to be the internal structure of CP. In X-bar terms, it looks something like this:

(3)



Focus and Topic in most languages have prosodic effects, so if words displaced from embedded clauses for prosodic reasons happen to have been raised to the beginning, it suggests that the word has become part of some form of articulated CP structure.

Since our parsing algorithm is inspired by minimalism, we cannot make use of the full X-bar system. Instead, we use Rizzi’s analysis to develop an analysis based on features and checking.

3 The Parser in Action

3.1 A Run-through

Our parser (2004) is incremental, meaning that it does not have access to the end of the sentence at the beginning of a derivation. It is also “semantically greedy”, meaning that it attempts to satisfy the semantic requirements (through checking) as soon as possible. So each step in the derivation consists of attempting to see whether or not checking can be accomplished using the current items in the “processing buffer” and those in the “input queue,” and if not, shifting a word from the input queue onto the processing buffer. The distinction is marked, in our notation, by a |: the words and trees before | are in the processing buffer, and those that are after | are in the input queue.

The algorithm also prefers move before merge. This also ensures that trees do not have multiple pending resolvable semantic dependencies, which can represent a state of ambiguity in determining which dependency to resolve and how.

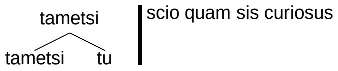
We will now present an example parse of the above sentence. But we will first present the general outline of the parse, rather than the full details using the formal representation; after that, we will demonstrate the formalism. We sketch the steps of the parse first so that we can deduce what features we would need to make it work with the system.

We first start with everything in the input queue, after the |:

(4) |tametsi tu scio quam sis curiosus

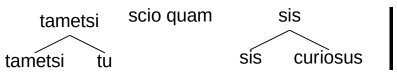
Now we need to shift (hear) two words for any parsing operations to be performed. So we shift *tametsi* and *tu*. *tametsi* (“although”) consists of *tamen*, *et*, and *si*: “nevertheless”, “and”, and “if.” These suggest that *tametsi* is part of a CP, and, most likely, Force. Since *tu* has been displaced from the embedded clause, probably for prosodic reasons, it likely has features that can be gleaned from the intonation and the context, such as Focus. Since these are part of our CP system, we merge them.

(5)



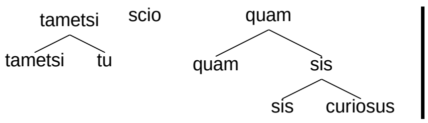
Now we have to shift *scio*. But the verb *scio* does not have a complement and cannot merge with *tametsi* until it is a complete VP. The same is true for *quam* ("how") and *sis* since *sis* ("you are") needs a complement: *curiosus*. So the system waits to shift everything and then merges *sis* and *curiosus*.

(6)



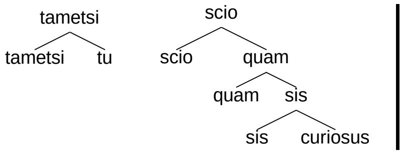
Now we can merge *sis* and *quam*, since *sis* now has a complement. Latin is a pro-drop language, so we can perform the merge without having an explicit subject, which is currently part of another tree.

(7)



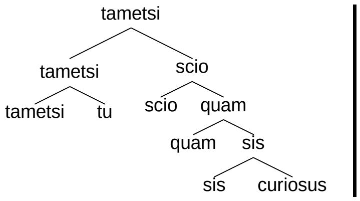
quam has been given its complement. Now as a complete CP, it is ready to be a complement of *scio*.

(8)



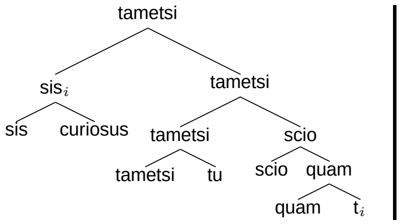
We have a CP (the *tametsi* tree) and a VP (*scio*), and we need to merge them to form one CP.

(9)



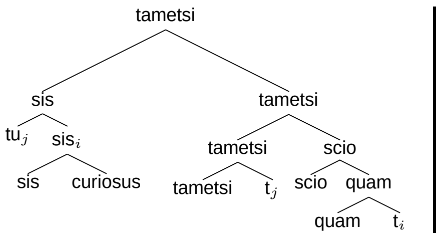
So this leaves us in the position of having a *tu* and *sis* in one tree. However, we cannot bring them together. In Sayeed and Szpakowicz (2004), we required (in order to limit tree searches) that movement during parsing be to positions that command the trace of movement. Clearly, *tu* does not command *sis*. We only permitted raising, so what should we raise? If we raised the entire CP, we would get a tree in which neither *tu* nor *sis* commands the other. We would have to make another move to get *sis* to command *tu*. So we take a simpler route and just move *sis*.

(10)



Now *sis* commands *tu*. We can now move *tu*.

(11)



Note that *sis* still projects after the merge, seeing that *sis* holds the requirement for a subject—*tu* is now in what would be known as a specifier position. It does not matter that *tu* does not presently command its trace; this is something in our account of parsing that differs from GB and minimalist accounts of movement in generation. Instead, the position with which it must be merged after movement can be the one that commands the original position. This allows the target position to be the one that projects, as *sis* has.

3.2 Now with Features

Now all dependencies are satisfied, and we have a complete tree. What we need to accomplish next is an account of the features required for this parse under the system in Sayeed and Szpakowicz (2004). We add one extra characteristic to Sayeed and Szpakowicz (2004) which we will explain in greater detail in forthcoming work: optionally-checked features; this is required primarily to avoid having to imagine empty categories when parsing such phenomena as dropped subjects, which exists in Latin.

First of all, let us account for the lexical entries of the initial two words, *tametsi* and *tu*. We need features that represent the discursive effect represented by the displacement of *tu*. We shall assume that this is Focus. Also, however, we need a feature that will prepare *tametsi* to merge with *scio*. So we represent these two as

(12) tametsi: {UNCH?(Disc:Focus), UNCH(Type:V)}
tu: {unch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}

Features are grouped together into feature bundles, which allow simultaneous checking of features. Note that the ? in one of the feature bundles of *tametsi* means that it is optional; it does not have to be checked with a focus feature on an adjacent constituent if such a feature does not exist, but it must if there is one.

For *tu* we are using feature paths as we defined in Sayeed and Szpakowicz (2004); what is to the right of a feature path cannot be checked before what is to the left. In this case, we must check the focus feature before we can check *tu* as a constituent of its proper VP (headed by *sis*).

We express the trees using the same horizontal indented representation as in Sayeed and Szpakowicz (2004). We use this notation because the nodes of this tree are too large for the "normal" tree representation used above. So we start with

```
(13)   | tametsi tu scio quam sis curiosus
```

We need to shift two words before we can do anything. We thus create nodes with the above features.

```
(14)   [tametsi {UNCH(Disc:Focus), UNCH(Type:V)}]
       [tu {unch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}]
       | scio quam sis curiosus
```

The Focus features can be checked. Using our system, unch and UNCH feature bundles are compatible for checking, and the node with the UNCH feature projects. This form of merge among the items already shifted can only be performed with the roots of adjacent trees. We specified this to prevent long-distance searches of the processing buffer.

```
(15)   [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
                UNCH(Type:V)}]
          tametsi
          [tu {ch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}]
       | scio quam sis curiosus
```

When UNCH and unch features bundles are checked, their features are unified (and replaced with the result of unification). UNCH and unch become CH and ch. Meanwhile, *tametsi* has acquired the features of *tu* in the CH bundle. The purpose of this mechanism is to transfer information up the tree in order to support incremental parsing of discontinuous NP constituents, but we find an additional use for this below.

We make one change here to the unification of feature bundles as described by Sayeed and Szpakowicz (2004): when we replace feature bundles with the result of unification, we replace them with the features of the entire path with which we are checking. This ensures that in the process of checking, we do not "hide" features that are further on in the path. So *tametsi* also gains the gender, person, and case features. This is actually quite a logical extension of the idea we expressed in Sayeed and Szpakowicz (2004) that a feature being checked with a feature further down a path should be compatible with all the previous features on the path. In both cases, the system should reflect the idea that features further down a path are dependent on the checking status of previous features. As with unification in general, compatibility means lack of a conflict in $\tau : \phi$ pairs (i.e., no case conflicts, and so on).

Now, as per 6, we need to shift all the remaining words into the buffer before we get a compatible set. So we need to determine lexical entries for all of the remaining words. First, *scio*:

```
(16)   scio: {UNCH?(Case:Nom, Pers:1, Num:Sg),
              UNCH(Wh:0) → unch(Type:V)}
```

We once again use a feature path. In this case, it means that *scio* ("know") must have a wh-phrase complement[2] before it is ready to be checked by something that takes a VP complement (such as a complementizer). So this leads us to an entry for *quam*:

```
(17)   quam: {UNCH?(Disc:Focus), UNCH(Type:V) → unch(Wh:0)}
```

For *quam*, we also have an optional Focus feature, because it is the head of a CP as *tametsi* is above. (We might have other optional discourse features there, but they would be superfluous for this discussion.) And, like *tametsi*, it has a feature that allows it to take a VP complement. Checking this feature releases the wh-feature that allows it to become the complement of *scio*.

Now we only need entries for *sis* and *curiosus*

```
(18)   sis: {UNCH?(Case:Nom, Pers:2, Num:Sg),
             UNCH(Case:Acc) → unch(Type:V)}
       curiosus: unch(Case:Acc, Gen:Masc, Num:Sg)
```

We use an optional feature for the requirement of a nominative subject on *sis*, subjects being optional in Latin. However, we do require it to take an accusative object. We are able to shift everything as we did prior to 6.

```
(19)   [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
                UNCH(Type:V)}]
          tametsi
          [tu {ch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}]
       [scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
              UNCH(Wh:0) → unch(Type:V)}]
       [quam {UNCH?(Disc:Focus), UNCH(Type:V) → unch(Wh:0)}]
       [sis {UNCH?(Case:Nom, Pers:2, Num:Sg),
             UNCH(Case:Acc) → unch(Type:V)}]
       [curiosus unch(Case:Acc, Gen:Masc, Num:Sg)] |
```

Now *sis* and *curiosus* can merge. The resulting merger between compatible unch and UNCH features, by Sayeed and Szpakowicz (2004), also causes the contents of those feature bundles to be unified.

```
(20)   [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
                UNCH(Type:V)}]
          tametsi
          [tu {ch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}]
```

[2] The 0 is just a placeholder meaning that the Wh is a singleton, not a pair like many of the other features.

```
[scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
     UNCH(Wh:0) → unch(Type:V)]
[quam {UNCH?(Disc:Focus), UNCH(Type:V) → unch(Wh:0)}]
[sis {UNCH?(Case:Nom, Pers:2, Num:Sg),
    CH(Case:Acc, Gen:Masc, NumSg) → unch(Type:V)}]
  sis
  [curiosus ch(Case:Acc, Gen:Masc, Num:Sg)] |
```

Now that the left feature on the feature path on *sis* is checked, the verb type feature is free. It can check with the corresponding feature on *quam*.

```
(21) [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
             UNCH(Type:V)}]
       tametsi
       [tu {ch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}]
     [scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
          UNCH(Wh:0) → unch(Type:V)]
     [quam {UNCH?(Disc:Focus), CH(Type:V) → unch(Wh:0)}]
       quam
       [sis {UNCH?(Case:Nom, Pers:2, Num:Sg),
           CH(Case:Acc, Gen:Masc, NumSg) → ch(Type:V)}]
         sis
         [curiosus ch(Case:Acc, Gen:Masc, Num:Sg)] |
```

Feature paths allow *quam* to merge with *scio* as in 8.

```
(22) [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
             UNCH(Type:V)}]
       tametsi
       [tu {ch(Disc:Focus) → unch(Case:Nom, Pers:2, Num:Sg)}]
     [scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
          CH(Wh:0) → unch(Type:V)]
       scio
       [quam {UNCH?(Disc:Focus), CH(Type:V) → ch(Wh:0)}]
         quam
         [sis {UNCH?(Case:Nom, Pers:2, Num:Sg),
             CH(Case:Acc, Gen:Masc, NumSg) → ch(Type:V)}]
           sis
           [curiosus ch(Case:Acc, Gen:Masc, Num:Sg)] |
```

And, lastly, *scio* merges with the CP headed by *tametsi*.

```
(23) [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
             CH(Type:V)}]
       tametsi
         tametsi
         [tu {ch(Disc:Focus)
             → unch(Case:Nom, Pers:2, Num:Sg)}]
       [scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
            CH(Wh:0) → ch(Type:V)]
         scio
         [quam {UNCH?(Disc:Focus), CH(Type:V) → ch(Wh:0)}]
           quam
           [sis {UNCH?(Case:Nom, Pers:2, Num:Sg),
               CH(Case:Acc, Gen:Masc, NumSg)
               → ch(Type:V)}]
             sis
             [curiosus ch(Case:Acc, Gen:Masc, Num:Sg)] |
```

We now have a single tree, but we are in the predicament of 9. We need to be able to move *sis* to a position where it commands *tu*. And that means moving it to join with *tametsi*.

In Sayeed and Szpakowicz (2004), we proposed a mechanism by which adjuncts displaced from discontinuous NPs could reunite with their NPs even if the NP had already been merged as a constituent of a verb. This was by allowing adjuncts to merge with the verb if the verb had a compatible CH feature (without actually checking the adjunct feature bundle). A CH feature advertises that the verb had previously merged with a compatible noun, since unification would have given the noun's features to the CH feature bundle.

In this case, *tametsi* does have a CH feature bundle that appears compatible with *sis*, but UNCH features are not features that cause adjunctions in our system. We propose a minimal stipulation that will solve this problem:

(24) UNCH features (i.e., features that indicate a requirement for a constituent) can be moved or merged to meet compatible CH features.

The main problem with 24 is the possibility that unnecessary movements caused by UNCH features may occur in such a way that the UNCH feature would be moved out of the way of compatible unch features.

But this is likely not a problem. Our system prefers to exhaust all possible movements before mergers in parsing. So, if an UNCH feature had been in the tree, and an unch feature is introduced later at the root (as specified in Sayeed and Szpakowicz (2004)), the constituent containing the UNCH feature would immediately have moved to claim it. Then if a compatible CH feature arrived, it would not matter, since the UNCH feature would itself have been checked. But if a compatible CH feature had been in the tree *before* the compatible unch feature had joined, what then? The constituent containing the UNCH feature would move to join it. Then the unch feature would join the tree. It would still command the UNCH feature, which would move to claim it.

There is only one unsafe case: if the CH feature arrives before the unch feature, and it is part of a head whose constituents contain a compatible unch feature on the *wrong* constituent, then the UNCH feature would be checked with the wrong constituent according to the mechanism above. After all, the UNCH feature would command the incorrect unch feature. This possibility, however, can only exist if there is another displaced item in the tree containing the original CH that is compatible with the UNCH feature but displaced from some *other* phrase. This requires further investigation into Latin grammar, as it seems unlikely that such constructions exist, given the rarity of displacement in the first place.

So let us implement our solution:

```
(25)  [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
                CH(Type:V)}]
        [sis {UNCH?(Case:Nom, Pers:2, Num:Sg),
             CH(Case:Acc, Gen:Masc, NumSg) → ch(Type:V)}]
          sis
          [curiosus ch(Case:Acc, Gen:Masc, Num:Sg)] |
        tametsi
          tametsi
            tametsi
            [tu {ch(Disc:Focus)
                  → unch(Case:Nom, Pers:2, Num:Sg)}]
          [scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
                 CH(Wh:0) → ch(Type:V)]
            scio
            [quam {UNCH?(Disc:Focus), CH(Type:V) → ch(Wh:0)}]
              quam
              <sis>
```

Note that the maximal projections move, not the heads of constituent trees. The maximal projections are the highest node containing the features, and we always take the highest node according to Sayeed and Szpakowicz (2004). Now *sis* commands *tu*. We can move *tu*.

```
(26)  [tametsi {CH(Disc:Focus, Case:Nom, Pers:2, Num:Sg),
                CH(Type:V)}]
        [sis {CH(Case:Nom, Pers:2, Num:Sg),
             CH(Case:Acc, Gen:Masc, NumSg) → ch(Type:V)}]
          [tu {ch(Disc:Focus) → ch(Case:Nom, Pers:2, Num:Sg)}]
          sis
            sis
            [curiosus ch(Case:Acc, Gen:Masc, Num:Sg)] |
        tametsi
          tametsi
            tametsi
            <tu>
          [scio {UNCH?(Case:Nom, Pers:1, Num:Sg),
                 CH(Wh:0) → ch(Type:V)]
            scio
            [quam {UNCH?(Disc:Focus), CH(Type:V)
                   → ch(Wh:0)}]
              quam
              <sis>
```

All optional unchecked features have been eliminated, and the derivation is complete.

4 Conclusions and Future Work

Using the system of Sayeed and Szpakowicz (2004), we have demonstrated a means to parse sentences with constituents extracted from embedded clauses for prosodic reasons in Latin—constituents that appear to be able to escape even subject islands. We were able to maintain the adjacency requirement of our system by making use of discourse features inspired by Rizzi's analysis of the left periphery in Italian in a GB framework. Thus, this highly constrained incremental system was able to parse a sentence with a long-distance displacement.

In order to do it, though, we had to add a stipulation to the system to allow the constituent that required the displaced one to move to a commanding position. We also took no heed to cyclicity in this system, which given the apparent island violation permitted by these constructions, may not seem so bad, especially since the displaced constituent only moves over one CP in the examples we gave. But Kessler finds that there are rare examples where it moves over two CPs. Of course, these cases are even more rare than displacement over a single CP. It could be that the difficulty in violating subjacency is what makes these cases rare, but the checking of the discourse feature that causes the displacement is more important.

One characteristic of our solution and, indeed, Sayeed and Szpakowicz (2004) in general is that in order to maintain incrementality, we do not attempt to return items displaced during generation to their original positions. We still perform only raising, just as in most GB and minimalist accounts of movement. This means that if the constituent of a phrase is higher than its rightful parent in the tree, the lower subtree raises to claim it. In this case, we had to stipulate that constituent subtrees searching for their own constituents could move to intermediate locations as adjuncts, something that Sayeed and Szpakowicz (2004) did not specify. However, we still maintain an essential property of our system: movement happens as soon as possible. This means that the first available compatible intermediate location is sought. It becomes an empirical question, then, whether an intermediate position could ever be a wrong position.

References

Liliane Haegeman. 1994. *Introduction to Government and Binding Theory*. Blackwell, Oxford, 2nd edition.

Brett Kessler. 1995. Discontinuous constituents in latin. http://www.artsci.wustl.edu/∼bkessler/latin-discontinuity/discontinuity.ps.

Luigi Rizzi. 1997. The fine structure of the left periphery. In L. Haegeman, editor, *Elements of Grammar*, pages 281–337. Kluwer, Dordrecht.

Asad Sayeed and Stan Szpakowicz. 2004. Developing a minimalist parser for free word order languages with discontinuous constituency. In José Luis Vicedo, Patricio Martínez-Barco, Rafael Muñoz, and Maximiliano Saiz, editors, *EsTAL—España for Natural Language Processing*. Springer-Verlag.

Edward P. Stabler. 2001. Minimalist grammars and recognition. In Christian Rohrer, Antje Roßdeutscher, and Hans Kamp, editors, *Linguistic Form and its Computation*. CSLI Publications, Stanford.

Centrality Measures in Text Mining: Prediction of Noun Phrases that Appear in Abstracts

Zhuli Xie
Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60607, U. S. A
zxie@cs.uic.edu

Abstract

In this paper, we study different centrality measures being used in predicting noun phrases appearing in the abstracts of scientific articles. Our experimental results show that centrality measures improve the accuracy of the prediction in terms of both precision and recall. We also found that the method of constructing Noun Phrase Network significantly influences the accuracy when using the centrality heuristics itself, but is negligible when it is used together with other text features in decision trees.

1 Introduction

Research on text summarization, information retrieval, and information extraction often faces the question of how to determine which words are more significant than others in text. Normally we only consider content words, i.e., the open class words. Non-content words or stop words, which are called function words in natural language processing, do not convey semantics so that they are excluded although they sometimes appear more frequently than content words. A content word is usually defined as a term, although a term can also be a phrase. Its significance is often indicated by Term Frequency (TF) and Inverse Document Frequency (IDF). The usage of TF comes from "the simple notion that terms which occur frequently in a document may reflect its meaning more strongly than terms that occur less frequently" (Jurafsky and Martin, 2000). On the contrary, IDF assigns smaller weights to terms which are contained in more documents. That is simply because "the more documents having the term, the less useful the term is in discriminating those documents having it from those not having it" (Yu and Meng, 1998).

TF and IDF also find their usage in automatic text summarization. In this circumstance, TF is used individually more often than together with IDF, since the term is not used to distinguish a document from another. Automatic text summarization seeks a way of producing a text which is much shorter than the document(s) to be summarized, and can serve as a surrogate for full-text. Thus, for extractive summaries, i.e., summaries composed of original sentences from the text to be summarized, we try to find those terms which are more likely to be included in the summary.

The overall goal of our research is to build a machine learning framework for automatic text summarization. This framework will learn the relationship between text documents and their corresponding abstracts written by human. At the current stage the framework tries to generate a sentence ranking function and use it to produce extractive summaries. It is important to find a set of features which represent most information in a sentence and hence the machine learning mechanism can work on it to produce a ranking function. The next stage in our research will be to use the framework to generate abstractive summaries, i.e. summaries which do not use sentences from the input text verbatim. Therefore, it is important to know what terms should be included in the summary.

In this paper we present the approach of using social network analysis technique to find terms, specifically noun phrases (NPs) in our experiments, which occur in the human-written abstracts. We show that centrality measures increase the prediction accuracy. Two ways of constructing noun

Proceedings of the ACL Student Research Workshop, pages 103–108,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

phrase network are compared. Conclusions and future work are discussed at the end.

2 Centrality Measures

Social network analysis studies linkages among social entities and the implications of these linkages. The social entities are called actors. A social network is composed of a set of actors and the relation or relations defined on them (Wasserman and Faust, 1994). Graph theory has been used in social network analysis to identify those actors who impose more influence upon a social network. A social network can be represented by a graph with the actors denoted by the nodes and the relations by the edges or links. To determine which actors are prominent, a measure called centrality is introduced. In practice, four types of centrality are often used.

Degree centrality measures how many direct connections a node has to other nodes in a network. Since this measure depends on the size of the network, a standardized version is used when it is necessary to compare the centrality across networks of different sizes.

$$\textit{DegreeCentrality}(n_i) = d(n_i)/(u-1),$$

where $d(n_i)$ is the degree of node i in a network and u is the number of nodes in that network.

Closeness centrality focuses on the distances an actor is from all other nodes in the network.

$$\textit{ClosenessCentrality}(n_i) = (u-1) \Big/ \sum_{j=1}^{u} d(n_i, n_j),$$

where $d(n_i, n_j)$ is the shortest distance between node i and j.

Betweenness centrality emphasizes that for an actor to be central, it must reside on many geodesics of other nodes so that it can control the interactions between them.

$$\textit{BetweennessCentrality}(n_i) = \frac{\sum_{j<k} g_{jk}(n_i)/g_{jk}}{(u-1)(u-2)/2},$$

where g_{jk} is the number of geodesics linking node j and k, $g_{jk}(n_i)$ is the number of geodesics linking the two nodes that contain node i.

Betweenness centrality is widely used because of its generality. This measure assumes that information flow between two nodes will be on the geodesics between them. Nevertheless, “It is quite possible that information will take a more circuitous route either by random communication or [by being] channeled through many intermediaries in order to 'hide' or 'shield' information”. (Stephenson and Zelen, 1989).

Stephenson and Zelen (1989) developed *information centrality* which generalizes betweenness centrality. It focuses on the information contained in all paths originating with a specific actor. The calculation for information centrality of a node is in the Appendix.

Recently centrality measures have started to gain attention from researchers in text processing. Corman et al. (2002) use vectors, which consist of NPs, to represent texts and hence analyze mutual relevance of two texts. The values of the elements in a vector are determined by the betweenness centrality of the NPs in a text being analyzed. Erkan and Radev (2004) use the PageRank method, which is the application of centrality concept to the Web, to determine central sentences in a cluster for summarization. Vanderwende et al. (2004) also use the PageRank method to pick prominent triples, i.e. (node i, relation, node j), and then use the triples to generate event-centric summaries.

3 NP Networks

To construct a network for NPs in a text, we try two ways of modeling the relation between them. One is at the sentence level: if two noun phrases can be sequentially parsed out from a sentence, a link is added between them. The other way is at the document level: we simply add a link to every pair of noun phrases which are parsed out in succession. The difference between the two ways is that the network constructed at the sentence level ignores the existence of certain connections between sentences.

We process a text document in four steps.

First, the text is tokenized and stored into an internal representation with structural information.

Second, the tokenized text is tagged by the Brill tagging algorithm POS tagger.[1]

Third, the NPs in a text document are parsed according to 35 parsing rules as shown in Figure 1. If a new noun phrase is found, a new node is formed and added to the network. If the noun phrase already exists in the network, the node containing it will be identified. A link will be added between two nodes if they are parsed out sequentially for

[1] The POS tagger we used can be obtained from http://web.media.mit.edu/~hugo/montytagger/

the network formed at the document level, or sequentially in the same sentence for the network formed at the sentence level.

Finally, after the text document has been processed, the centrality of each node in the network is updated.

4 Predicting NPs Occurring in Abstracts

In this paper, we refer the NPs occur both in a text document and its corresponding abstract as Co-occurring NPs (CNPs).

4.1 CMP-LG Corpus

In our experiment, a corpus of 183 documents was used. The documents are from the Computation and Language collection and have been marked in XML with tags providing basic information about the document such as title, author, abstract, body, sections, etc. This corpus is a part of the TIPSTER Text Summarization Evaluation Conference (SUMMAC) effort acting as a general resource to the information retrieval, extraction and summarization communities. We excluded five documents from this corpus which do not have abstracts.

4.2 Using Noun Phrase Centrality Heuristics

We assume that a noun phrase with high centrality is more likely to be a central topic being addressed in a document than one with low centrality. Given this assumption, we performed an experiment, in which the NPs with highest centralities are retrieved and compared with the actual NPs in the abstracts. To evaluate this method, we use Precision, which measures the fraction of true CNPs in all predicted CNPs, and Recall, which measures the fraction of correctly predicted CNPs in all CNPs.

NX --> CD
NX --> CD NNS
NX --> NN
NX --> NN NN
NX --> NN NNS
NX --> NN NNS NN
NX --> NNP
NX --> NNP CD
NX --> NNP NNP
NX --> NNP NNPS
NX --> NNP NN
NX --> NNP NNP NNP
NX --> JJ NN
NX --> JJ NNS
NX --> JJ NN NNS
NX --> PRP$ NNS
NX --> PRP$ NN
NX --> PRP$ NN NN
NX --> NNS
NX --> PRP
NX --> WP$ NNS
NX --> WDT
NX --> EX
NX --> WP
NX --> DT JJ NN
NX --> DT CD NNS
NX --> DT VBG NN
NX --> DT NNS
NX --> DT NN
NX --> DT NN NN
NX --> DT NNP
NX --> DT NNP NN
NX --> DT NNP NNP
NX --> DT NNP NNP NNP
NX -->DT NNP NNP NN NN

Figure 1. NP Parsing Rules

After establishing the NP network for a document and ranking the nodes according to their centralities, we must decide how many NPs should be retrieved. This number should not be too big; otherwise the Precision value will be very low, although the Recall will be higher. If this number is very small, the Recall will decrease correspondingly. We adopted a compound metric — F-measure, to balance the selection:

F-measure=2*Precision*Recall/(Precision+Recall)

Based on our study of 178 documents in the CMP-LG corpus, we find that the number of CNPs is roughly proportional to the number of NPs in the abstract. We obtain a linear regression model for the data shown in Figure 2 and use this model to calculate the number of nodes we should retrieve from the NP network, given the number of NPs in the abstract known a priori:

Number of Common NPs =
0.555 * Number of NPs in Abstract + 2.435

One could argue that the number of abstract NPs is unknown a priori and thus the proposed method is of limited use. However, the user can provide an estimate based on the desired number of words in the summary. Here we can adopt the same way of asking the user to provide a limit for the NPs in the summary. We used the actual number of NPs the author used in his/her abstract in our experiment.

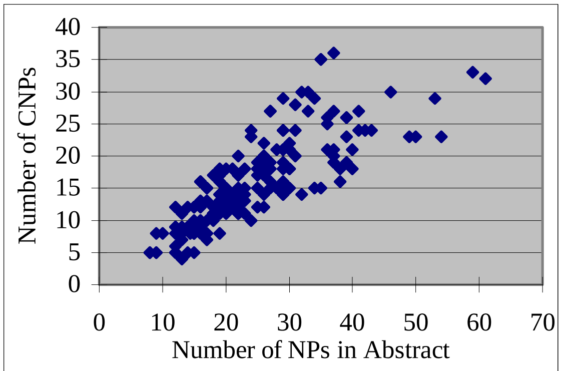


Figure 2. Scatter Plot of CNPs

Our experiment results are shown in Figure 3(a) and 3(b). In 3(a) the NP network is formed at sentence level. In this way, it is possible the graph will be composed of disconnected subgraphs. In such case, we calculate the closeness centrality (*cc*), betweenness centrality (*bc*), and the information centrality (*ic*) within the subgraphs while the degree centrality (*dc*) is still computed for the overall graph. In 3(b), the network is constructed at the document level. Therefore, it is guaranteed that every node is reachable from all other node.

Figure 3(a) shows the simplest centrality measure *dc* performs best, with Precision, Recall, and F-measure all greater than 0.2, which are twice of *bc* and almost ten times of *cc* and *ic*.

In Figure 3(b), however, all four measures are around 0.25 in all three evaluation metrics. This result suggests to us that when we choose a centrality to represent the prominence of a NP in the text, not only does the kind of the centrality matter, but also the way of forming the NP network.

Overall, the heuristic of using centrality itself does not achieve impressive scores. We will see in the next section that using decision trees is a much better way to perform the predictions, when using centrality together with other text features.

4.3 Using Decision Trees

We obtain the following features for all NPs in a document from the CMP-LG corpus:

Position: the order of a NP appearing in the text, normalized by the total number of NPs.
Article: three classes are defined for this attribute: INDEfinite (contains a or an), DEFInite (contains the), and NONE (all others).
Degree centrality: obtained from the NP network
Closeness centrality: obtained from the NP network
Betweenness centrality: obtained from the NP network
Information centrality: obtained from the NP network
Head noun POS tag: a head noun is the last word in the NP. Its POS tag is used here.
Proper name: whether the NP is a proper name, by looking at the POS tags of all words in the NP.
Number: whether the NP is just one number.
Frequency: how many times a NP occurs in a text, normalized by its maximum.
In abstract: whether the NP appears in the author-provided abstract. This attribute is the target for the decision trees to classify.

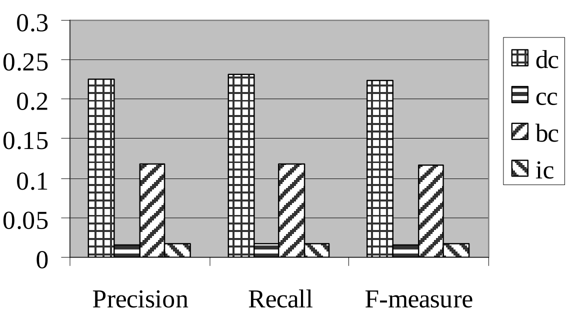


Figure 3(a). Centrality Heuristics (Network at Sentence Level)

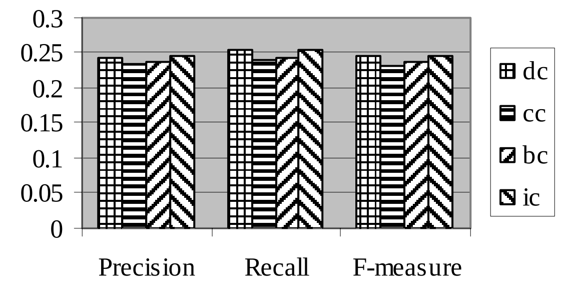


Figure 3(b). Centrality Heuristics (Network at Document Level)

In order to learn which type of centrality measures helps to improve the accuracy of the predictions, and to see whether centrality measures are better than term frequency, we experiment with six groups of feature sets and compare their performances. The six groups are:
All: including all features above.
DC: including only the degree centrality measure, and other non-centrality measures except for Frequency.
CC: same as DC except for using closeness centrality instead of degree centrality.
BC: same as DC except for using betweenness centrality instead of degree centrality.
IC: same as DC except for using information centrality instead of degree centrality.
FQ: including Frequency and all other non-centrality features.

The 178 documents have generated more than 100,000 training records. Among them only a very small portion (2.6%) belongs to the positive class. When using decision tree algorithm on such imbalanced attribute, it is very common that the class with absolute advantages will be favored (Japkowicz, 2000; Kubat and Matwin, 1997). To reduce

		All				DC				CC			
Sentence Level	Precision	0.817	0.816	0.795	**0.809**	0.767	0.787	0.732	**0.762**	0.774	0.795	0.769	**0.779**
	Recall	0.971	0.984	0.96	**0.972**	0.791	0.866	0.8	**0.819**	0.651	0.696	0.639	**0.662**
	F-measure	0.887	0.892	0.869	**0.883**	0.779	0.825	0.764	**0.789**	0.706	0.742	0.696	**0.715**
Document Level	Precision	0.795	0.82	0.795	**0.803**	0.772	0.806	0.768	**0.782**	0.767	0.806	0.766	**0.78**
	Recall	0.944	0.976	0.946	**0.955**	0.79	0.892	0.755	**0.812**	0.72	0.892	0.644	**0.752**
	F-measure	0.863	0.891	0.864	**0.873**	0.781	0.846	0.761	**0.796**	0.743	0.846	0.698	**0.763**
		Set 1	Set 2	Set 3	**Mean**	Set 1	Set 2	Set 3	**Mean**	Set 1	Set 2	Set 3	**Mean**
		BC				IC				FQ			
Sentence Level	Precision	0.738	0.799	0.745	**0.761**	0.722	0.759	0.743	**0.742**	0.774	0.79	0.712	**0.759**
	Recall	0.698	0.874	0.733	**0.768**	0.666	0.799	0.667	**0.711**	0.763	0.878	0.78	**0.807**
	F-measure	0.716	0.835	0.737	**0.763**	0.693	0.779	0.702	**0.724**	0.768	0.831	0.744	**0.781**
Document Level	Precision	0.767	0.799	0.75	**0.772**	0.756	0.798	0.759	**0.771**	0.734	0.794	0.74	**0.756**
	Recall	0.672	0.814	0.666	**0.717**	0.769	0.916	0.72	**0.802**	0.728	0.886	0.707	**0.774**
	F-measure	0.716	0.806	0.705	**0.742**	0.762	0.853	0.738	**0.784**	0.73	0.837	0.722	**0.763**
		Set 1	Set 2	Set 3	**Mean**	Set 1	Set 2	Set 3	**Mean**	Set 1	Set 2	Set 3	**Mean**

Table 1. Results for Using 6 Feature Sets with YaDT

the unfair preference, one way is to boost the weak class, e.g., by replicating instances in the minority class (Kubat and Matwin, 1997; Chawla et al., 2000). In our experiments, the 178 documents were arbitrarily divided into three roughly equal groups, generating 36,157, 37,600, and 34,691 records, respectively. After class balancing, the records are increased to 40,109, 42,210, and 38,499. The three data sets were then run through the decision tree algorithm YaDT (Yet another Decision Tree builder), which is much more efficient than C4.5 (Ruggieri, 2004),[2] with 10-fold cross validation.

The experiment results of using YaDT with three data sets and six feature groups to predict the CNPs are shown in Table 1. The mean values of three metrics are also shown in Figure 4(a) and 4(b). Decision trees achieve much higher scores compared with the scores obtained by using centrality heuristics. Together with other text features, DC, CC, BC, and IC obtain scores over 0.7 in all three metric, which are comparable to the scores obtained by using FQ. Moreover, when using all the features, decision trees achieve over 0.8 in precision and over 0.95 in recall. F-measure is as high as 0.88. To see whether F-measure of All is statistically better than that of other settings, we run *t*-tests to compare them using values of F-measure obtained in the 10-fold cross-validation from the three data sets. The results show the mean value of F-measure of All is significantly higher (*p-value =0.000*) than that of other settings.

Differently from the experiments that use centrality heuristics by itself, almost no obvious distinctions can be observed when comparing the performances of YaDT with NP network formed in two ways.

5 Conclusions and Future work

We have studied four kinds of centrality measures in order to identify prominent noun phrases in text documents. Overall, the centrality heuristic itself does not demonstrate its superiority. Among four centrality measures, degree centrality performs the best in the heuristic when the NP network is constructed at the sentence level, which indicates other centrality measures obtained from the subgraphs can not represent very well the prominence of the NPs in the global NP network. When the NP network is constructed at the document level, the differences between the centrality measures become negligible. However, networks formed at the document level overlook the connections between sentences as there is only one kind of link; on the other hand, NP networks formed at the sentence level ignore connections between sentences. We plan to extend our study to construct NP networks with weighted links. The key problem will be how to determine the weights for links between two NPs in the same sentence, in the same paragraph but different sentences, and in different paragraphs. We consider introducing the concept of entropy from Information Theory to solve this problem.

In our experiments with YaDT, it seems the ways of forming NP network are not critical. We learn that, at least in this circumstance, the decision trees algorithm is more robust than the centrality heuristic. When using all features in YaDT, recall reaches 0.95, which means the decision trees find out 95% of CNPs in the abstracts from the text documents, without increasing mistakes as the

[2] The YaDT software can be obtained from http://www.di.unipi.it/~ruggieri/software.html

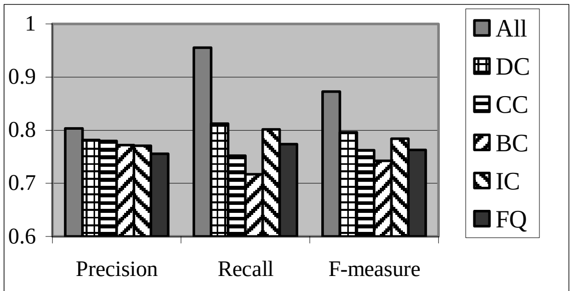


Figure 4(a). Results with NP Network Formed in Sentence Level

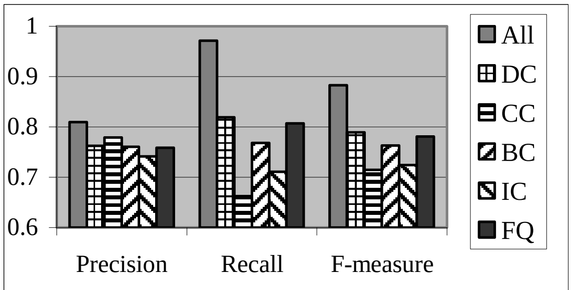


Figure 4(b). Results with NP Network Formed in Document Level

precision is improved at the same time. Using all features in YaDT achieves better results than using centrality feature or frequency individually with other features implies centrality features may capture somewhat different information from the text.

To make this research more robust, we will include reference resolution into our study. We will also include centrality measures as sentence features in producing extractive summaries.

References

N. Chawla, K. Bowyer, L. Hall, and W. P. Kegelmeyer. 2000. SMOTE: synthetic minority over-sampling technique. In *Proc. of the International Conference on Knowledge Based Computer Systems*, India.

S. Corman, T. Kuhn, R. McPhee, and K. Dooley. 2002. Studying complex discursive systems: Centering resonance analysis of organizational communication. *Human Communication Research*, *28*(2):157-206.

G. Erkan and D. R. Radev. 2004. The University of Michigan at DUC 2004. In *Document Understanding Conference 2004*, Boston, MA.

N. Japkowicz. 2000. The class imbalance problem: significance and strategies. In *Proc. of the 2000 International Conference on Artificial Intelligence*.

D. Jurafsky and J. H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ.

M. Kubat and S. Matwin. 1997. Addressing the curse of imbalanced data sets: one-sided sampling. In *Proc. of the Fourteenth International Conference on Machine Learning*, Morgan Kauffman, 179–186.

S. Ruggieri. 2004. YaDT: Yet another Decision Tree builder. In *Proc. of the 16th International Conference on Tools with Artificial Intelligence (ICTAI 2004),* 260-265. Boca Raton, FL

K. Stephenson and M. Zelen. 1989. Rethinking centrality: Methods and applications. *Social Networks.* 11:1-37.

L. Vanderwende, M. Banko and A. Menezes. 2004. Event-Centric Summary Generation. In *Document Understanding Conference 2004.* Boston, MA.

S. Wasserman and K. Faust. 1994. *Social Network Analysis: Methods and applications.* Cambridge University Press.

C. T. Yu and W. Meng. 1998. *Principles of Database Query Processing for Advanced Applications.* Morgan Kaufmann Publishers, San Francisco, CA.

Appendix: Calculation of Information Centrality

Consider a network with n points where every pair of points is reachable. Define the $n \times n$ matrix $B = (b_{ij})$ by:

$$b_{ij} = \begin{cases} 0 & \text{if points } i \text{ and } j \text{ are incident} \\ 1 & \text{otherwise;} \end{cases}$$

$$b_{ii} = 1 + \text{degree of point } i$$

Define the matrix $C = (c_{ij}) = B^{-1}$. The value of $\boldsymbol{I}_{ij}$ (the information in the combined path P_{ij}) is given explicitly by

$$I_{ij} = (c_{ii} + c_{jj} - 2c_{ij})^{-1}.$$

We can write

$$\sum_{j=1}^{n} 1/I_{ij} = \sum_{j=1}^{n} (c_{ii} + c_{jj} - 2c_{ij}) = nc_{ii} + T - 2R,$$

where

$$T = \sum_{j=1}^{n} c_{jj} \quad \text{and} \quad R = \sum_{j=1}^{n} c_{ij}.$$

Therefore the centrality for point i can be explicitly written as

$$I_i = \frac{n}{nc_{ii} + T - 2R} = \frac{1}{c_{ii} + (T - 2R)/n}.$$

(Stephenson and Zelen 1989).

A corpus-based approach to topic in Danish dialog[*]

Philip Diderichsen
Lund University Cognitive Science
Lund University
Sweden
philip.diderichsen@lucs.lu.se

Jakob Elming
CMOL / Dept. of Computational Linguistics
Copenhagen Business School
Denmark
je.id@cbs.dk

Abstract

We report on an investigation of the pragmatic category of topic in Danish dialog and its correlation to surface features of NPs. Using a corpus of 444 utterances, we trained a decision tree system on 16 features. The system achieved near-human performance with success rates of 84–89% and F_1-scores of 0.63–0.72 in 10-fold cross validation tests (human performance: 89% and 0.78). The most important features turned out to be preverbal position, definiteness, pronominalisation, and non-subordination. We discovered that NPs in epistemic matrix clauses (e.g. "I think . . . ") were seldom topics and we suspect that this holds for other interpersonal matrix clauses as well.

1 Introduction

The pragmatic category of topic is notoriously difficult to pin down, and it has been defined in many ways (Büring, 1999; Davison, 1984; Engdahl and Vallduví, 1996; Gundel, 1988; Lambrecht, 1994; Reinhart, 1982; Vallduví, 1992). The common denominator is the notion of topic as what an utterance is about. We take this as our point of departure in this corpus-based investigation of the correlations between linguistic surface features and pragmatic topicality in Danish dialog.

Danish is a verb-second language. Its word order is fixed, but only to a certain degree, in that it allows any main clause constituent to occur in the preverbal position. The first position thus has a privileged status in Danish, often associated with topicality (Harder and Poulsen, 2000; Togeby, 2003). We were thus interested in investigating how well the topic correlates with the preverbal position, along with other features, if any.

Our findings could prove useful for the further investigation of local dialog coherence in Danish. In particular, it may be worthwile in future work to study the relation of our notion of topic to the C_b of Grosz et al.s (1995) Centering Theory.

2 The corpus

The basis of our investigation was two dialogs from a corpus of doctor-patient conversations (Hermann, 1997). Each of the selected dialogs was between a woman in her thirties and her doctor. The doctor was the same in the two conversations, and the overall topic of both was the weight problems of the patient. One of the dialogs consisted of 125 utterances (165 NPs), the other 319 (449 NPs).

3 Method

The investigation proceeded in three stages: first, the topic expressions (see below) of all utterances were identified[1]; second, all NPs were annotated for linguistic surface features; and third, decision trees

[*] We thank Daniel Hardt and two anonymous reviewers for many helpful comments on drafts of this paper.

[1] Utterances with dicourse regulating purpose (e.g. yes/no-answers), incomplete utterances, and utterances without an NP were excluded.

Proceedings of the ACL Student Research Workshop, pages 109–114,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

were generated in order to reveal correlations between the topic expressions and the surface features.

3.1 Identification of topic expressions

Topics are distinguished from *topic expressions* following Lambrecht (1994). Topics are entities pragmatically construed as being what an utterance is about. A topic expression, on the other hand, is an NP that formally expresses the topic in the utterance. Topic expressions were identified through a two-step procedure; 1) identifying topics and 2) determining the topic expressions on the basis of the topics.

First, the topic was identified strictly based on pragmatic aboutness using a modified version of the '*about* test' (Lambrecht, 1994; Reinhart, 1982).

The *about* test consists of embedding the utterance in question in an 'about-sentence' as in Lambrecht's example shown below as (1):

(1) He said about the children that they went to school.

This is a paraphrase of the sentence *the children went to school* which indicates that the referent of *the children* is the topic because it is appropriate (in the imagined discourse context) to embed this referent as an NP in the *about* matrix clause. (Again, the referent of *the children* is the topic, while the NP *the children* is the topic expression.)

We adapted the *about* test for dialog by adding a request to 'say something about ...' or 'ask about ...' before the utterance in question. Each utterance was judged in context, and the best topic was identified as illustrated below. In example (2), the last utterance, (2-D3), was assigned the topic TIME OF LAST WEIGHING. This happened after considering which *about* construction gave the most coherent and natural sounding result combined with the utterance. Example (3) shows a few *about* constructions that the coder might come up with, and in this context (3-iv) was chosen as the best alternative.

(2) D_1 sid ned og lad mig høre, Annette (made-up name)
sit down and let me hear, Annette
P_1 jeg skal bare vejes
I shall just be.weighed
P_2 og så skal jeg have svar fra sidste gang
and then shall I have answer from last time
D_2 så skal vi se en gang
then let us see one time
D_3 **det**... er... fjorten dage siden du blev vejet...
it... is... fourteen days since you were weighed...

(3) i. Say something about THE PATIENT (=you).
ii. Say something about THE WEIGHING OF THE PATIENT.
iii. Say something about THE LAST WEIGHING OF THE PATIENT.
iv. Say something about THE TIME OF LAST WEIGHING OF THE PATIENT.

Creating the *about* constructions involved a great deal of creativity and made them difficult to compare. Sometimes the coders chose the exact same topic, at other times they were obviously different, but frequently it was difficult to decide. For instance, for one utterance Coder 1 chose OTHER CAUSES OF EDEMA SYMPTOM, while Coder 2 chose THE EDEMA'S CONNECTION TO OTHER THINGS. Slightly different wordings like these made it impossible to test the intersubjectivity of the topic coding.

The second step consisted in actually identifying the topic expression. This was done by selecting the NP in the utterance that was the best formal representation of the topic, using 3 criteria:

1. The topic expression is the NP in the utterance that refers to the topic.
2. If no such NP exists, then the topic expression is the NP whose referent the topic is a property or aspect of.
3. If no NP fulfills one of these criteria, then the utterance has no topic expression.

In the example from before, (2-D3), it was judged that *det* 'it' (emphasized) was the topic expression of the utterance, because it shared reference with the chosen topic from (3-iv).

If two NPs in an utterance had the same reference, the best topic representative was chosen. In reflexive constructions like (4), the non-reflexive NP, in this case *jeg* 'I', is considered the best representative.

(4) men jeg har ikke tabt mig
but I have not lost me (i.e. lost weight)

In syntactially complex utterances, the best representative of the topic was considered the one occurring in the clause most closely related to the topic. In the following example, since the topic was THE PATIENT'S HANDLING OF EATING, the topic expression had to be one of the two instances of *jeg* 'I'. Since the topic arguably concerns 'handling' more than 'eating', the NP in the matrix clause (emphasized) is the topic expression.

(5) **jeg** har slet ikke tænkt på hvad jeg har spist
I have really not thought about what I have eaten

A final example of several NPs referring to the same topic has to do with left-dislocation. In example (6), the preverbal object *ham* 'him' is immediately preceded by its antecedent *min far* 'my father'. Both NPs express the topic of the utterance. In Danish, resumptive pronouns in left-dislocation constructions always occur in preverbal position, and in cases where they express the topic there will thus always be two NPs directly adjacent to each other which both refer to the topic. In such cases, we consider the resumptive pronoun the topic expression, partly because it may be considered a more integrated part of the sentence (cf. Lambrecht (1994)).

(6) min far **ham** så jeg sjældent
my father him saw I seldom

The intersubjectivity of the topic expression annotation was tested in two ways. First, all the topic expression annotations of the two coders were compared. This showed that topic expressions can be annotated reasonably reliably (κ = 0.70 (see table 1)). Second, to make sure that this intersubjectivity was not just a product of mutual influence between the two authors, a third, independent coder annotated a small, random sample of the data for topic expressions (50 NPs). Comparing this to the annotation of the two main coders confirmed reasonable reliability (κ = 0.70).

3.2 Surface features

After annotating the topics and topic expressions, 16 grammatical, morphological, and prosodic features were annotated. First the smaller corpus was annotated by the two main coders in collaboration in order to establish annotating policies in unclear cases. Then the features were annotated individually by the two coders in the larger corpus.

Grammatical roles. Each NP was categorized as grammatical subject (sbj), object (obj), or oblique (obl).These features can be annotated reliably (sbj: C1 (number of sbj's identified by Coder 1) = 208, C2 (sbj's identified by Coder 2) = 207, C1+2 (Coder 1 and 2 overlap) = 207, κ_{sbj} = **1.00**; obj: C1 = 110, C2 = 109, C1+2 = 106, κ_{obj} = **0.97**; obl: C1 = 30, C2 = 50, C1+2 = 29, κ_{obl} = **0.83**).

Morphological and phonological features. NPs were annotated for pronominalisation (pro), definiteness (def), and main stress (str). (Note that the main stress distinction only applies to pronouns in Danish.) These can also be annotated reliably (pro: C1 = 289, C2 = 289, C1+2 = 289, κ_{pro} = **1.00**; def: C1 = 319, C2 = 318, C1+2 = 318, κ_{def} = **0.99**; str: C1 = 226, C2 = 226, C1+2 = 203, κ_{str} = **0.80**).

Unmarked surface position. NPs were annotated for occurrence in pre-verbal (pre) or post-verbal (post) position relative to their subcategorizing verb. Thus, in the following example, *det* 'it' is +pre, but –post, because *det* is not subcategorized by *tror* 'think'.

(7) Ø tror [$_{\text{+pre,–post}}$ **det**] hjælper lidt
(I) think [$_{\text{+pre,–post}}$ it] helps a little

In addition to this, NPs occurring in pre-verbal position were annotated for whether they were repetitions of a left-dislocated element (ldis). Example (8) further exemplifies the three position-related features.

(8) min far [$_{\text{+ldis,+pre}}$ ham] så [$_{\text{+post}}$ jeg] sjældent
my father [$_{\text{+ldis,+pre}}$ him] saw [$_{\text{+post}}$ I] seldom

All three features can be annotated highly reliably (pre: C1 = 142, C2 = 142, C1+2 = 142, κ_{pre} = **1.00**; post: C1 = 88, C2 = 88, C1+2 = 88, κ_{post} = **1.00**; ldis: C1 = 2, C2 = 2, C1+2 = 2, κ_{ldis} = **1.00**).

Marked NP-fronting. This group contains NPs fronted in marked constructions such as the passive (pas), clefts (cle), Danish 'sentence intertwining' (dsi), and XVS-constructions (xvs).

NPs fronted as subjects of passive utterances were annotated as +pas.

(9) [$_{\text{+pas}}$ **jeg**] skal bare vejes
[$_{\text{+pas}}$ I] shall just be.weighed

A cleft construction is defined as a complex construction consisting of a copula matrix clause with a relative clause headed by the object of the matrix clause. The object of the matrix clause is also an argument or adjunct of the relative clause predicate. The clefted element *det* 'that', which we annotate as +cle, leaves an 'empty slot', *e*, in the relative clause, as shown in example (10):

(10) det er jo ikke [$_{\text{+cle}}$ **det**$_i$] du skal tabe dig
it is after all not [$_{\text{+cle}}$ that$_i$] you shall lose weight
af $\boldsymbol{e_i}$ som sådan
from e_i as such

Danish sentence intertwining can be defined as a special case of extraction where a non-WH constituent of a subordinate clause occurs in the first position of the matrix clause. As in cleft constructions, an ‘empty slot’ is left behind in the subordinate clause. NPs in the fronted position were annotated as +dsi:

(11) [$_{\text{+dsi}}$ **det**$_i$] tror jeg ikke det gør $\boldsymbol{e}_i$
[$_{\text{+dsi}}$ that$_i$] think I not it does e_i

The XVS construction is defined as a simple declarative sentence with anything but the subject in the preverbal position. Since only one constituent is allowed preverbally[2], the subject occurs after the finite verb. In example (12), the finite verb is an auxiliary, and the canonical position of the object after the main verb is indicated with the ‘empty slot’ marker *e*. The preverbal element in XVS-constructions is annotated as +xvs.

(12) [$_{\text{+xvs}}$ **det**$_i$] har jeg altså haft $\boldsymbol{e}_i$ før
[$_{\text{+xvs}}$ that$_i$] have I truly had e_i before

All four features can be annotated highly reliably (pas: C1 = 1, C2 = 1, C1+2 = 1, κ_{pas} = **1.00**; cle: C1 = 4, C2 = 4, C1+2 = 4, κ_{cle} = **1.00**; dsi C1 = 3, C2 = 3, C1+2 = 3, κ_{dsi} = **1.00**; xvs: C1 = 18, C2 = 18, C1+2 = 18, κ_{xvs} = **1.00**).

Sentence type and subordination. Each NP was annotated with respect to whether or not it appeared in an interrogative sentence (int) or a subordinate clause (sub), and finally, all NPs were coded as to whether they occurred in an epistemic matrix clause or in a clause subordinated to an epistemic matrix clause (epi). An epistemic matrix clause is defined as a matrix clause whose function it is to evaluate the truth of its subordinate clause (such as “*I think* . . . ”). The following example illustrates how we annotated both NPs in the epistemic matrix clause and NPs in its immediate subordinate clause as +epi, but not NPs in further subordinated clauses. The +epi feature requires a +/–sub feature in order to determine whether the NP in question is in the epistemic matrix clause or subordinated under it. Subordination is shown here using parentheses.

(13) [$_{\text{+epi}}$ jeg] tror mere ([$_{\text{+epi,+sub}}$ det] er fordi (at
[$_{\text{+epi}}$ I] think rather ([$_{\text{+epi,+sub}}$ it] is because (that
[$_{\text{+sub}}$ man] spiser på [$_{\text{+sub}}$ dumme tidspunkter] ik’))
[$_{\text{+sub}}$ you] eat at [$_{\text{+sub}}$ stupid times] right))

All features in this group can be annotated reliably (int: C1 = 55, C2 = 55, C1+2 = 55, κ_{int} = **1.00**; sub: C1 = 117, C2 = 111, C1+2 = 107, κ_{sub} = **0.93**; epi: C1 = 38, C2 = 45, C1+2 = 37, κ_{epi} = **0.92**).

3.3 Decision trees

In the third stage of our investigation, a decision tree (DT) generator was used to extract correlations between topic expressions and surface features. Three different data sets were used to train and test the DTs, all based on the larger dialog.

Two of these data sets were derived from the complete set of NPs annotated by each main coder individually. These two data sets will be referred to below as the ‘Coder 1’ and ‘Coder 2’ data sets.

The third data set was obtained by including only NPs annotated identically by both main coders in relevant features[3]. This data set represents a higher degree of intersubjectivity, especially in the topic expression category, but at the cost of a smaller number of NPs. 63 out of a total of 449 NPs had to be excluded because of inter-coder disagreement, 50 due to disagreement on the topic expression category. This data set will be referred to below as the ‘Intersection’ data set.

A DT was generated for each of these three data sets, and each DT was tested using 10-fold cross validation, yielding the success rates reported below.

4 Results

Our results were on the one hand a subset of the features examined that correlated with topic expressions, and on the other the discovery of the importance of different types of subordination. These results are presented in turn.

4.1 Topic-indicating features

The optimal classification of topic expressions included a subset of important features which appeared in every DT, i.e. +pro, +def, +pre, and –sub. Several other features occurred in some of the DTs, i.e. dsi, int, and epi. The performance of all the DTs is summarized in table 2 below.

[2] Only one constituent is allowed in the *intrasentential* preverbal position. Left-dislocated elements are not considered part of the sentence proper, and thus do not count as preverbal elements, cf. Lambrecht (1994).

[3] “Relevant features” were determined in the following way: A DT was generated using a data set consisting only of NPs annotated identically by the two coders in all the features, i.e. the 16 surface features as well as the topic expression feature. The features constituting this DT, i.e. pro, def, sub, and pre, as well as the topic expression category, were relevant features for the third data set, which consisted only of NPs coded identically by the two coders in these 5 features.

The DT for the Coder 1 data set contains the features def, pro, dsi, sub, and pre. According to this classification, a definite pronoun in the fronted position of a Danish sentence intertwining construction is a topic expression, and other than that, definite pronouns in the preverbal position of non-subordinate clauses are topic expressions. The 10-fold cross validation test yields an 84% success rate. F_1-score: 0.63.

The Coder 2 DT contains the features pro, def, sub, pre, int, and epi. Here, if a definite pronoun occurs in a subordinate clause it is not a topic expression, and otherwise it is a topic expression if it occurs in the preverbal position. If it does not occur in preverbal position, but in a question, it is also a topic expression unless it occurs in an epistemic matrix clause. Success rate: 85%. F_1-score: 0.67.

Finally, the Intersection DT contains the features pro, def, sub, and pre. According to this DT, only definite pronouns in preverbal position in non-subordinate clauses are topic expressions. The DT has a high success rate of 89% in the cross validation test — which is not surprising, given that a large number of possibly difficult cases have been removed (mainly the 50 NPs where the two coders disagreed on the annotation of topic expressions). F_1-score: 0.72.

Since there is no gold standard for annotating topic expressions, the best evaluation of the human performance is in terms of the amount of agreement between the two coders. Success rate and F_1 analogs for human performance were therefore computed as follows, using the figures displayed in table 1.

		Coder 2		Total
		Topic	Non-topic	
Coder 1	Topic	88	27	115
	Non-topic	23	311	334
Total		111	338	449

Table 1: The topic annotation of Coder 1 and Coder 2.

Success rate analog: The agreement percentage between the human coders when annotating topic expressions ($\frac{449\text{ NPs}-(23+27)\text{ NPs}}{449\text{ NPs}}\times 100 = 89\%$).

***F_1* analog**: The performance of Coder 1 evaluated against the performance of Coder 2 ("Precision": $\frac{88}{88+27} = 0.77$; "Recall": $\frac{88}{88+23} = 0.79$; "F_1": $2 \times \frac{0.77\times 0.79}{0.77+0.79} = 0.78$).

Data set	Coder 1	Coder 2	Intersect.	Human
Total NPs	449	449	386	449
Success rate	84%	85%	89%	89%
Precision	0.77	0.74	0.79	0.79
Recall	0.53	0.61	0.67	0.77
F_1-score	0.63	0.67	0.72	0.78

Table 2: Success rates, Precision, Recall, and F_1-scores for the three different data sets. For comparison, we added success rate and F_1 analogs for human performance.

4.2 Interpersonal subordination

We found that syntactic subordination does not have an invariant function as far as information structure is concerned. The emphasized NPs in the following examples are definite pronouns in preverbal position in syntactically non-subordinate clauses. But none of them are perceived as topic expressions.

(14) så **det** kan godt være at hvis man har... tabt noget
so it may well be that if you have... lost some
mere i løbet af ugen ik'
more during the.week right

(15) **jeg** tror mere det er fordi at man spiser på
I think rather it is because that you eat at
dumme tidspunkter ik'
stupid times right

The reason seems to be that these NPs occur in epistemic matrix clauses (+epi).

The following utterances have not been annotated for the +epi feature, since the matrix clauses do not seem to state the speaker's attitude towards the truth of the subordinate clause. However, the emphasized NPs seem to stand in a very similar relation to the message being conveyed, and none of them were perceived as topic expressions.

(16) men altså **jeg** har bare bemærket at at det
but you know I have just noticed that that it
er blevet værre ik'
has become worse right

(17) og **det** kan man da sige på tre uger det er
and that can you though say in three weeks that is
da ikke vildt meget
surely not wildly much

This suggests that a more general type of matrix clause than the epistemic matrix clause, namely the *interpersonal matrix clause* (Jensen, 2003) would be relevant in this context. This category would cover all of the above cases. It is defined as a matrix clause that expresses some attitude towards the message conveyed in its subordinate clause. This more general category presumably signals non-topicality rather than topicality just like the special case of epistemic subordination.

5 Summary and future work

We have shown that it is possible to generate algorithms for Danish dialog that are able to predict the topic expressions of utterances with near-human performance (success rates of 84–89%, F_1 scores of 0.63–0.72).

Furthermore, our investigation has shown that the most characteristic features of topic expressions are preverbal position (+pre), definiteness (+def), pronominal realisation (+pro), and non-subordination (–sub). This supports the traditional view of topic as the constituent in preverbal position.

Most interesting is subordination in connection with certain matrix clauses. We discovered that NPs in epistemic matrix clauses were seldom topics. In complex constructions like these the topic expression occurs in the subordinate clause, not the matrix clause as would be expected. We suspect that this can be extended to the more general category of inter-personal matrix clauses.

Future work on dialog coherence in Danish, particularly pronoun resolution, may benefit from our results. The centering model, originally formulated by Grosz et al. (1995), models discourse coherence in terms of a ‘local center of attention’, viz. the *backward-looking center*, C_b. Insofar as the C_b corresponds to a notion like topic, the corpus-based investigation reported here might serve as the empirical basis for an adaptation for Danish dialog of the centering model. Attempts have already been made to adapt centering to dialog (Byron and Stent, 1998), and, importantly, work has also been done on adapting the centering model to other, freer word order languages such as German (Strube and Hahn, 1999).

References

Daniel Büring. 1999. Topic. In Peter Bosch and Rob van der Sandt, editors, *Focus — Linguistic, Cognitive, and Computational Perspectives*, pages 142–165. Cambridge University Press.

Donna K. Byron and Amanda J. Stent. 1998. A preliminary model of centering in dialog. Technical report, The University of Rochester.

Alice Davison. 1984. Syntactic markedness and the definition of sentence topic. *Language*, 60(4).

Elisabeth Engdahl and Enric Vallduví. 1996. Information packaging in HPSG. *Edinburgh working papers in cognitive science: Studies in HPSG*, 12:1–31.

Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: a framework for modeling the local coherence of discourse. *Computational linguistics*, 21(2):203–225.

Jeanette K. Gundel. 1988. Universals of topic-comment structure. In Michael Hammond, Edith Moravcsik, and Jessica Wirth, editors, *Studies in syntactic typology*, volume 17 of *Studies in syntactic typology*, pages 209–239. John Benjamins Publishing Company, Amsterdam/Philadelphia.

Peter Harder and Signe Poulsen. 2000. Editing for speaking: first position, foregrounding and object fronting in Danish and English. In Elisabeth Engberg-Pedersen and Peter Harder, editors, *Ikonicitet og struktur*, pages 1–22. Netværk for funktionel lingvistik, Copenhagen.

Jesper Hermann. 1997. Dialogiske forståelser og deres grundlag. In Peter Widell and Mette Kunøe, editors, *6. møde om udforskningen af dansk sprog*, pages 117–129. MUDS, Århus.

K. Anne Jensen. 2003. *Clause Linkage in Spoken Danish*. Ph.D. thesis from the University of Copenhagen, Copenhagen.

Knud Lambrecht. 1994. *Information structure and sentence form: topic, focus and the mental representations of discourse referents*. Cambridge University Press, Cambridge.

Tanya Reinhart. 1982. Pragmatics and linguistics. an analysis of sentence topics. *Distributed by the Indiana University Linguistics Club.*, pages 1–38.

Michael Strube and Udo Hahn. 1999. Functional centering — grounding referential coherence in information structure. *Computational linguistics*, 25(3):309–344.

Ole Togeby. 2003. *Fungerer denne sætning? – Funktionel dansk sproglære*. Gads forlag, Copenhagen.

Enric Vallduví. 1992. *The informational component*. Ph.D. thesis from the University of Pennsylvania, Philadelphia.

Learning Information Structure in The Prague Treebank

Oana Postolache
Department of Computational Linguistics
University of Saarland, Saarbrücken, Germany
oana@coli.uni-sb.de

Abstract

This paper investigates the automatic identification of aspects of Information Structure (IS) in texts. The experiments use the Prague Dependency Treebank which is annotated with IS following the Praguian approach of Topic Focus Articulation. We automatically detect t(opic) and f(ocus), using node attributes from the treebank as basic features and derived features inspired by the annotation guidelines. We show the performance of C4.5, Bagging, and Ripper classifiers on several classes of instances such as nouns and pronouns, only nouns, only pronouns. A baseline system assigning always f(ocus) has an F-score of 42.5%. Our best system obtains 82.04%.

1 Introduction

Information Structure (IS) is a partitioning of the content of a sentence according to its relation to the discourse context. There are numerous theoretical approaches describing IS and its semantics (Halliday, 1967; Sgall, 1967; Vallduví, 1990; Steedman, 2000) and the terminology used is diverse (see (Kruijff-Korbayová & Steedman, 2003) for an overview). However, all theories consider at least one of the following two distinctions: (i) a topic/focus[1] distinction that divides the linguistic meaning of the sentence into parts that link the content to the context, and others that advance the discourse, i.e. add or modify information; and (ii) a background/kontrast[2] distinction between parts of the utterance which contribute to distinguishing its actual content from alternatives the context makes available. Existing theories, however, state their principles using carefully selected illustrative examples. Because of this, they fail to adequately explain what possibly different linguistic dimensions cooperate to realize IS and how they do it.

In this paper we report the results of an experiment aimed to automatically identify aspects of IS. This effort is part of a larger investigation aimed to get a more realistic view on the realization of IS in naturally occurring texts.

For such an investigation, the existence of a corpus annotated with some kind of 'informativity status' is of great importance. Fully manual annotation of such a corpus is tedious and time-consuming. Our plan is to initially annotate a small amount of data and then to build models to automatically detect IS in order to apply bootstrapping techniques to create a larger corpus.

This paper describes the results of a pilot study; its aim is to check if the idea of learning IS works by trying it on an already existing corpus. For our experiments, we have used the Prague Dependency Treebank (PDT) (Hajič, 1998), as it is the only corpus annotated with IS (following the theory of Topic-Focus Articulation). We trained three different classifiers, C4.5, Bagging and Ripper, using basic features from the treebank and derived features inspired by the annotation guidelines. We have evaluated the performance of the classifiers against a baseline that simulates the preprocessing procedure that preceded the manual annotation of PDT, and

[1] We use the Praguian terminology for this distinction.

[2] The notion 'kontrast' with a 'k' has been introduced in (Vallduví and Vilkuna, 1998) to replace what Steedman calls 'focus', and to avoid confusion with other definitions of focus.

Proceedings of the ACL Student Research Workshop, pages 115–120,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

against a rule-based system which we implemented following the annotation instructions.

The organization of the paper is as follows. Section 2 describes the Prague Dependency Treebank, Section 3 presents the Praguian approach of Topic-Focus Articulation, from two perspectives: of the theoretical definition and of the annotation guidelines that have been followed to annotate the PDT. Section 4 presents the experimental setting, evaluation metric and results. The paper closes with conclusions and issues for future research (Section 5).

2 Prague Dependency Treebank

The Prague Dependency Treebank (PDT) consists of newspaper articles from the Czech National Corpus (Čermaák, 1997) and includes three layers of annotation. **The morphological layer** gives a full morphemic analysis in which 13 categories are marked for all sentence tokens (including punctuation marks). **The analytical layer**, on which the "surface" syntax (Hajič, 1998) is annotated, contains analytical tree structures, in which every token from the surface shape of the sentence has a corresponding node labeled with main syntactic functions like SUBJ, PRED, OBJ, ADV. **The tectogrammatical layer** renders the deep (underlying) structure of the sentence (Sgall et al., 1986; Hajičová et al., 1998). Tectogrammatical tree structures (TGTSs) contain nodes corresponding only to the autosemantic words of the sentence (e.g., no preposition nodes) and to deletions on the surface level; the condition of projectivity is obeyed, i.e. no crossing edges are allowed; each node of the tree is assigned a functor such as ACTOR, PATIENT, ADDRESSEE, ORIGIN, EFFECT, the list of which is very rich; elementary coreference links are indicated, in the case of pronouns.

3 Topic Focus Articulation (TFA)

The tectogrammatical level of the PDT was motivated by the more and more obvious need of large corpora that treat not only the morphological and syntactic structure of the sentence but also semantic and discourse-related phenomena. Thus, TGTSs have been enriched with features displaying the information structure of the sentence which is a means of showing its contextual potential.

3.1 Theory

In the Praguian approach to IS, the content of the sentence is divided in two parts: the Topic is "what the sentence is about" and the Focus represents the information asserted about the Topic. A prototypical declarative sentence asserts that its Focus holds (or does not hold) about its Topic: Focus(Topic) or not-Focus(Topic).

The TFA definition uses the distinction between Context-Bound (CB) and Non-Bound (NB) parts of the sentence. To distinguish which items are CB and which are NB, the question test is applied, (i.e., the question for which a given sentence is the appropriate answer is considered). In this framework, weak and zero pronouns and those items in the answer which reproduce expressions present (or associated to those present) in the question are CB. Other items are NB.

In example (1), (b) is the sentence under investigation, in which CB and NB items are marked, (a) is the context in which the sentence is uttered, and (c) is the question for which the given sentence is an appropriate answer:

(1) (a) Tom and Mary both came to John's party.
(b) John$_{CB}$ invited$_{CB}$ only$_{NB}$ her$_{NB}$.
(c) Whom did John invite?

The following rules determine which lexical items (CB or NB) belong to the Topic or to the Focus (Hajičová et al., 1998; Hajičová and Sgall, 2001):

1. The main verb and any of its direct dependents belong to the Focus if they are NB;
2. Every item that does not depend directly on the main verb and is subordinated to an element of Focus belongs to Focus (where "subordinated to" is defined as the irreflexive transitive closure of "depend on");
3. If the main verb and all its dependents are CB, then those dependents k_i of the verb which have subordinated items l_m that are NB are called 'proxi foci'; the items l_m together with all items subordinated to them belong to Focus, where $i, m > 1$;
4. Every item not belonging to Focus according to 1 – 3 belongs to Topic.

3.2 Annotation guidelines

Within PDT, the TFA attribute has been annotated for all nodes (including the restored ones) at the tectogrammatical level. Instructions for the assignment of TFA attribute have been specified in (Buráňová et al., 2000) and are summarized in Table 1. These instructions are based on the surface word order, the position of the sentence stress (intonation center – IC[3]) and the canonical order of the dependents.

The TFA attribute has 3 values: t, for non-contrastive CB items; f, for NB items; and c, for contrastive CB items. In this paper, we do not distinguish between contrastive and non-contrastive items, considering both of them as being just t. In the PDT annotation, the values t (from topic) and f (from focus) have been chosen to be used because, in the most cases, in prototypical sentences, t items belong to the Topic and f items to the Focus.

Before the manual annotation, the corpus has been preprocessed to mark all nodes with the TFA attribute of f, as it is the more common value. Then the annotators changed the value according to the guidelines in Table 1.

4 Automatic extraction of TFA

In this section we consider the automatic identification of t and f using machine learning techniques trained on the annotated data.

The data set consists of 1053 files (970,920 words) from the pre-released version of PDT 2.0.[4] We restrict our experiments by considering only noun- and pronoun-nodes. The total number of instances (nouns and pronouns) in the data is 297,220 out of which 254,242 (86.54%) are nouns and 39,978 (13.46%) are pronouns. The t/f distribution of these instances is 172,523 f (58.05%) and 124,697 t (41.95%).

We experimented with three different classifiers, C4.5, Bagging and Ripper, because they are based on different machine learning techniques (decision trees, bagging, rules induction) and we wanted to see which of them performs better on this task. We used Weka implementations of these classifiers (Witten and Frank, 2000).

4.1 Features

The experiments use two types of features: (1) basic features of the nodes taken directly from the treebank (node attributes), and (2) derived features inspired by the annotation guidelines.

The basic features are the following (the first 4 are boolean, and 5 and 6 are nominal):

1. **is-noun**: true, if the node is a noun;
2. **is-root**: true, if the node is the root of the tree;
3. **is-coref-pronoun**: true, if the node is a coreferential pronoun;
4. **is-noncoref-pronoun**: true, if the node is a non-coreferential pronoun (in Czech, many pronouns are used in idiomatic expressions in which they do not have an coreferential function, e.g., *svého času*, lit. ‘in its (reflexive) time’, ‘some time ago’);
5. **SUBPOS**: detailed part of speech which differentiates between types of pronouns: personal, demonstrative, relative, etc.;
6. **functor**: type of dependency relations: MOD, MANN, ATT, OTHER.

The derived features are computed using the dependency information from the tectogrammatical level of the treebank and the surface order of the words corresponding to the nodes[5]. Also, we have used lists of forms of Czech pronouns that are used as weak pronouns, indexical expressions, pronouns with general meaning, or strong pronouns. All the derived features have boolean values:

7. **is-rightmost-dependent-of-the-verb**;
8. **is-rightside-dependent-of-the-verb**;
9. **is-leftside-dependent**;
10. **is-embedded-attribute**: true, if the node’s parent is not the root;
11. **has-repeated-lemma**: true, in case of nouns, when another node with the same lemma appears in the previous 10 sentences.
12. **is-in-canonical-order**;
13. **is-weak-pronoun**;
14. **is-indexical-expression**;
15. **is-pronoun-with-general-meaning**;
16. **is-strong-pronoun-with-no-prep**;

[3] In the PDT the intonation center is not annotated. However, the annotators were instructed to use their opinion where the IC is when they utter the sentence.

[4] We are grateful to our colleagues at the Charles University in Prague for providing us the experimental data before the PDT 2.0 official release.

[5] On the tectogramatical level in the PDT, the order of the nodes has been changed during the annotation process of the TFA attribute, so that all t items precede all f items. Our features use the surface order of the words corresponding to the nodes.

1.	The bearer of the IC (typically, the rightmost child of the verb)	f
2.	If IC is not on the rightmost child, everything after IC	t
3.	A left-side child of the verb (unless it carries IC)	t
4.	The verb and the right children of the verb before the f-node (cf. 1) that are canonically ordered	f
5.	Embedded attributes (unless repeated or restored)	f
6.	Restored nodes	t
7.	Indexical expressions (*já* I, *ty* you, *těd* now, *tady* here), weak pronouns, pronominal expressions with a general meaning (*někdo* somebody, *jednou* once) (unless they carry IC)	t
8.	Strong forms of pronouns not preceded by preposition (unless they carry IC)	t

Table 1: Annotation guidelines; IC = Intonation Center

4.2 Evaluation framework

In order to perform the evaluation, we randomly selected 101,054 instances (1/3 of the data) from all the instances, which represents our test set; the remaining 2/3 of the data we used as a training set. The same test set is used by all three classifiers. In our experiments we have not tweaked the features and thus we have not set aside a development set. In the test set 87% of the instances are nouns and 13% are pronouns. The t/f distribution in the test set is as follows: 58% of the instances are t, and 42% instances are f.

We have built models using decision trees (C4.5), bagging and rule-induction (Ripper) machine learning techniques to predict the Information Structure.

We have also implemented a deterministic, rule-based system that assigns t or f according to the annotation guidelines presented in Table 1. The rule-based system does not have access to what intonation center (IC) is.

The baseline simulates the preprocessing procedure used before the manual annotation of TFA attribute in the PDT, i.e., assigns always the class that has the most instances.

Our machine learning models are compared against the baseline and the rule-based system. As a metric we have used the Weighted Averaged F-score which is computed as follows:

```
%_f*F-score_f+%_t*F-score_t
```

The reason why we have chosen this metric (instead of Correctly Classified, for example) is that it gives a more realistic evaluation of the system, considering also the distribution of t and f items[6].

4.3 Results

The results of the experiment using all instances (nouns and pronouns) are shown in Table 2 in the second column. C4.5 and Bagging achieve the best performance improving on the results of the rule-based system by 6.99%.

The top of the decision tree generated by C4.5 in the training phase looks like this:

```
is-coref-pronoun = true
|    is-leftside-dependent = true
|    |    SUBPOS = ...
is-coref-pronoun = false
|    is-leftside-dependent = true
|    |    is-in-canonical-order = true
```

The overall tree has 129 leaves out of 161 nodes.

In order to achieve a better understanding of the difficulty of the task for nouns and pronouns, we considered evaluations on the following classes of instances:

- only nouns;
- nouns that are direct dependents of the verb (verb children);
- nouns that are not direct dependents of the verb (non-verb children);
- only pronouns;
- coreferential pronouns;
- non-coreferential pronouns.

We also wanted to investigate if the three classifiers perform differently with respect to different classes of instances (in which case we could have a general system, that uses more classifiers, and for certain classes of instances we would ‘trust’ a certain classifier, according to its performance on the development data).

[6] Consider, for example, the case in which the test set consists of 70% f items and 30% t items. The Baseline system would have as much as 70% correctly classified instances, just because the t/f distribution is as such. The Weighted Averaged F-score would be in this case 57.64% which is a more adequate value that reflects better the poorness of such a system.

Systems	nouns & pronouns	only nouns			only pronouns		
		all	verb children	non-verb children	all	coref	non-coref
Baseline	42.50	51.43	41.90	73.08	81.35	96.94	58.79
Rule-based	76.68	75.59	79.09	69.06	82.23	95.51	62.44
C4.5	82.04	79.98	80.38	73.87	93.77	97.25	68.60
Bagging	82.04	79.97	80.37	73.86	93.71	97.34	68.36
Ripper	81.78	79.88	80.31	73.86	93.55	97.35	68.36

Table 2: Overall results: Weighted Averaged F-score as percentage

Table 2, in columns three and onwards, shows the results on different classes of instances. The test set for each class of instances represents 1/3 randomly extracted instances from all instances in the data belonging to that class, in the same fashion as for the overall split.

The baseline is higher for some classes, yet the classifiers perform always better, even than the rule-based system, which for non-verb children performs worse than the baseline. However, the difference between the three classifiers is very small, and only in one case (for the coreferential pronouns) C4.5 is outperformed by Ripper.

To improve the results even more, there are two possibilities: either providing more training data, or considering more features. To investigate the effect of the size of the training data we have computed the learning curves for the three classifiers. Figure 1 shows the C4.5 learning curve for the overall experiment on nouns and pronouns; the learning curves for the other two classifiers are similar, and not included in the figure.

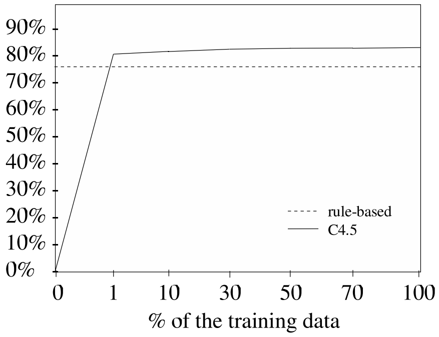


Figure 1: Learning curve for the C4.5 classifier

The curve is interesting, showing that after only 1% of the training set (1961 instances) C4.5 can already perform well, and adding more training data improves the F-score only slightly. To ensure the initial 1% aren't over-representative of the kind of IS phenomena, we experimented with different 1% parts of the training set, and the results were similar. We also did a 10-fold cross validation experiment on the training set, which resulted in a Weighted Averaged F-score of 82.12% for C4.5.

The slight improvement achieved by providing more data indicates that improvements are likely to come from using more features.

Table 3 shows the contribution of the two types of features (basic and derived) for the experiment with all instances (nouns and pronouns). For comparison we have displayed again the baseline and the rule-based system F-score.

Features System	Basic	Derived	All
C4.5	62.82	77.51	82.04
Bagging	62.83	77.50	81.99
Ripper	62.48	77.28	81.78
Rule-based	76.68		
Baseline	42.50		

Table 3: Contribution of different features. F-score given as a percentage.

The results show that the model trained only with basic features performs much better than the baseline, yet it is not as good as the rule-based system. However, removing the basic features completely and keeping only the derived features considerably lowers the score (by more than 4%). This indicates that adding more basic features (which are easy to obtain from the treebank) could actually improve the results.

The derived features, however, have the biggest impact on the performance of the classifiers. Yet, adding more sophisticated features that would help in this task (e.g., coreferentiality for nouns) is difficult because they cannot be computed reliably.

5 Conclusions

In this paper we investigated the problem of learning aspects of Information Structure from annotated data. We presented results from a study trying to verify whether Information Structure can be learned using mostly syntactic features. We used the Prague Dependency Treebank which is annotated with IS following the Praguian theory of Topic Focus Articulation. The results show that we can reliably identify t(opic) and f(ocus) with over 82% Weighted Averaged F-score while the baseline is at 42%.

Issues for further research include, on the one hand, a deeper investigation of the Topic-Focus Articulation in the Prague Dependency Treebank, by improving the feature set, considering also the distinction between contrastive and non-contrastive t items and, most importantly, by investigating how we can use the t/f annotation in PDT (and respectively our results) in order to detect the Topic/Focus partitioning of the whole sentence.

On the other hand, we want to benefit from the experience with the Czech data in order to create an English corpus annotated with Information Structure. We want to exploit a parallel English-Czech corpus available as part of the PDT, in order to extract correlations between different linguistic dimensions and Topic/Focus in the Czech data and investigate how they can be transferred to the English version of the corpus.

References

Eva Buránová, Eva Hajičová & Petr Sgall. 2000. *Tagging of very large corpora: Topic-Focus Articulation.* Proceedings of the 18th International Conference on Computational Linguistics, COLING 2000, 139-144.

František Čermák. 1997. *Czech National Corpus: A Case in Many Contexts.* International Journal of Corpus Linguistics, 2(2):181-197.

Jan Hajič. 1998. *Building a syntactically annotated corpus: The Prague Dependency Treebank.* Issues of valency and Meaning. Studies in Honor of Jarmila Panevová, ed. by E. Hajičová. Karolinum, Prague.

Eva Hajičová, Barbara Partee & Petr Sgall. 1998. *Topic-focus articulation, tripartite structures, and semantic content.* Studies in Linguistics and Philosiphy, 71 Dordrecht: Kluwer.

Eva Hajičová & Petr Sgall. 2001. *Topic-focus and salience.* Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, ACL 2001, 268-273. Toulose, France.

M. Halliday. 1967. *Notes on transitivity and theme in English, Part II.* Journal of Linguistic, 3:199-244.

Ivana Kruijff-Korbayová and Mark Steedman. 2003. *Discourse and Information Structure.* Journal of Logic, Language and Information 12:249-259. Kluwer, Amsterdam.

Petr Sgall. 1967. *Functional sentence perspective in a generative description.* Prague Studies in Mathematical Linguistics, 2:203-225.

Petr Sgall, Eva Hajičová & Jarmila Panevová. 1986. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects.* Reidel, Dordrecht.

Mark Steedman. 2000. *Information Structure and the syntax-phonology interface.* Linguistic Inquiry, 34:649-689.

Enrich Vallduví. 1990. *The information component.* Ph.D Thesis, University of Pennsylvania.

Enric Vallduví & Maria Vilkuna. 1998. *On rheme and kontrast.* Syntax and Semantics, Vol. 29: The Limits of Syntax, ed. by P. Culicover and L. McNally. Academic Press, San Diego.

Ian H. Witten & Eibe Frank. 2000. *Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, San Francisco.

Speech Recognition of Czech - Inclusion of Rare Words Helps

Petr Podveský and **Pavel Machek**
Institute of Formal and Applied Linguistics
Charles University
Prague, Czech Republic
{podvesky,machek}@ufal.mff.cuni.cz

Abstract

Large vocabulary continuous speech recognition of inflective languages, such as Czech, Russian or Serbo-Croatian, is heavily deteriorated by excessive out of vocabulary rate. In this paper, we tackle the problem of vocabulary selection, language modeling and pruning for inflective languages. We show that by explicit reduction of out of vocabulary rate we can achieve significant improvements in recognition accuracy while almost preserving the model size. Reported results are on Czech speech corpora.

1 Introduction

Large vocabulary continuous speech recognition of inflective languages is a challenging task for mainly two reasons. Rich morphology generates huge number of forms which are not captured by limited-size dictionaries, and therefore leads to worse recognition results. Relatively free word order admits enormous number of word sequences and thus impoverishes n-gram language models. In this paper we are concerned with the former issue.

Previous work which deals with excessive vocabulary growth goes mainly in two lines. Authors have either decided to break words into sub-word units or to adapt dictionaries in a multi-pass scenario. On Czech data, (Byrne et al., 2001) suggest to use linguistically motivated recognition units. Words are broken down to stems and endings and used as the recognition units in the first recognition phase. In the second phase, stems and endings are concatenated. On Serbo-Croatian, (Geutner et al., 1998) also tested morphemes as the recognition units. Both groups of authors agreed that this approach is not beneficial for speech recognition of inflective languages. Vocabulary adaptation, however, brought considerable improvement. Both (Icring and Psutka, 2001) on Czech and (Geutner et al., 1998) on Serbo-Croatian reported substantial reduction of word error rate. Both authors followed the same procedure. In the first pass, they used a dictionary composed of the most frequent words. Generated lattices were then processed to get a list of all words which appeared in them. This list served as a basis for a new adapted dictionary into which morphological variants were added.

It can be concluded that large corpora contain a host of words which are ignored during estimation of language models used in first pass, despite the fact that these rare words can bring substantial improvement. Therefore, it is desirable to explore how to incorporate rare or even unseen words into a language model which can be used in a first pass.

2 Language Model

Language models used in a first pass of current speech recognition systems are usually built in the following way. First, a text corpus is acquired. In case of broadcast news, a newspaper collection or news transcriptions are a good source. Second, most frequent words are picked out to form a dictionary. Dictionary size is typically in tens of thousand words. For English, for example, dictionaries of size

Proceedings of the ACL Student Research Workshop, pages 121–126,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

of 60k words sufficiently cover common domains. (Of course, for recognition of entries listed in the Yellow pages, such limited dictionaries are clearly inappropriate.) Third, an n-gram language model is estimated. In case of Katz back-off model, the conditional bigram word probability is estimated as

$$P_1(w_i|w_{i-1}) = \begin{cases} \widetilde{P}(w_i|w_{i-1}) \text{ if } C(w_{i-1}, w_i) > k \\ BO(w_{i-1}) \cdot \widetilde{P}(w_i) \quad \text{otherwise} \end{cases} \tag{1}$$

where $\widetilde{P}$ represents a smoothed probability distribution, $BO()$ stands for the back-off weight, and $C(.)$ denotes the count of its argument. Back-off model can be also nicely viewed as a finite state automaton as depicted in Figure 1.

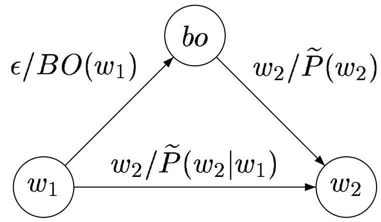


Figure 1: A fragment of a bigram back-off model represented as a finite-state automaton.

To alleviate the problem of a high OOV, we suggest to gather supplementary words and add them into the model in the following way.

$$P(w_i|w_{i-1}) = \begin{cases} P_1(w_i|w_{i-1}) & w_i \in D \\ BO(w_{i-1}) \cdot Q(w_i) & w_i \in S \end{cases} \tag{2}$$

$P_1()$ refers to the regular back-off model, D denotes the regular dictionary from which the back-off model was estimated, S is the supplementary dictionary which does not overlap with D.

Several sources can be exploited to obtain supplementary dictionaries. Morphology tools can derive words which are close to those observed in corpus. In such a case, $Q(w_i)$ can be set as a constant function and estimated on held-out data to maximize recognition accuracy.

$$Q(w_i) = const \quad \text{for } w_i \text{ generated by morphology} \tag{3}$$

Having prior domain knowledge, new words which are expected to appear in audio recordings might be collected and added into S. Consider an example of transcribing an ice-hockey tournament. Names of new players are desirably in the vocabulary. Another source of S are the words which fell below the selection threshold of D. In large corpora, there are hundreds of thousands words which are omitted from the estimated language model. We suggest to put them into S. As it turned out, unigram probability of these words is very low, so it is suitable to increase their score to make them competitive with other words in D during recognition. $Q(w_i)$ is then computed as

$$Q(w_i) = \textit{shift} \cdot f(w_i) \tag{4}$$

where $f(w_i)$ refers to the relative frequency of w_i in a given corpus, *shift* denotes a shifting factor which should be tuned on some held-out data.

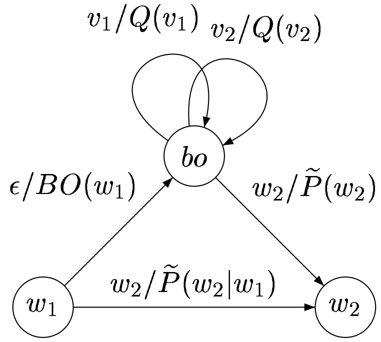


Figure 2: A fragment of a bigram back-off model injected by a supplementary dictionary

Note that the probability of a word given its history is no longer proper probability. It does not adds up to one. We decided not to normalize the model for two reasons. First, we used a decoder which searches for the best path using Viterbi criterion, so there's no need for normalization. Second, normalization would have involved recomputing all back-off model weights and could also enforce re-tuning of the language model scaling factor. To rule out any variation which the re-tuning of the scaling factor could bring, we decided not to normalize the new model.

In finite-state representation, injection of a new dictionary was implemented as depicted in Figure 2. Supplementary words form a loop in the back-off state.

3 Experiments

We have evaluated our approach on two corpora, Czech Broadcast News and the Czech portion of MALACH data.

3.1 Czech Broadcast News Data

The Czech Broadcast News (Radová et al., 2004) is a collection of both radio and TV news in Czech. Weather forecast, traffic announcements and sport news were excluded from this corpus. Our training portion comprises 22 hours of speech. To tune the language model scaling factor and additional LM parameters, we set aside 100 sentences. The test set consists of 2500 sentences.

We used the HTK toolkit (Young et al., 1999) to extract acoustic features from sampled signal and to estimate acoustic models. As acoustic features we used 12 Mel-Frequency Cepstral Coefficients plus energy and delta and delta-delta features. We trained a triphone acoustic model with tied mixtures of continuous density Gaussians.

As a LM training corpus we exploited a collection of newspaper articles from the Lidové Noviny (LN) newspaper. This collection was published as a part of the Prague Dependency Treebank by LDC (Hajič et al., 2001). This corpus contains 33 million tokens. Its vocabulary contains more than 650k word forms. OOV rates are displayed in Table 1.

Dict. size	OOV
60k	8.27%
80k	6.92%
124k	5.20%
371k	2.23%
658k	1.63%

Table 1: OOV rate of transcriptions of the test data. Dictionaries contain the most frequent words.

As can be readily observed, moderate-size vocabularies don't sufficiently cover the test data transcriptions. Therefore they are one of the major sources of poor recognition performance.

The baseline language model was estimated from 60k most frequent words. It was a bigram Katz back-off model with Knesser-Ney smoothing pruned by the entropy-based method (Stolcke, 1998).

As the supplementary dictionary we took the rest of words from the LN corpus. To learn the impact of injection of infrequent words, we carried out two experiments.

First, we built a uniform loop which was injected into the back-off model. The uniform distribution was tuned on the held-out data. Tuning of this constant is displayed in Table 2.

Uniform scale	WER
12	18.89%
11	18.68%
10	18.40%
9	21.00%

Table 2: Tuning of uniform distribution on the held-out set. *WER* denotes the word error rate.

Second, we took relative frequencies multiplied by a shift coefficient as the injected model scores. This shift coefficient was again tuned on held-out data as shown in Table 3.

Unigram shift	WER
no shift	19.52%
e^3	18.54%
e^4	17.91%
e^5	18.75%

Table 3: Tuning of the shift coefficient of unigram model on the held-out set.

Then, we took the best parameters and used them for recognition of the test data. Recognition results are depicted in Figure 4. The injection of supplementary words helped decrease both recognition word error rate and oracle word error rate. By oracle WER is meant WER of the path, stored in the generated lattice, which best matches the utterance regardless the scores. In other words, oracle WER gives us a bound on how well can we get by tuning scores in a given lattice. Injection of shifted unigram model brought relative improvement of 13.6% in terms of WER over the 60k baseline model. Uniform injection brought also significant improvement despite its simplicity. Indeed, we observed more than 10% relative improvement in terms of WER. In terms of oracle WER, unigram injection brought more than 30% relative improvement.

Model	WER	OWER
Baseline 60k	29.17%	15.90%
Baseline 80k	27.44%	14.31%
60k + Uniform injection	26.12%	11.10%
60k + Unigram injection	25.21%	11.03%

Table 4: Evaluation on 2500 test sentences. *OWER* stands for the oracle error rate.

It's worthwhile to mention the model size, since it could be argued that the improvement was achieved by an enormous increase of the model. We decided to measure the model size using two factors. The disk space occupied by the language model and the disk space taken up by the so-called *CLG*. By *CLG* we mean a transducer which maps triphones to words augmented with the model scores. This transducer represents the search space investigated during recognition. More details on transducers in speech recognition can be found in (Mohri et al., 2002). Table 5 summarizes the sizes of the evaluated models.

Model	CLG size	G size
Baseline 60k	399MB	106MB
60k + Uniform	405MB	115MB
60k + Unigram	405MB	115MB
Baseline 80k	441MB	116MB

Table 5: Model size comparison measured in disk space. *G* denotes a language model compiled as a finite-state automaton. *CLG* denotes transducer mapping triphones to words augmented with model scores.

Injection of supplementary words increased the model size only slightly. To see the difference in the size of injected models and traditionally built ones, we constructed a model of 80k most frequent words and pruned with the same threshold as the 60k LM. Not only did this 80k model give worse recognition results, but it also proved to be bigger.

3.2 MALACH Data

The next data we tested our approach on was the Czech portion of the MALACH corpus (`http://www.clsp.jhu.edu/research/malach`). MALACH is a multilingual audio-visual corpus. It contains recordings of survivors of World War II talking about war events. 600 people spoke in Czech, but only 350 recordings had been digitized till end of 2003. The interviewer and the interviewee had separate microphones, and were recorded on separate stereo channels. Recordings were stored in the MPEG-1 format. Average length of a testimony is 1.9 hours.

30 minutes from each testimony were transcribed and used as training data. 10 testimonies were transcribed completely and used for testing. The acoustic model used 15-dimensional PLP cepstral features, sampled at 10 msec. Modeling was done using the HTK Toolkit.

The baseline language model was estimated from transcriptions of the survivors' testimonies. We worked with the standardized version of the transcriptions. More details regarding the Czech portion of the MALACH data can be found in (Psutka et al., 2004). Transcriptions are 610k words long and the entire vocabulary comprises 41k words. We refer to this corpus as *TR_41k*.

To obtain a supplementary vocabulary, we used Czech morphology tools (Hajič and Vidová-Hladká, 1998). Out of 41k words we generated 416k words which were the inflected forms of the observed words in the corpus. Note that we posed restrictions on the generation procedure to avoid obsolete, archaic and uncommon expressions. To do so, we ran a Czech tagger on the transcriptions and thus obtained a list of all morphological tags of observed forms. The morphological generation was then confined to this set of tags.

Since there is no corpus to train unigram scores of generated words on, we set the LM score of the generated forms to a constant.

The transcriptions are not the only source of text data in the MALACH project. (Psutka et al., 2004) searched the Czech National Corpus (CNC) for sentences which are similar to the transcriptions. This additional corpus contains almost 16 million words, 330k types. CNC vocabulary overlaps to a large extent with TR vocabulary. This fact is not surprising since the selection criterion was based on a lemma unigram probability. Table 6 summarizes OOV rates of several dictionaries.

We estimated several language models. The baseline models are pruned bigram back-off models with Knesser-Ney smoothing. The baseline word error

Dictionary		OOV
Name	Size	
TR41k	41k	5.07 %
TR41k + Morph416k	416k	2.74 %
TR41k + CNC60k	79k	3.04 %
TR41k + CNC100k	114k	2.62 %
TR41k + CNC160k	171k	2.25%
TR41k + CNC329k	337k	1.76 %
All together	630k	1.46 %

Table 6: OOV for several dictionaries. *TR*, *CNC* denote the transcriptions, the Czech National Corpus, respectively. *Morph* refers to the dictionary generated by the morphology tools from from *TR*. Numbers in the dictionary names represent the dictionary size.

rate of the model built solely from transcriptions was 37.35%. We injected constant loop of morphological variants into this model. In terms of text coverage, this action reduced OOV from 5.07% to 2.74%. In terms of recognition word error rate, we observed a relative improvement of 3.5%.

In the next experiment we took as the baseline LM a linear interpolation of the LM built from transcriptions and a model estimated from the CNC corpus. Into this model, we injected a unigram loop of all the available words. That is the rest of words from the CNC corpus with unigram scores and words provided by morphology which were not already in the model. Table 7 summarizes the achieved WER and oracle WER. Given the fact that the injection only slightly reduced the OOV rate, a small relative reduction of 2.3% matched our expectations.

Model	Acc	OAcc
TR41k	37.35%	14.40%
TR41k + Uniform_Morph	36.06%	12.48%
TR41k + CNC_100k	34.47%	11.95%
TR41k + CNC_100k + Inj	33.67%	10.79%
TR41k + CNC_160k	34.19%	11.65%

Table 7: Word error rate and oracle WER for baseline and injected models. *Uniform_Morph* refers to the constant uniform loop of the morphology-generated words. *Inj* denotes the loop of the rest of words of the CNC corpus and the morphology-generated words.

To learn how the injection affected model size, we measured size of the language model automaton and the optimized triphone-to-word transducer. As in the case of the LN corpus, injection increased the model size only moderately. Sizes of the models are shown in Table 8.

model	CLG	G
TR41k	38MB	5.6MB
TR41k + Morph	54MB	11MB
TR41k + CNC_100k	283MB	53MB
TR41k + CNC_100k + Inj	307MB	61MB
TR41k + CNC_160k	312MB	59MB

Table 8: Disk usage of tested models. *G* refers to a language model compiled into an automaton, *CLG* denotes triphone-to-word transducer. *CNC* and *Morph* refer to a LM estimated from transcriptions and the Czech National Corpus, respectively. Morph represents the loop of words generated by morphology. *Inj* is the loop of all words from *CNC* which were not included in *CNC* language model, moreover, *Inj* also contains words generated by the morphology.

4 Conclusion

In this paper, we have suggested to inject a loop of supplementary words into the back-off state of a first-pass language model. As it turned out, addition of rare or morphology-generated words into a language model can considerably decrease both recognition word error rate and oracle WER in single recognition pass. In the recognition of Czech Broadcast News, we achieved 13.6% relative improvement in terms of word error rate. In terms of oracle error rate, we observed more than 30% relative improvement. On the MALACH data, we attained only marginal word error rate reduction. Since the text corpora already covered the transcribed speech relatively well, a smaller OOV reduction translated into a smaller word error rate reduction. In the near future, we would like to test our approach on agglutinative languages, where the problems with high OOV are even more challenging. We would also like to experiment with more complex language models.

5 Acknowledgements

We would like to thank our colleagues from the University of Western Bohemia for providing us with acoustic models. This work has been done under the support of the project of the Ministry of Education of the Czech Republic No. MSM0021620838 and the grant of the Grant Agency of the Charles University (GAUK) No. 375/2005.

References

W. Byrne, J. Hajič, P. Ircing, F. Jelinek, S. Khudanpur, P. Krbec, and J. Psutka. 2001. On large vocabulary continuous speech recognition of highly inflectional language - Czech. In *Eurospeech 2001*.

P. Geutner, M. Finke, and P. Scheytt. 1998. Adaptive Vocabulariesfor Transcribing Multilingual Broadcast News. In *ICASSP*, Seattle, Washington.

Jan Hajič and Barbora Vidová-Hladká. 1998. Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. In *Proceedings of the Conference COLING ACL '98*, pages 483-490, Mountreal, Canada.

Jan Hajič, Eva Hajičová, Petr Pajas, Jarmila Panevová, Petr Sgall, and Barbora Vidová-Hladká. 2001. Prague dependency treebank 1.0. Linguistic Data Consortium (LDC), catalog number LDC2001T10.

P. Icring and J. Psutka. 2001. Two-Pass Recognition of Czech Speech Using Adaptive Vocabulary. In *TSD*, Železnaá Ruda, Czech Republic.

M. Mohri, F. Pereira, and M. Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16:69-88.

J. Psutka, P. Ircing, V. Radova, and J. V. Psutka. 2004. Issues in annotation of the Czech spontaneous speech corpus in the MALACH project. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

Vlasta Radová, Josef Psutka, Luděk Müller, William Byrne, J.V. Psutka, Pavel Ircing, and Jindřich Matoušek. 2004. Czech broadcast news speech. Linguistic Data Consortium (LDC), catalog number LDC2004S01.

A. Stolcke. 1998. Entropy-based pruning of backoff language models. In *In Proceedings of the ARPA Workshop on Human Language Technology.*

S. Young et al. 1999. *The HTK Book.* Entropic Inc.

Using bilingual dependencies to align words in Enlish/French parallel corpora

Sylwia Ozdowska
ERSS - CNRS & Université de Toulouse le Mirail
5 allées Antonio Machado
31058 Toulouse Cedex France
ozdowska@univ-tlse2.fr

Abstract

This paper describes a word and phrase alignment approach based on a dependency analysis of French/English parallel corpora, referred to as alignment by "syntax-based propagation." Both corpora are analysed with a deep and robust dependency parser. Starting with an anchor pair consisting of two words that are translations of one another within aligned sentences, the alignment link is propagated to syntactically connected words.

1 Introduction

It is now an acknowledged fact that alignment of parallel corpora at the word and phrase level plays a major role in bilingual linguistic resource extraction and machine translation. There are basically two kinds of systems working at these segmentation levels: the most widespread rely on statistical models, in particular the IBM ones (Brown *et al.*, 1993); others combine simpler association measures with different kinds of linguistic information (Arhenberg *et al.*, 2000; Barbu, 2004). Mainly dedicated to machine translation, purely statistical systems have gradually been enriched with syntactic knowledge (Wu, 2000; Yamada & Knight, 2001; Ding *et al.*, 2003; Lin & Cherry, 2003). As pointed out in these studies, the introduction of linguistic knowledge leads to a significant improvement in alignment quality.

In the method described hereafter, syntactic information is the kernel of the alignment process. Indeed, syntactic dependencies identified on both sides of English/French bitexts with a parser are used to discover correspondences between words. This approach has been chosen in order to capture frequent alignments as well as sparse and/or corpus-specific ones. Moreover, as stressed in previous research, using syntactic dependencies seems to be particularly well suited to coping with the problem of linguistic variation across languages (Hwa *et al.*, 2002). The implemented procedure is referred to as "syntax-based propagation".

2 Starting hypothesis

The idea is to make use of dependency relations to align words (Debili & Zribi, 1996). The reasoning is as follows (Figure 1): if there is a pair of anchor words, i.e. if two words $w1_i$ (*community* in the example) and $w2_m$ (*communauté*) are aligned at the sentence level, and if there is a dependency relation between $w1_i$ (*community*) and $w1_j$ (*ban*) on the one hand, and between $w2_m$ (*communauté*) and $w2_n$ (*interdire*) on the other hand, then the alignment link is propagated from the anchor pair (*community, communauté*) to the syntactically connected words (*ban*, *interdire*).

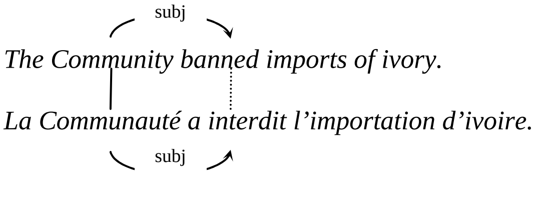

subj
The Community banned imports of ivory.
La Communauté a interdit l'importation d'ivoire.
subj

Figure 1. Syntax-based propagation

Proceedings of the ACL Student Research Workshop, pages 127–132,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

We describe hereafter the overall design of the syntax-based propagation process. We present the results of applying it to three parsed English/French bitexts and compare them to the baseline obtained with the giza++ package (Och & Ney, 2000).

3 Corpora and parsers

The syntax-based alignment was tested on three parallel corpora aligned at the sentence level: INRA, JOC and HLT. The first corpus was compiled at the National Institute for Agricultural Research (INRA)[1] to enrich a bilingual terminology database used by translators. It comprises 6815 aligned sentences[2] and mainly consists of research papers and popular-science texts.

The JOC corpus was made available in the framework of the ARCADE project, which focused on the evaluation of parallel text alignment systems (Veronis & Langlais, 2000). It contains written questions on a wide variety of topics addressed by members of the European Parliament to the European Commission, as well as the corresponding answers. It is made up of 8765 aligned sentences.

The HLT corpus was used in the evaluation of word alignment systems described in (Mihalcea & Pederson, 2003). It contains 447 aligned sentences from the Canadian Hansards (Och & Ney, 2000).

The corpus processing was carried out by a French/English parser, SYNTEX (Fabre & Bourigault, 2001). SYNTEX is a dependency parser whose input is a POS tagged[3] corpus — meaning each word in the corpus is assigned a lemma and grammatical tag. The parser identifies dependencies in the sentences of a given corpus, for instance subjects and direct and indirect objects of verbs. The parsing is performed independently in each language, yet the outputs are quite homogeneous since the syntactic dependencies are identified and represented in the same way in both languages.

In addition to parsed English/French bitexts, the syntax-based alignment requires pairs of anchor words be identified prior to propagation.

4 Identification of anchor pairs

To derive a set of words that are likely to be useful for initiating the propagation process, we implemented a widely used method of co-occurrence counts described notably in (Gale & Church, 1991; Ahrenberg *et al*., 2000). For each source (*w1*) and target (*w2*) word, the Jaccard association score is computed as follows:

$$j(w1, w2) = f(w1, w2)/f(w1) + f(w2) - f(w1, w2)$$

The Jaccard is computed provided the number of overall occurrences of *w1* and *w2* is higher than 4, since statistical techniques have proved to be particularly efficient when aligning frequent units. The alignments are filtered according to the j(*w1*, *w2*) value, and retained if this value was 0.2 or higher. Moreover, two further tests based on cognate recognition and mutual correspondence condition are applied.

The identification of anchor pairs, consisting of words that are translation equivalents within aligned sentences, combines both the projection of the initial lexicon and the recognition of cognates for words that have not been taken into account in the lexicon. These pairs are used as the starting point of the propagation process[4].

Table 1 gives some characteristics of the corpora.

	INRA	JOC	HLT
aligned sentences	6815	8765	477
anchor pairs	4376	60762	996
w1/source sentence	21	25	15
w2/target sentence	24	30	16
anchor pairs/sentence	6.38	6.93	2.22

Table 1. Identification of anchor pairs

5 Syntax-based propagation

5.1 Two types of propagation

The syntax-based propagation may be performed in two different directions, as a given word is likely to be both governor and dependent with respect to other words. The first direction starts with dependent anchor words and propagates the alignment link to the governors (Dep-to-Gov propagation). The Dep-to-Gov propagation is *a priori* not ambiguous since one dependent is governed at

[1] We are grateful to A. Lacombe who allowed us to use this corpus for research purposes.

[2] The sentence-level alignment was performed using Japa (http://www.rali.iro.umontreal.ca).

[3] The French and English versions of Treetagger (http://www.ims.uni-stuttgart.de) are used.

[4] The process is not iterative up to date so the number of words it allows to align depends on the initial number of anchor words per sentence.

most by one word. Thus, there is just one relation on which the propagation can be based. The second direction goes the opposite way: starting with governor anchor words, the alignment link is propagated to their dependents (Gov-to-Dep propagation). In this case, several relations that may be used to achieve the propagation are available, as it is possible for a governor to have more than one dependent. So the propagation is potentially ambiguous. The ambiguity is particularly widespread when propagating from head nouns to their nominal and adjectival dependents. In Figure 2, there is one occurrence of the relation pcomp in English and two in French. Thus, it is not possible to determine *a priori* whether to propagate using the relations mod/pcomp2, on the one hand, and pcomp1/pcomp2', on the other hand, or mod/pcomp2' and pcomp1/pcomp2. Moreover, even if there is just one occurrence of the same relation in each language, it does not mean that the propagation is of necessity performed through the same relation, as shown in Figure 3.

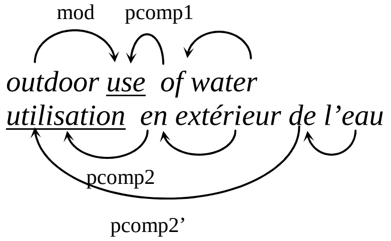


Figure 2. Ambiguous propagation from head nouns

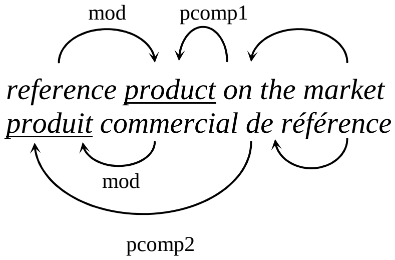

mod
pcomp1
reference product on the market
produit commercial de référence
mod
pcomp2

Figure 3. Ambiguous propagation from head nouns

In the following sections, we describe the two types of propagation. The propagation patterns we rely on are given in the form CDep-rel-CGov, where CDep is the POS of the dependent, rel is the dependency relation and CGov, the POS of the governor. The anchor element is underlined and the one aligned by propagation is in bold.

5.2 Alignment of verbs

Verbs are aligned according to eight propagation patterns.

DEP-TO-GOV PROPAGATION TO ALIGN GOVERNOR VERBS. The patterns are: Adv-mod-**V** (1), N-subj-**V** (2), N-obj-**V** (3), N-pcomp-**V** (4) and V-pcomp-**V** (5).
(1) The net ***is*** *then* ***hauled*** *to the shore.*
Le filet ***est*** *ensuite* ***halé*** *à terre.*
(2) The fish ***are*** *generally* ***caught*** *when they migrate from their feeding areas.*
Généralement les poissons ***sont capturés*** *quand ils migrent de leur zone d'engraissement.*
(3) Most of the young shad ***reach*** *the sea.*
La plupart des alosons ***gagne*** *la mer.*
(4) The eggs are very small and ***fall*** *to the bottom.*
Les oeufs de très petite taille ***tombent*** *sur le fond.*
(5) X is a model which ***was designed*** *to stimulate…*
X est un modèle qui ***a été conçu*** *pour stimuler…*

GOV-TO-DEP PROPAGATION TO ALIGN DEPENDENT VERBS. The alignment links are propagated from the dependents to the verbs using three propagation patterns: **V**-pcomp-V (1), **V**-pcomp-N (2) and **V**-pcomp-Adj (3).
(1) Ploughing tends to ***destroy*** *the soil microaggregated structure.*
Le labour tend à ***rompre*** *leur structure microagrégée.*
(2) The capacity to ***colonize*** *the digestive mucosa…*
L'aptitude à ***coloniser*** *le tube digestif…*
(3) An established infection is impossible to ***control****.*
Toute infection en cours est impossible à ***maîtriser****.*

5.3 Alignment of adjectives and nouns

The two types of propagation described in section 5.2 for use with verbs are also used to align adjectives and nouns. However, these latter categories cannot be treated in a fully independent way when propagating from head noun anchor words in order to align the dependents. The syntactic structure of noun phrases may be different in English and French, since they rely on a different type of composition to produce compounds and on the same one to produce free noun phrases. Thus, the potential ambiguity arising from the Gov-to-Dep propagation from head nouns mentioned in section 5.1

may be accompanied by variation phenomena affecting the category of the dependents. For instance, a noun may be rendered by an adjective, or vice versa: *tax treatment profits* is translated by *traitement fiscal des bénéfices*, so the noun *tax* is in correspondence with the adjective *fiscal*. The syntactic relations used to propagate the alignment links are thus different.

In order to cope with the variation problem, the propagation is performed regardless of whether the syntactic relations are identical in both languages, and regardless of whether the POS of the words to be aligned are the same. To sum up, adjectives and nouns are aligned separately of each other by means of Dep-to-Gov propagation or Gov-to-Dep propagation provided that the governor is not a noun. They are not treated separately when aligning by means of Gov-to-Dep propagation from head noun anchor pairs.

Dep-to-Gov propagation to align adjectives. The propagation patterns involved are: Adv-mod-**Adj** (1), N-pcomp-**Adj** (2) and V-pcomp-**Adj** (3).

(1) The white cedar exhibits a very ***common*** *physical defect.*
Le Poirier-pays présente un défaut de forme très ***fréquent****.*
(2) The area presently ***devoted*** *to agriculture represents…*
La surface actuellement ***consacrée*** *à l'agriculture représenterait…*
(3) Only four plots were ***liable*** *to receive this input.*
Seulement quatre parcelles sont ***susceptibles*** *de recevoir ces apports.*

Dep-to-Gov propagation to align nouns. Nouns are aligned according to the following propagation patterns: Adj-mod-**N** (1), N-mod-**N**/N-pcomp-**N** (2), N-pcomp-**N** (3) and V-pcomp-**N** (4).

(1) Allis shad remain on the continental ***shelf****.*
La grande alose reste sur le ***plateau*** *continental.*
(2) Nature of micropollutant ***carriers****.*
La nature des ***transporteurs*** *des micropolluants.*
(3) The ***bodies*** *of shad are generally fusiform.*
Le ***corps*** *des aloses est généralement fusiforme.*
(4) ***Ability*** *to react to light.*
Capacité *à réagir à la lumière.*

Unambiguous Gov-to-Dep propagation to align nouns. The propagation is not ambiguous when dependent nouns are not governed by a noun. This is the case when considering the following three propagation patterns: **N**-subj|obj-V (1), **N**-pcomp-V (2) and **N**-pcomp-Adj (3).

(1) The ***caterpillars*** *can inoculate the* ***fungus****.*
Les ***chenilles*** *peuvent inoculer le* ***champignon****.*
(2) The roots are placed in ***tanks****.*
Les racines sont placées en ***bacs****.*
(3) ...a fungus responsible for ***rot****.*
... un champignon responsable de la ***pourriture****.*

Potentially ambiguous Gov-to-Dep propagation to align nouns and adjectives. Considering the potential ambiguity described in section 5.1, the algorithm which supports Gov-to-Dep propagation from head noun anchor words (*n1, n2*) takes into account three situations which are likely to occur.

First, each of *n1* and *n2* has only one dependent, respectively *dep1* and *dep2,* involving one of the mod or pcomp relation; *dep1* and *dep2* are aligned.

the drained whey
le lactosérum d'égouttage
⇒ (***drained, égouttage***)

Second, *n1* has one dependent *dep1* and *n2* several $\{dep2_1, dep2_2, \ldots, dep2_n\}$, or vice versa. For each $dep2_i$, check if one of the possible alignments has already been performed, either by propagation or anchor word spotting. If such an alignment exists, remove the others ($dep1, dep2_k$) such that k ≠ i, or vice versa. Otherwise, retain all the alignments ($dep1, dep2_i$), or vice versa, without resolving the ambiguity.

stimulant substances which are absent from…
substances solubles stimulantes absentes de…
(*stimulant*, {*soluble, stimulant, absent*})
already_aligned(*stimulant, stimulant*) = 1
⇒ (***stimulant***, ***stimulant***)

Third, both *n1* and *n2* have several dependents, $\{dep1_1, dep1_2, \ldots, dep1_m\}$ and $\{dep2_1, dep2_2, \ldots, dep2_n\}$ respectively. For each $dep1_i$ and each $dep2_j$, check if one/several alignments have already been performed. If such alignments exist, remove all the alignments ($dep1_k$, $dep2_l$) such that k ≠ i or l ≠ j. Otherwise, retain all the alignments ($dep1_i$, $dep2_j$) without resolving the ambiguity.

unfair trading practices
pratiques commerciales déloyales
(*unfair*, {*commercial, déloyal*})
(*trading*, {*commercial, déloyal*})
already_aligned(*unfair, déloyal*) = 1

⇒ (***unfair***, ***déloyal***)
⇒ (***trading***, ***commercial***)

a big rectangular *<u>net</u>, which is lowered…*
un vaste *<u>filet</u> rectangulaire immergé…*
(*big*, {*vaste, rectangulaire, immergé*})
(*rectangular*, {*vaste, rectangulaire, immergé*})
already_aligned(*rectangular*, *rectangulaire*) = 1
⇒ (***rectangular, rectangulaire***)
⇒ (*big*, {*vaste, immergé*})

The implemented propagation algorithm has two major advantages: it permits the resolution of some alignment ambiguities, taking advantage of alignments that have been previously performed. This algorithm also allows the system to cope with the problem of non-correspondence between English and French syntactic structures and makes it possible to align words using various syntactic relations in both languages, even though the category of the words under consideration is different.

5.4 Comparative evaluation

The results achieved using the syntax-based alignment (sba) are compared to those obtained with the baseline provided by the IBM models implemented in the giza++ package (Och & Ney, 2000) (Table 2 and Table 3). More precisely, we used the intersection of IBM-4 Viterbi alignments for both translation directions. Table 2 shows the precision assessed against a reference set of 1000 alignments manually annotated in the INRA and the JOC corpus respectively. It can be observed that the syntax-based alignment offers good accuracy, similar to that of the baseline.

	INRA		**JOC**	
	sba	**giza++**	**sba**	**giza++**
Precision	0.93	0.96	0.95	0.94

Table 2. sba ~ giza++: INRA & JOC

More complete results (precision, recall and f-measure) are presented in Table 3. They have been obtained using reference data from an evaluation of word alignment systems (Mihalcea & Pederson, 2003). It should be noted that the figures concerning the syntax-based alignment were assessed in respect to the annotations that do not involve empty words, since up to now we focused only on content words. Whereas the baseline precision[5] for the HLT corpus is comparable to the one reported in Table 2, the syntax-based alignment score decreases. Moreover, the difference between the two approaches is considerable with regard to the recall. This may be due to the fact that our syntax-based alignment approach basically relies on isomorphic syntactic structures, i.e. in which the two following conditions are met: i) the relation under consideration is identical in both languages and ii) the words involved in the syntactic propagation have the same POS. Most of the cases of non-isomorphism, apart from the ones presented section 5.1, are not taken into account.

	HLT	
	sba	**giza++**
Precision	0.83	0.95
Recall	0.58	0.85
F-measure	0.68	0.89

Table 3. sba ~ giza++: HLT

6 Discussion

The results achieved by the syntax-based propagation method are quite encouraging. They show a high global precision rate — 93% for the INRA corpus and 95% for the JOC — comparable to that reported for the giza++ baseline system. The figures vary more from the HLT reference set. One possible explanation is the fact that the gold standard has been established according to specific annotation criteria. Indeed, the HLT project concerned above all statistical alignment systems aiming at language modelling for machine translation. In approaches such as Lin and Cherry’s (2003), linguistic knowledge is considered secondary to statistical information even if it improves the alignment quality. The syntax-based alignment approach was designed to capture both frequent alignments and those involving sparse or corpus-specific words as well as to cope with the problem of non-correspondance across languages. That is why we chose the linguistic knowledge as the main information source.

[5] Precision, recall and f-measure reported by Och and Ney (2003) for the intersection of IBM-4 Viterbi alignments from both translation directions.

7 Conclusion

We have presented an efficient method for aligning words in English/French parallel corpora. It makes the most of dependency relations to produce highly accurate alignments when the same propagation pattern is used in both languages, i.e. when the syntactic structures are identical, as well as in cases of noun/adjective transpositions, even if the category of the words to be aligned varies (Ozdowska, 2004). We are currently pursuing the study of non-correspondence between syntactic structures in English and French. The aim is to determine whether there are some regularities in the rendering of specific English structures into given French ones. If variation across languages is subject to such regularities, as assumed in (Dorr, 1994; Fox, 2002; Ozdowska & Bourigault, 2004), the syntax-based propagation could then be extended to cases of non-correspondence in order to improve recall.

References

Ahrenberg L., Andersson M. & Merkel M. 2000. A knowledge-lite approach to word alignment. In Véronis J. (Ed.), *Parallel Text Processing: Alignment and Use of Translation Corpora*, Dordrecht: Kluwer Academic Publishers, pp. 97-138.

Barbu A. M. 2004. Simple linguistic methods for improving a word alignment algorithm. In *Actes de la Conférence JADT.*

Brown P., Della Pietra S. & Mercer R. 1993. The mathematics of statistical machine translation: parameter estimation. In *Computational Linguistics*, 19(2), pp. 263-311.

Debili F. & Zribi A. 1996. Les dépendances syntaxiques au service de l'appariement des mots. In *Actes du 10ème Congrès RFIA.*

Ding Y., Gildea D. & Palmer M. 2003. An Algorithm for Word-Level Alignment of Parallel Dependency Trees. In *Proceedings of the 9th MT Summit of International Association of Machine Translation*.

Dorr B. 1994. Machine translation divergences: a formal description and proposed solution. In *Computational Linguistics*, 20(4), pp. 597-633.

Fabre C. & Bourigault D. 2001. Linguistic clues for corpus-based acquisition of lexical dependencies. In *Proceedings of the Corpus Linguistic Conference*.

Fox H. J. 2002. Phrasal Cohesion and Statistical Machine Translation. In *Proceedings of EMNLP-02*, pp. 304-311.

Gale W. A. & Church K. W. 1991. Identifying Word Correspondences in Parallel Text. In *Proceedings of the DARPA Workshop on Speech and Natural Language*.

Hwa R., Resnik P., Weinberg A. & Kolak O. 2002. Evaluating Translational Correspondence Using Annotation Projection. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics*.

Lin D. & Cherry C. 2003. ProAlign: Shared Task System Description. In *HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond.*

Mihalcea R. & Pedersen T. 2003. An Evaluation Exercise for Word Alignment. In *HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*.

Och F. Z. & Ney H., 2003. A Systematic Comparison of Various Statistical Alignment Models. In *Computational Linguistics*, 29(1), pp. 19-51.

Ozdowska S. 2004. Identifying correspondences between words: an approach based on a bilingual syntactic analysis of French/English parallel corpora. In *COLING 04 Workshop on Multilingual Linguistic Resources*.

Ozdowska S. & Bourigault D. 2004. Détection de relations d'appariement bilingue entre termes à partir d'une analyse syntaxique de corpus. In *Actes des 14ème Congrès RFIA.*

Véronis J. & Langlais P. 2000. Evaluation of parallel text alignment systems. The ARCADE project. In Véronis J. (ed.), *Parallel Text Processing: Alignment and Use of Translation Corpora*, Dordrecht: Kluwer Academic Publishers, pp. 371-388

Wu D. 2000. Bracketing and aligning words and constituents in parallel text using Stochastic Inversion Transduction Grammars. In Véronis, J. (Ed.), *Parallel Text Processing: Alignment and Use of Translation Corpora*, Dordrecht: Kluwer Academic Publishers, pp. 139-167.

Yamada K. & Knight K. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Conference of the Association for Computational Linguistics*.

An Unsupervised System for Identifying English Inclusions in German Text

Beatrice Alex
School of Informatics
University of Edinburgh
Edinburgh, EH8 9LW, UK
v1balex@inf.ed.ac.uk

Abstract

We present an unsupervised system that exploits linguistic knowledge resources, namely English and German lexical databases and the World Wide Web, to identify English inclusions in German text. We describe experiments with this system and the corpus which was developed for this task. We report the classification results of our system and compare them to the performance of a trained machine learner in a series of in- and cross-domain experiments.

1 Introduction

The recognition of foreign words and foreign named entities (NEs) in otherwise mono-lingual text is beyond the capability of many existing approaches and is only starting to be addressed. This language mixing phenomenon is prevalent in German where the number of anglicisms has increased considerably.

We have developed an unsupervised and highly efficient system that identifies English inclusions in German text by means of a computationally inexpensive lookup procedure. By unsupervised we mean that the system does not require any annotated training data and only relies on lexicons and the Web. Our system allows linguists and lexicographers to observe language changes over time, and to investigate the use and frequency of foreign words in a given language and domain. The output also represents valuable information for a number of applications, including polyglot text-to-speech (TTS) synthesis and machine translation (MT).

We will first explain the issue of foreign inclusions in German text in greater detail with examples in Section 2. Sections 3 and 4 describe the data we used and the architecture of our system. In Section 5, we provide an evaluation of the system output and compare the results with those of a series of in- and cross-domain machine learning experiments outlined in Section 6. We conclude and outline future work in Section 7.

2 Motivation

In natural language, new inclusions typically fall into two major categories, foreign words and proper nouns. They cause substantial problems for NLP applications because they are hard to process and infinite in number. It is difficult to predict which foreign words will enter a language, let alone create an exhaustive gazetteer of them. In German, there is frequent exposure to documents containing English expressions in business, science and technology, advertising and other sectors. A look at current headlines confirms the existence of this phenomenon:

(1) “*Security-Tool* verhindert, dass *Hacker* über *Google* Sicherheitslücken finden”[1]
Security tool prevents hackers from finding security holes via Google.

An automatic classifier of foreign inclusions would prove valuable for linguists and lexicographers who

[1] Published in Computerwelt on 10/01/2005: http://www.computerwelt.at

Proceedings of the ACL Student Research Workshop, pages 133–138,
Ann Arbor, Michigan, June 2005.

study this language-mixing phenomenon because lexical resources need to be updated and reflect this trend. As foreign inclusions carry critical content in terms of pronunciation and semantics, their correct recognition will also provide vital knowledge in applications such as polyglot TTS synthesis or MT.

3 Data

Our corpus is made up of a random selection of online German newspaper articles published in the Frankfurter Allgemeine Zeitung between 2001 and 2004 in the domains of (1) *internet & telecomms*, (2) *space travel* and (3) *European Union*. These domains were chosen to examine the different use and frequency of English inclusions in German texts of a more technological, scientific and political nature. With approximately 16,000 tokens per domain, the overall corpus comprises of 48,000 tokens (Table 1).

We created a manually annotated gold standard using an annotation tool based on NITE XML (Carletta et al., 2003). We annotated two classes whereby English words and abbreviations that expand to English terms were classed as "English" (EN) and all other tokens as "Outside" (O).[2] Table 1 presents the number of English inclusions annotated in each gold standard set and illustrates that English inclusions are very sparse in the EU domain (49 tokens) but considerably frequent in the documents in the internet and space travel domains (963 and 485 tokens, respectively). The type-token ratio (TTR) signals that the English inclusions in the space travel data are less diverse than those in the internet data.

Domain		Tokens	Types	TTR
Internet	Total	15919	4152	0.26
	English	963	283	0.29
Space	Total	16066	3938	0.25
	English	485	73	0.15
EU	Total	16028	4048	0.25
	English	49	30	0.61

Table 1: English token and type statistics and type-token-ratios (TTR) in the gold standard

4 System Description

Our system is a UNIX pipeline which converts HTML documents to XML and applies a set of modules to add linguistic markup and to classify nouns as German or English. The pipeline is composed of a pre-processing module for tokenisation and POS-tagging as well as a lexicon lookup and Google lookup module for identifying English inclusions.

4.1 Pre-processing Module

In the pre-processing module, the downloaded Web documents are firstly cleaned up using `Tidy`[3] to remove HTML markup and any non-textual information and then converted into XML. Subsequently, two rule-based grammars which we developed specifically for German are used to tokenise the XML documents. The grammar rules are applied with `lxtransduce`[4], a transducer which adds or rewrites XML markup on the basis of the rules provided. `Lxtransduce` is an updated version of `fsgmatch`, the core program of LT TTT (Grover et al., 2000). The tokenised text is then POS-tagged using TnT trained on the German newspaper corpus Negra (Brants, 2000).

4.2 Lexicon Lookup Module

For the initial lookup, we used CELEX, a lexical database of English, German and Dutch containing full and inflected word forms as well as corresponding lemmas. CELEX lookup was only performed for tokens which TnT tagged as nouns (NN), foreign material (FM) or named entities (NE) since anglicisms representing other parts of speech are relatively infrequent in German (Yeandle, 2001). Tokens were looked up twice, in the German and the English database and parts of hyphenated compounds were checked individually. To identify capitalised English tokens, the lookup in the English database was made case-insensitive. We also made the lexicon lookup sensitive to POS tags to reduce classification errors. Tokens were found either only in the German lexicon (1), only in the English lexicon (2) in both (3) or in neither lexicon (4).

(1) The majority of tokens found exclusively in

[2] We did not annotate English inclusions if part of URLs (*www.stepstone.de*), mixed-lingual unhyphenated compounds (*Shuttleflug*) or with German inflections (*Receivern*) as further morphological analysis is required to recognise them. Our aim is to address these issues in future work.

[3] `http://tidy.sourceforge.net`

[4] `http://www.ltg.ed.ac.uk/~richard/lxtransduce.html`

the German lexicon are actual German words. The remaining are English words with German case inflection such as *Computern*. The word *Computer* is used so frequently in German that it already appears in lexicons and dictionaries. To detect the base language of the latter, a second lookup can be performed checking whether the lemma of the token also occurs in the English lexicon.

(2) Tokens found exclusively in the English lexicon such as *Software* or *News* are generally English words and do not overlap with German lexicon entries. These tokens are clear instances of foreign inclusions and consequently tagged as English.

(3) Tokens which are found in both lexicons are words with the same orthographic characteristics in both languages. These are words without inflectional endings or words ending in *s* signalling either the German genitive singular or the German and English plural forms of that token, e.g. *Computers*. The majority of these lexical items have the same or similar semantics in both languages and represent assimilated loans and cognates where the language origin is not always immediately apparent. Only a small subgroup of them are clearly English loan words (e.g. *Monster*). Some tokens found in both lexicons are interlingual homographs with different semantics in the two languages, e.g. *Rat* (*council* vs. *rat*). Deeper semantic analysis is required to classify the language of such homographs which we tagged as German by default.

(4) All tokens found in neither lexicon are submitted to the Google lookup module.

4.3 Google Lookup Module

The Google lookup module exploits the World Wide Web, a continuously expanding resource with documents in a multiplicity of languages. Although the bulk of information available on the Web is in English, the number of texts written in languages other than English has increased rapidly in recent years (Crystal, 2001; Grefenstette and Nioche, 2000).

The exploitation of the Web as a linguistic corpus is developing into a growing trend in computational linguistics. The sheer size of the Web and the continuous addition of new material in different languages make it a valuable pool of information in terms of language in use. The Web has already been used successfully for a series of NLP tasks such as MT (Grefenstette, 1999), word sense disambiguation (Agirre and Martinez, 2000), synonym recognition (Turney, 2001), anaphora resolution (Modjeska et al., 2003) and determining frequencies for unseen bi-grams (Keller and Lapata, 2003).

The Google lookup module obtains the number of hits for two searches per token, one on German Web pages and one on English ones, an advanced language preference offered by Google. Each token is classified as either German or English based on the search that returns the higher normalised score of the number of hits. This score is determined by weighting the number of raw hits by the size of the Web corpus for that language. We determine the latter following a method proposed by Grefenstette and Niochi (2000) by using the frequencies of a series of representative tokens within a standard corpus in a language to determine the size of the Web corpus for that language. We assume that a German word is more frequently used in German text than in English and vice versa. As illustrated in Table 2, the German word *Anbieter* (provider) has a considerably higher weighted frequency in German Web documents (DE). Conversely, the English word *provider* occurs more often in English Web documents (EN). If both searches return zero hits, the token is classified as German by default. Word queries that return zero or a low number of hits can also be indicative of new expressions that have entered a language.

Google lookup was only performed for the tokens found in neither lexicon in order to keep computational cost to a minimum. Moreover, a preliminary experiment showed that the lexicon lookup is already sufficiently accurate for tokens contained exclusively in the German or English databases. Current Google search options are also limited in that queries cannot be treated case- or POS-sensitively. Consequently, interlingual homographs would often mistakenly be classified as English.

Language	DE		EN	
Hits	Raw	Normalised	Raw	Normalised
Anbieter	3.05	0.002398	0.04	0.000014
Provider	0.98	0.000760	6.42	0.002284

Table 2: Raw counts (in million) and normalised counts of two Google lookup examples

5 Evaluation of the Lookup System

We evaluated the system's performance for all tokens against the gold standard. While the accuracies in Table 3 represent the percentage of all correctly tagged tokens, the F-scores refer to the English tokens and are calculated giving equal weight to precision (P) and recall (R) as $F = (2 * P * R)/(P + R)$.

The system yields relatively high F-scores of 72.4 and 73.1 for the internet and space travel data but only a low F-score of 38.6 for the EU data. The latter is due to the sparseness of English inclusions in that domain (Table 1). Although recall for this data is comparable to that of the other two domains, the number of false positives is high, causing low precision and F-score. As the system does not look up one-character tokens, we implemented further post-processing to classify individual characters as English if followed by a hyphen and an English inclusion. This improves the F-score by 4.8 for the internet data to 77.2 and by 0.6 for the space travel data to 73.7 as both data sets contain words like *E-Mail* or *E-Business*. Post-processing does not decrease the EU score. This indicates that domain-specific post-processing can improve performance.

Baseline accuracies when assuming that all tokens are German are also listed in Table 3. As F-scores are calculated based on the English tokens in the gold standard, we cannot report comparable baseline F-scores. Unsurprisingly, the baseline accuracies are relatively high as most tokens in a German text are German and the amount of foreign material is relatively small. The added classification of English inclusions yielded highly statistical significant improvements ($p<0.001$) over the baseline of 3.5% for the internet data and 1.5% for the space travel data. When classifying English inclusions in the EU data, accuracy decreased slightly by 0.3%.

Table 3 also shows the performance of `TextCat`, an n-gram-based text categorisation algorithm of Cavnar and Trenkle (1994). While this language idenfication tool requires no lexicons, its F-scores are low for all 3 domains and very poor for the EU data. This confirms that the identification of English inclusions is more difficult for this domain, coinciding with the result of the lookup system. The low scores also prove that such language identification is unsuitable for token-based language classification.

Domain	Method	Accuracy	F-score
Internet	Baseline	94.0%	-
	Lookup	97.1%	72.4
	Lookup + post	97.5%	77.2
	TextCat	92.2%	31.0
Space	Baseline	97.0%	-
	Lookup	98.5%	73.1
	Lookup + post	98.5%	73.7
	TextCat	93.8%	26.7
EU	Baseline	99.7%	-
	Lookup	99.4%	38.6
	Lookup + post	99.4%	38.6
	TextCat	96.4%	4.7

Table 3: Lookup results (with and without post-processing) compared to TextCat and baseline

6 Machine Learning Experiments

The recognition of foreign inclusions bears great similarity to classification tasks such as named entity recognition (NER), for which various machine learning techniques have proved successful. We were therefore interested in determining the performance of a trained classifier for our task. We experimented with a conditional Markov model tagger that performed well on language-independent NER (Klein et al., 2003) and the identification of gene and protein names (Finkel et al., 2005).

6.1 In-domain Experiments

We performed several 10-fold cross-validation experiments with different feature sets. They are referred to as in-domain (ID) experiments as the tagger is trained and tested on data from the same domain (Table 4). In the first experiment (ID1), we use the tagger's standard feature set including words, character sub-strings, word shapes, POS-tags, abbreviations and NE tags (Finkel et al., 2005). The resulting F-scores are high for the internet and space travel data (84.3 and 91.4) but are extremely low for the EU data (13.3) due to the sparseness of English inclusions in that data set. ID2 involves the same setup as ID1 but eliminating all features relying on the POS-tags. The tagger performs similarly well for the internet and space travel data but improves by 8 points to an F-score of 21.3 for the EU data. This can be attributed to the fact that the POS-tagger does not perform with perfect accuracy particularly on data containing foreign inclusions. Providing the tagger with this information is therefore not necessarily useful for this task, especially when the data is sparse. Nevertheless, there is a big discrepancy between the F-score for the EU data and those of the other two data sets. ID3 and ID4 are set up as ID1 and ID2 but incorporating the output of the lookup system as a gazetteer feature. The tagger benefits considerably from this lookup feature and yields better F-scores for all three domains in ID3 (internet: 90.6, space travel: 93.7, EU: 44.4).

Table 4 also compares the best F-scores produced with the tagger's own feature set (ID2) to the best results of the lookup system and the baseline. While the tagger performs much better for the internet and the space travel data, it requires hand-annotated training data. The lookup system, on the other hand, is essentially unsupervised and therefore much more portable to new domains. Given the necessary lexicons, it can easily be run over new text and text in a different language or domain without further cost.

Domain		Accuracy	F-score
Internet	ID1	98.4%	84.3
	ID2	98.3%	84.3
	ID3	98.9%	90.6
	ID4	98.9%	90.8
	Best Lookup	97.5%	77.2
	Baseline	94.0%	-
Space	ID1	99.5%	91.4
	ID2	99.5%	91.3
	ID3	99.6%	93.7
	ID4	99.6%	92.8
	Best Lookup	98.5%	73.7
	Baseline	97.0%	-
EU	ID1	99.7%	13.3
	ID2	99.7%	21.3
	ID3	99.8%	44.4
	ID4	99.8%	44.4
	Best Lookup	99.4%	38.6
	Baseline	99.7%	-

Table 4: Accuracies and F-scores for ID experiments

6.2 Cross-domain Experiments

The tagger achieved surprisingly high F-scores for the internet and space travel data, considering the small training data set of around 700 sentences used for each ID experiment described above. Although both domains contain a large number of English inclusions, their type-token ratio amounts to 0.29 in the internet data and 0.15 in the space travel data (Table 1), signalling that English inclusions are frequently repeated in both domains. As a result, the likelihood of the tagger encountering an unknown inclusion in the test data is relatively small.

	Accuracy	F-score	UTT
CD1	97.9%	54.2	81.9%
Best Lookup	98.5%	73.7	-
Baseline	97.0%	-	-
CD2	94.6%	22.2	93.9%
Best Lookup	97.5%	77.2	-
Baseline	94.0%	-	-

Table 5: Accuracies, F-scores and percentages of unknown target types (UTT) for cross-domain experiments compared to best lookup and baseline

To examine the tagger's performance on a new domain containing more unknown inclusions, we ran two cross-domain (CD) experiments: CD1, training on the internet and testing on the space travel data, and CD2, training on the space travel and testing on the internet data. We chose these two domain pairs to ensure that both the training and test data contain a relatively large number of English inclusions. Table 5 shows that the F-scores for both CD experiments are much lower than those obtained when training and testing the tagger on documents from the same domain. In experiment CD1, the F-score only amounts to 54.2 while the percentage of unknown target types in the space travel test data is 81.9%. The F-score is even lower in the second experiment at 22.2 which can be attributed to the fact that the percentage of unknown target types in the internet test data is higher still at 93.9%.

These results indicate that the tagger's high performance in the ID experiments is largely due to the fact that the English inclusions in the test data are known, i.e. the tagger learns a lexicon. It is therefore more complex to train a machine learning classifier to perform well on new data with more and more new anglicisms entering German over time. The amount of unknown tokens will increase constantly unless new annotated training data is added.

7 Conclusions and Future Work

We have presented an unsupervised system that exploits linguistic knowledge resources including lexicons and the Web to classify English inclusions in German text on different domains. Our system can be applied to new texts and domains with little computational cost and extended to new languages as long as lexical resources are available. Its main advantage is that no annotated training data is required.

The evaluation showed that our system performs well on non-sparse data sets. While being outperformed by a machine learner which requires a trained model and therefore manually annotated data, the output of our system increases the performance of the learner when incorporating this information as an additional feature. Combining statistical approaches with methods that use linguistic knowledge resources can therefore be advantageous.

The low results obtained in the CD experiments indicate however that the machine learner merely learns a lexicon of the English inclusions encountered in the training data and is unable to classify many unknown inclusions in the test data. The Google lookup module implemented in our system represents a first attempt to overcome this problem as the information on the Web never remains static and at least to some extent reflects language in use.

The current system tracks full English word forms. In future work, we aim to extend it to identify English inclusions within mixed-lingual tokens. These are words containing morphemes from different languages, e.g. English words with German inflection (*Receivern*) or mixed-lingual compounds (*Shuttleflug*). We will also test the hypothesis that automatic classification of English inclusions can improve text-to-speech synthesis quality.

Acknowledgements

Thanks go to Claire Grover and Frank Keller for their input. This research is supported by grants from the University of Edinburgh, Scottish Enterprise Edinburgh-Stanford Link (R36759) and ESRC.

References

Eneko Agirre and David Martinez. 2000. Exploring automatic word sense disambiguation with decision lists and the Web. In *Proceedings of the Semantic Annotation and Intelligent Annotation workshop, COLING*.

Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference*.

Jean Carletta, Stefan Evert, Ulrich Heid, Jonathan Kilgour, Judy Robertson, and Holgar Voormann. 2003. The NITE XML toolkit: flexible annotation for multimodal language data. *Behavior Research Methods, Instruments, and Computers*, 35(3):353–363.

William B. Cavnar and John M. Trenkle. 1994. N-gram-based text categorization. In *Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval*.

David Crystal. 2001. *Language and the Internet*. Cambridge University Press.

Jenny Finkel, Shipra Dingare, Christopher Manning, Malvina Nissim, Beatrice Alex, and Claire Grover. 2005. Exploring the boundaries: Gene and protein identification in biomedical text. *BMC Bioinformatics*. In press.

Gregory Grefenstette and Julien Nioche. 2000. Estimation of English and non-English language use on the WWW. In *Proceedings of RIAO 2000*.

Gregory Grefenstette. 1999. The WWW as a resource for example-based machine translation tasks. In *Proceedings of ASLIB'99 Translating and the Computer*.

Claire Grover, Colin Matheson, Andrei Mikheev, and Moens Marc. 2000. LT TTT - a flexible tokenisation tool. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*.

Frank Keller and Mirella Lapata. 2003. Using the Web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3):458–484.

Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D. Manning. 2003. Named entity recognition with character-level models. In *Proceedings of the 7th Conference on Natural Language Learning*.

Natalia Modjeska, Katja Markert, and Malvina Nissim. 2003. Using the Web in machine learning for other-anaphora resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Peter D. Turney. 2001. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning*.

David Yeandle. 2001. Types of borrowing of Anglo-American computing terminology in German. In Marie C. Davies, John L. Flood, and David N. Yeandle, editors, *Proper Words in Proper Places: Studies in Lexicology and Lexicography in Honour of William Jervis Jones*, pages 334–360. Stuttgarter Arbeiten zur Germanistik 400, Stuttgart, Germany.

Corpus-Oriented Development of Japanese HPSG Parsers

Kazuhiro Yoshida
Department of Computer Science,
University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0033
kyoshida@is.s.u-tokyo.ac.jp

Abstract

This paper reports the corpus-oriented development of a wide-coverage Japanese HPSG parser. We first created an HPSG treebank from the EDR corpus by using heuristic conversion rules, and then extracted lexical entries from the treebank. The grammar developed using this method attained wide coverage that could hardly be obtained by conventional manual development. We also trained a statistical parser for the grammar on the treebank, and evaluated the parser in terms of the accuracy of semantic-role identification and dependency analysis.

1 Introduction

In this study, we report the corpus-oriented development of a Japanese HPSG parser using the EDR Japanese corpus (2002). Although several researchers have attempted to utilize linguistic grammar theories, such as LFG (Bresnan and Kaplan, 1982), CCG (Steedman, 2001) and HPSG (Pollard and Sag, 1994), for parsing real-world texts, such attempts could hardly be successful, because manual development of wide-coverage linguistically motivated grammars involves years of labor-intensive effort.

Corpus-oriented grammar development is a grammar development method that has been proposed as a promising substitute for conventional manual development. In corpus-oriented methods, a treebank of a target grammar is constructed first, and various grammatical constraints are extracted from the treebank. Previous studies reported that wide-coverage grammars can be obtained at low cost by using this method. (Hockenmaier and Steedman, 2002; Miyao et al., 2004) The treebank can also be used for training statistical disambiguation models, and hence we can construct a statistical parser for the extracted grammar.

The corpus-oriented method enabled us to develop a Japanese HPSG parser with semantic information, whose coverage on real-world sentences is 95.3%. This high coverage allowed us to evaluate the parser in terms of the accuracy of dependency analysis on real-world texts, the evaluation measure that is previously used for more statistically-oriented parsers.

2 HPSG

Head-Driven Phrase Structure Grammar (HPSG) is classified into lexicalized grammars (Schabes et al., 1988). It attempts to model linguistic phenomena by interactions between a small number of grammar rules and a large number of lexical entries. Figure 1 shows an example of an HPSG derivation of a Japanese sentence 'kare ga shinda,' which means, 'He died.' In HPSG, linguistic entities such as words and phrases are represented by typed feature structures called *signs*, and the grammaticality of a sentence is verified by applying grammar rules to a sequence of signs. The sign of a lexical entry encodes the type and valence (i.e. restriction on the types of phrases that can appear around the word) of a corresponding word. Grammar rules of HPSG consist of

Proceedings of the ACL Student Research Workshop, pages 139–144,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

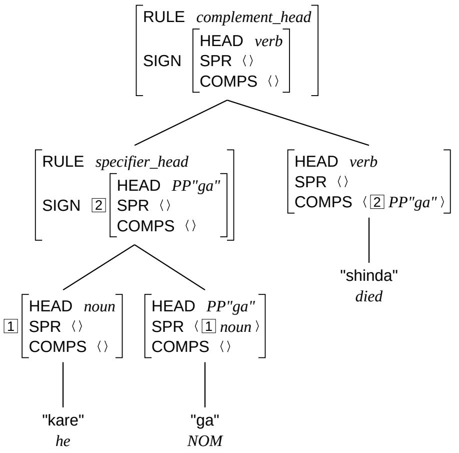


Figure 1: Example of HPSG analysis.

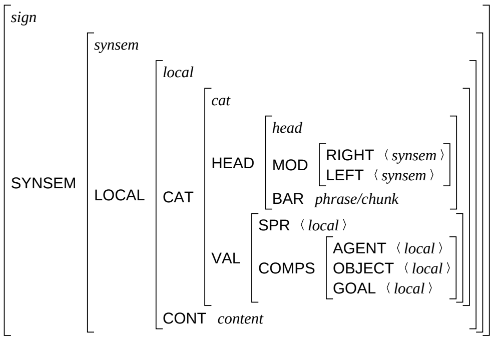


Figure 2: Sign of the grammar.

schemata and *principles*, the former enumerate possible patterns of phrase structures, and the latter are basically for controlling the inheritance of daughters' features to the parent.

In the current example, the lexical entry for "shinda" is of the type *verb*, as indicated in its *HEAD*, and its *COMPS* feature restricts its preceding phrase to be of the type *PP"ga"*. The *HEAD* feature of the root node of the derivation is inherited from the lexical entry for "shinda", because *complement-head* structures are head-final, and the *head feature* principle states that the *HEAD* feature of a phrase must be inherited from its head daughter.

There are several implementations of Japanese HPSG grammars. JACY (Siegel and Bender, 2002) is a hand-crafted Japanese HPSG grammar that provides semantic information as well as linguistically motivated analysis of complex constructions. However, the evaluation of the grammar has not been done on domain-independent real-world texts such as newspaper articles. Although Bond et al. (2004) attempted to improve the coverage of the JACY grammar through the development of an HPSG treebank, they limited the target of their treebank annotation to short sentences from dictionary definitions. SLUNG (Mitsuishi et al., 1998) is an HPSG grammar whose coverage on real-world sentences is about 99%, but the grammar is *underspecified*, which means that the constraints of the grammar are not sufficient for conducting semantic analysis. By employing corpus-oriented development, we aim to develop a wide-coverage HPSG parser that enables semantic analysis of real-word texts.

3 Grammar Design

First, we provide a brief description of some characteristics of Japanese. Japanese is head final, and phrases are typically headed by function words. Arguments of verbs usually have no fixed order (this phenomenon is called *scrambling*) and are freely omitted. Arguments' semantic relations to verbs are chiefly determined by their head postpositions. For example, 'boku/*I* ga/*NOM* kare/*he* wo/*ACC* koroshi/*kill* ta/*DECL*' (I killed him) can be paraphrased as 'kare wo boku ga koroshi ta,' without changing the meaning.

The *case alternation* phenomenon must also be taken into account. Case alternation is caused by special auxiliaries "(sa)se" and "(ra)re," which are causative and passive auxiliaries, respectively, and the verbs change their subcategorization behavior when they are combined with these auxiliaries.

The following sections describe the design of our grammar. Especially, treatment of the scrambling and case alternation phenomena is provided in detail.

3.1 Fundamental Phrase Structures

Figure 2 presents the basic structure of signs of our grammar. The *HEAD* feature specifies phrasal categories, the *MOD* feature represents restrictions on the left and right modifiees, and the *VAL* feature encodes valence information. (For the explanation of the *BAR* feature, see the description of the *promo-*

Table 1: Schemata and their uses.

schema name	common use of the rule
specifier-head	PP or NP + postposition VP + verbal ending NP + suffix
complement-head	argument (PP/NP) + verb
compound-noun	NP + NP
modifier-head	modifier + head
head-modifier	phrase + punctuation
promotion	promotes chunks to phrases

tion schema below.) [1] For some types of phrases, additional features are specified as *HEAD* features.

Now, we provide a detailed explanation of the design of the schemata and how the features in Figure 2 work. The following descriptions are also summarized in Table 1.

specifier-head schema Words are first concatenated by this schema to construct basic word chunks. Postpositional phrases (PPs), which consist of postpositions and preceding phrases, are the most typical example of *specifier-head* structures. For postpositions, we specify a head feature *PFORM*, with the postposition's surface string as its value, in addition to the features in Figure 2, because differences of postpositions play a crucial role in disambiguating semantic-structures of Japanese. For example, the postposition 'wo' has a *PFORM* feature whose value is "wo," and it accepts an NP as its specifier. As a result, a PP such as "kare wo" inherits the value of *PFORM* feature "wo" from 'wo.'

The schema is also used when VPs are constructed from verbs and their endings (or, sometimes auxiliaries. See also Section 3.2).

complement-head schema This schema is used for combining VPs with their subcategorized arguments (see Section 3.2 for details).

compound-noun schema Because nouns can be freely concatenated to form compound nouns, a special schema is used for compound nouns.

modifier-head schema This schema is for modifiers and their heads. Binary structures that cannot be captured by the above three schemata are also considered to be modifier-head structures.[2]

head-modifier schema This schema is used when the *modifier-head* schema is not appropriate. In the current implementation, it is used for a phrase and its following punctuation.

promotion schema This unary schema changes the value of the *BAR* feature from *chunk* to *phrase*. The distinction between these two types of constituents is for prohibiting some kind of spurious ambiguities. For example, 'kinou/*yesterday* koroshi/*kill* ta/*DECL*' can be analyzed in two different ways, i.e. '(kinou (koroshi ta))' and '((kinou koroshi) ta).' The latter analysis is prevented by restricting "kinou"'s modifiee to be a *phrase*, and "ta"'s specifier to be a *chunk*, and by assuming "koroshi" to be a *chunk*.

3.2 Scrambling and Case Alternation

Scrambling causes problems in designing a Japanese HPSG grammar, because original HPSG, designed for English, specifies the subcategorization frame of a verb as an ordered list, and the semantic roles of arguments are determined by their order in the complement list.

Our implementation treats the complement feature as a list of semantic roles. Semantic roles for which verbs subcategorize are *agent*, *object*, and *goal*.[3] Correspondingly, we assume three subtypes of the *complement-head* schema: the *agent-head*, *object-head*, and *goal-head* schemata. When verbs take their arguments, arguments receive semantic roles which are permitted by the subcategorization of verbal signs. We do not restrict the order of application of the three types of *complement-head* schemata, so that a single verbal lexical entry can accept arguments that are scrambled in arbitrary order. In Figure 3, "kare ga" is a ga-marked PP, so it is analyzed as an agent of "koro(su)." [4]

Case alternation is caused by special auxiliaries "(sa)se" and "(ra)re." For instance, in 'boku/*I*

[1] The *CONTENT* feature, which should contain information about the semantic contents of syntactic entities, is ignored in the current implementation of the grammar.

[2] Current implementation of the grammar treats complex structures such as relative clause constructions and coordinations just the same as simple modification.

[3] These are the three roles most commonly found in EDR.

[4] We assume that a single semantic role cannot be occupied by more than one syntactic entities. This assumption is sometimes violated in EDR's annotation, causing failures in grammar extraction.

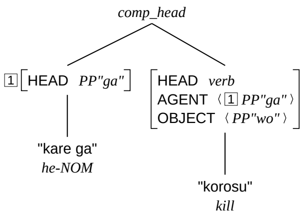


Figure 3: Verb and its argument.

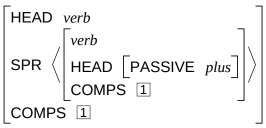


Figure 4: Lexical sign of "(ra)re".

ga/*NOM* kare/*he* ni/*DAT* korosa/*kill* re/*PASSIVE* ta/*DECL*' (I was killed by him), "korosa" takes a "ga"-marked PP as an object and a "ni"-marked PP as an agent, though without "(sa)re," it takes a "ga"-marked PP as an agent and a "wo"-marked PP as an object.

We consider auxiliaries as a special type of verbs which do not have their own subcategorization frames. They inherit the subcategorization frames of verbs.[5] To capture the case alternation phenomenon, each verb has distinct lexical entries for its passive and causative uses. This distinction is made by binary valued *HEAD* features, *PASSIVE* and *CAUSATIVE*. The passive (causative) auxiliary restricts the value of its specifier's *PASSIVE* (*CAUSATIVE*) feature to be *plus*, so that it can only be combined with properly case-alternated verbal lexical entries.

Figure 4 presents the lexical sign of the passive auxiliary "(ra)re." Our analysis of an example sentence is presented in Figure 5. Note that the passive auxiliary "re(ta)" requires the value of the *PASSIVE* feature of its specifier be *plus*, and hence "koro(sa)" cannot take the same lexical entry as in Figure 3.

4 Grammar Extraction from EDR

The EDR Japanese corpus consists of 207802 sentences, mainly from newspapers and magazines. The annotation of the corpus includes word segmentation, part-of-speech (POS) tags, phrase structure annotation, and semantic information.

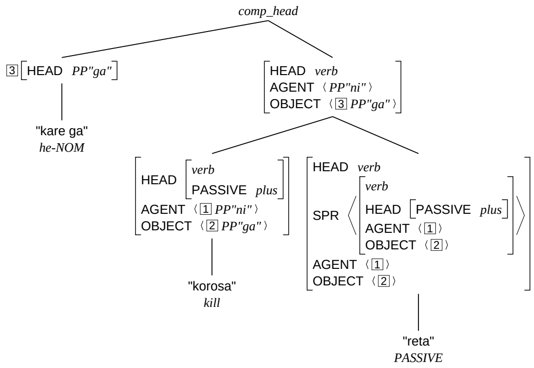


Figure 5: Example of passive construction.

The heuristic conversion of the EDR corpus into an HPSG treebank consists of the following steps. A sentence '((kare/NP-*he* wo/PP-*ACC*) (koro/VP-*kill* shi/VP-*ENDING* ta/VP-*DECL*))' ([I] killed him yesterday) is used to provide examples in some steps.

Phrase type annotation Phrase type labels such as NP and VP are assigned to non-terminal nodes. Because Japanese is head final, the label of the right-most daughter of a phrase is usually percolated to its parent. After this step, the example sentence will be '((PP kare/NP wo/PP) (VP koro/VP shi/VP ta/VP)).'

Assign head features The types of head features of terminal nodes are determined, chiefly from their phrase types. Features specific to some categories, such as *PFORM*, are also assigned in this step.

Binarization Phrases for which EDR employs flat annotation are converted into binary structures. The binarized phrase structure of the example sentence will be '((kare wo) ((koro shi) ta)).'

Assign schema names Schema names are assigned according to the patterns of phrase structures. For instance, a phrase structure which consists of PP and VP is identified as a *complement-head* structure, if the VP's argument and the PP are coindexed. In the example sentence, 'kare wo' is annotated as 'koro"s object in EDR, so the *object-head* schema is applied to the root node of the derivation.

Inverse schema application The consistency of the derivation of the obtained HPSG treebank is ver-

[5] The control phenomena caused by auxiliaries are currently unsupported in our grammar.

ified by applying the schemata to each node of the derivation trees in the treebank.

Lexicon Extraction Lexical entries are extracted from the terminal nodes of the obtained treebank.

5 Disambiguation Model

We also train disambiguation models for the grammar using the obtained treebank. We employ log-linear models (Berger et al., 1996) for the disambiguation. The probability of a parse P of a sentence S is calculated as follows:

$$p(P|S) = \frac{\exp(\sum_i f_i(P)\lambda_i)}{\sum_{P'} \exp(\sum_i f_i(P')\lambda_i)}$$

where f_i are feature functions, λ_i are strengths of the feature functions, and P' spans all possible parses of S. We employ Gaussian MAP estimation (Chen and Rosenfeld, 1999) as a criterion for optimizing λ_i. An algorithm proposed by Miyao et. al. (2002) provides an efficient solution to this optimization problem.

6 Experiments

Because the aim of our research is to construct a Japanese parser that can extract semantic information from real-world texts, we evaluated our parser in terms of its coverage and semantic-role identification accuracy. We also compare the accuracy of our parser with that of an existing statistical dependency analyzer, in order to investigate the necessity of further improvements to our disambiguation model.

The following experiments were conducted using the EDR Japanese corpus. An HPSG grammar was extracted from 51951[6] sentences of the corpus, and the same set of sentences were used as a training set for the disambiguation model. 47767 sentences (91.9%) of the training set were successfully converted into an HPSG treebank, from which we extracted lexical entries.

When we construct a lexicon from the extracted lexical entries, we reserved lexical entry templates for infrequent words as default templates for unknown words of each POS, in order to achieve sufficient coverage. The threshold for 'infrequent' words were determined to be 30 from the results of preliminary experiments.

We used 2079 EDR sentences as a test set. (Another set of 2078 sentences were used as a development set.) The test set is also converted into an HPSG treebank, and the conversion was successful for 1913 sentences. (We will call the obtained HPSG treebank the "test treebank.")

As features of the log-linear model, we extracted the POS of the head, template name of the head, surface string and its ending of the head, punctuation contained in the phrase, and distance between heads of daughters, from each sign in derivation trees. These features are used in combinations.

The coverage of the parser[7] on the test set was 95.3% (1982/2079). Though it is still below the coverage achieved by SLUNG (Mitsuishi et al., 1998), our grammar has richer information that enables semantic analysis, which is lacking in SLUNG.

We evaluated the parser in terms of its accuracy in identifying semantic roles of arguments of verbs. For each phrase which is in *complement-head* relation with some VP, a semantic role is assigned according to the type[8] of the *complement-head* structure. The performance of our parser on the test treebank was 63.8%/57.8% in precision/recall of semantic roles.

As most studies on syntactic parsing of Japanese have focused on *bunsetsu*-based dependency analysis, we also attempted an evaluation in this framework.[9] In order to evaluate our parser by *bunsetsu* dependency, we converted the phrase structures of EDR and the output of our parser into dependency structures of the right-most content word of each *bunsetsu*. *Bunsetsu* boundaries of the EDR sentences were determined by using simple heuristic rules. The dependency accuracies and the sentential accuracies of our parser and Kanayama et. al.'s analyzer are shown in Table 2. (*failure* sentences are not counted for calculating accuracies.) Our results were still significantly lower than those of

[6] We could not use the entire corpus for the experiments, because of the limitation of computational resources.

[7] Coverage of the parser can be somewhat lower than that of the grammar, because we employed a beam thresholding technique proposed by Tsuruoka et al. (Tsuruoka et al., 2004).

[8] As described in Section 3.2, there are three types of *complement-head* structures.

[9] *Bunsetsu* is a widely accepted syntactic unit of Japanese, which usually consists of a content word followed by a function word.

	accuracy (dependency)		accuracy (sentence)		# failure
(Kanayama et al., 2000)	88.6%	(23078/26062)	46.9%	(1560/3326)	1.4% (46/3372)
This paper	85.0%	(13201/15524)	37.4%	(705/1887)	1.4% (26/1913)

Table 2: Accuracy of dependency analysis.

Kanayama et. al., which are the best reported dependency accuracies on EDR.

This experiment revealed that the accuracy of our parser requires further improvement, although our grammar achieved high coverage. Our expectation is that incorporating grammar rules for complex structures which is ignored in the current implementation (e.g. control, relative clause, and coordination constructions) will improve the accuracy of the parser. In addition, we should investigate whether the semantic analysis our parser provides can contribute the performance of more application-oriented tasks such as information extraction.

7 Conclusion

We developed a Japanese HPSG grammar by means of the corpus-oriented method, and the grammar achieved the high coverage, which we consider to be nearly sufficient for real-world applications. However, the accuracy of the parser in terms of dependency analysis was significantly lower than that of the existing parser. We expect that the accuracy can be improved through further elaboration of the grammar design and disambiguation method.

References

Adam L. Berger, Stephen Della Pietra, and Vincent J. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1).

Francis Bond, Sanae Fujita, Chikara Hashimoto, Kaname Kasahara, Shigeko Nariyama, Eric Nichols, Akira Ohtani, Takaaki Tanaka, and Shigeaki Amano. 2004. The Hinoki Treebank: A Treebank for Text Understanding. In *Proc. of IJCNLP-04*.

J. Bresnan and R. M. Kaplan. 1982. Introduction: Grammars as mental representations of language. In *The Mental Representation of Grammatical Relations*. MIT Press.

S. Chen and R. Rosenfeld. 1999. A Gaussian prior for smoothing maximum entropy models. In *Technical Report CMUCS*.

Julia Hockenmaier and Mark Steedman. 2002. Acquiring Compact Lexicalized Grammars from a Cleaner Treebank. In *Proc. of Third LREC*.

Hiroshi Kanayama, Kentaro Torisawa, Mitsuishi Yutaka, and Jun'ichi Tsujii. 2000. A Hybrid Japanese Parser with Hand-crafted Grammar and Statistics. In *Proc. of the 18th COLING*, volume 1.

Yutaka Mitsuishi, Kentaro Torisawa, and Jun'ichi Tsujii. 1998. HPSG-Style Underspecified Japanese Grammar with Wide Coverage. In *Proc. of the 17th COLING–ACL*.

Yusuke Miyao and Jun'ichi Tsujii. 2002. Maximum Entropy Estimation for Feature Forests. In *Proc. of HLT 2002*.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2004. Corpus-oriented Grammar Development for Acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank. In *Proc. of IJCNLP-04*.

National Institute of Information and Communications Technology. 2002. EDR Electronic Dictionary Version 2.0 Technical Guide.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. The University of Chicago Press.

Y. Schabes, A. Abeille, and A. K. Joshi. 1988. Parsing Strategies with 'Lexicalized' Grammars: Application to Tree Adjoining Grammars. In *Proc. of the 12th COLING*.

Melanie Siegel and Emily M. Bender. 2002. Efficient Deep Processing of Japanese. In *Proc. of the 3rd Workshop on Asian Language Resources and International Standardization. COLING 2002 Post-Conference Workshop, August 31*.

Mark Steedman. 2001. *The Syntactic Process*. MIT Press.

Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Tsujii. 2004. Towards efficient probabilistic HPSG parsing: integrating semantic and syntactic preference to guide the parsing. In *Proc. of IJCNLP-04 Workshop: Beyond shallow analyses - Formalisms and statistical modeling for deep analyses*.

Unsupervised Discrimination and Labeling of Ambiguous Names

Anagha K. Kulkarni
Department of Computer Science
University Of Minnesota
Duluth, MN 55812
kulka020@d.umn.edu
http://senseclusters.sourceforge.net

Abstract

This paper describes adaptations of unsupervised word sense discrimination techniques to the problem of name discrimination. These methods cluster the contexts containing an ambiguous name, such that each cluster refers to a unique underlying person or place. We also present new techniques to assign meaningful labels to the discovered clusters.

1 Introduction

A name assigned to an entity is often thought to be a unique identifier. However this is not always true. We frequently come across multiple people sharing the same name, or cities and towns that have identical names. For example, the top ten results for a Google search of *John Gilbert* return six different individuals: A famous actor from the silent film era, a British painter, a professor of Computer Science, etc. Name ambiguity is relatively common, and makes searching for people, places, or organizations potentially very confusing.

However, in many cases a human can distinguish between the underlying entities associated with an ambiguous name with the help of surrounding context. For example, a human can easily recognize that a document that mentions *Silent Era*, *Silver Screen*, and *The Big Parade* refers to John Gilbert the actor, and not the professor. Thus the neighborhood of the ambiguous name reveals distinguishing features about the underlying entity.

Our approach is based on unsupervised learning from raw text, adapting methods originally proposed by (Purandare and Pedersen, 2004). We do not utilize any manually created examples, knowledge bases, dictionaries, or ontologies in formulating our solution. Our goal is to discriminate among multiple contexts that mention a particular name strictly on the basis of the surrounding contents, and assign meaningful labels to the resulting clusters that identify the underlying entity.

This paper is organized as follows. First, we review related work in name discrimination and cluster labeling. Next we describe our methodology step-by-step and then review our experimental data and results. We conclude with a discussion of our results and outline our plans for future work.

2 Related Work

A number of previous approaches to name discrimination have employed ideas related to context vectors. (Bagga and Baldwin, 1998) proposed a method using the vector space model to disambiguate references to a person, place, or event across multiple documents. Their approach starts by using the CAMP system to find related references within a single document. For example, it might determine that *he* and *the President* refers to *Bill Clinton*. CAMP creates co-reference chains for each entity in a single document, which are then extracted and represented in the vector space model. This model is used to find the similarity among referents, and thereby identify the same referent that occurs in multiple documents.

(Mann and Yarowsky, 2003) take an approach to

Proceedings of the ACL Student Research Workshop, pages 145–150,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

name discrimination that incorporates information from the World Wide Web. They propose to use various contextual characteristics that are typically found near and within an ambiguous proper-noun for the purpose of disambiguation. They utilize categorical features (e.g., age, date of birth), familial relationships (e.g., wife, son, daughter) and associations that the entity frequently shows (e.g. country, company, organization). Such biographical information about the entities to be disambiguated is mined from the Web using a bootstrapping method. The Web pages containing the ambiguous name are assigned a vector depending upon the extracted features and then these vectors are grouped using agglomerative clustering.

(Pantel and Ravichandran, 2004) have proposed an algorithm for labeling semantic classes, which can be viewed as a form of cluster. For example, a semantic class may be formed by the words: *grapes, mango, pineapple, orange* and *peach*. Ideally this cluster would be labeled as the semantic class of *fruit*. Each word of the semantic class is represented by a feature vector. Each feature consists of syntactic patterns (like verb-object) in which the word occurs. The similarity between a few features from each cluster is found using point-wise mutual information (PMI) and their average is used to group and rank the clusters to form a grammatical template or signature for the class. Then syntactic relationships such as *Noun like Noun* or *Noun such as Noun* are searched for in the templates to give the cluster an appropriate name label. The output is in the form of a ranked list of concept names for each semantic class.

3 Feature Identification

We start by identifying features from a corpus of text which we refer to as the feature selection data. This data can be the test data, i.e., the contexts to be clustered (each of which contain an occurrence of the ambiguous name) or it may be a separate corpus. The identified features are used to translate each context in the test data to a vector form.

We are exploring the use of bigrams as our feature type. These are lexical features that consist of an ordered pair of words which may occur next to each other, or have one intervening word. We are interested in bigrams since they tend to be less ambiguous and more specific than individual unigrams. In order to reduce the amount of noise in the feature set, we discard all bigrams that occur only once, or that have a log-likelihood ratio of less than 3.841. The latter criteria indicates that the words in the bigram are not independent (i.e., are associated) with 95% certainty. In addition, bigrams in which either word is a stop word are filtered out.

4 Context Representation

We employ both first and second order representations of the contexts to be clustered. The first order representation is a vector that indicates which of the features identified during the feature selection process occur in this context.

The second order context representation is adapted from (Schütze, 1998). First a co-occurrence matrix is constructed from the features identified in the earlier stage, where the rows represent the first word in the bigram, and the columns represent the second word. Each cell contains the value of the log-likelihood ratio for its respective row and column word-pair.

This matrix is both large and sparse, so we use Singular Value Decomposition (SVD) to reduce the dimensionality and smooth the sparsity. SVD has the effect of compressing similar columns together, and then reorganizing the matrix so that the most significant of these columns come first in the matrix. This allows the matrix to be represented more compactly by a smaller number of these compressed columns.

The matrix is reduced by a factor equal to the minimum of 10% of the original columns, or 300. If the original number of columns is less than 3,000 then the matrix is reduced to 10% of the number of columns. If the matrix has greater than 3,000 columns, then it is reduced to 300.

Each row in the resulting matrix is a vector for the word the row represents. For the second order representation, each context in the test data is represented by a vector which is created by averaging the word vectors for all the words in the context.

The philosophy behind the second order representation is that it captures indirect relationships between bigrams which cannot be done using the first order representation. For example if the word *ergonomics* occurs along with *science*, and *workplace* occurs with *science*, but not with *ergonomics*, then *workplace* and *ergonomics* are second order co-occurrences by virtue of their respective co-occurrences with *science*.

Once the context is represented by either a first order or a second order vector, then clustering can follow. A hybrid method known as Repeated Bisections is employed, which tries to balance the quality of agglomerative clustering with the speed of partitional methods. In our current approach the number of clusters to be discovered must be specified. Making it possible to automatically identify the number of clusters is one of our high priorities for future work.

5 Labeling

Once the clusters are created, we assign each cluster a *descriptive* and *discriminating* label. A label is a list of bigrams that act as a simple summary of the contents of the cluster.

Our current approach for *descriptive* labels is to select the top N bigrams from contexts grouped in a cluster. We use similar techniques as we use for feature identification, except now we apply them on the clustered contexts. In particular, we select the top 5 or 10 bigrams as ranked by the log-likelihood ratio. We discard bigrams if either of the words is a stop-word, or if the bigram occurs only one time. For *discriminating* labels we pick the top 5 or 10 bigrams which are unique to the cluster and thus capture the contents that separates one cluster from another.

6 Experimental Data

Our experimental data consists of two or more unambiguous names whose occurrences in a corpus have been conflated in order to create ambiguity. These conflated forms are sometimes known as pseudo words. For example, we take all occurrences of Tony Blair and Bill Clinton and conflate them into a single name that we then attempt to discriminate.

Further, we believe that the use of artificial pseudo words is suitable for the problem of name discrimination, perhaps more so than is the case in word sense disambiguation in general. For words there is always a debate as to what constitutes a word sense, and how finely drawn a sense distinction should be made. However, when given an ambiguous name there are distinct underlying entities associated with that name, so evaluation relative to such true categories is realistic.

Our source of data is the New York Times (January 2000 to June 2002) corpus that is included as a part of the English GigaWord corpus.

In creating the contexts that include our conflated names, we retain 25 words of text to the left and also to the right of the ambiguous conflated name. We also preserve the original names in a separate tag for the evaluation stage.

We have created three levels of ambiguity: 2-way, 3-way, and 4-way. In each of the three categories we have 3-4 examples that represent a variety of different degrees of ambiguity. We have created several examples of intra-category disambiguation, including Bill Clinton and Tony Blair (political leaders), and Mexico and India (countries). We also have inter-category disambiguation such as Bayer, Bank of America, and John Grisham (two companies and an author).

The 3-way examples have been chosen by adding one more dimension to the 2-way examples. For example, Ehud Barak is added to Bill Clinton and Tony Blair, and the 4-way examples are selected on similar lines.

7 Experimental Results

Table 1 summarizes the results of our experiments in terms of the F-Measure, which is the harmonic mean of precision and recall. Precision is the percentage of contexts clustered correctly out of those that were attempted. Recall is the percentage of contexts clustered correctly out of the total number of contexts given.

The variable M in Table 1 shows the number of contexts of that target name in the input data. Note that we divide the total input data into equal-sized test and feature selection files, so the number of feature selection and test contexts is half of what is shown, with approximately the same distribution of names. (N) specifies the total number of contexts in the input data. MAJ. represents the percentage of the majority name in the data as a whole, and can be viewed as a baseline measure of performance that

Table 1: Experimental Results (F-measure)

	MAJ.	**K**	**Order 1**		**Order 2**			
Target Word(M);+	**(N)**		FSD	TST	FSD	FSD/S	TST	TST/S
BAYER(1271);	60.0	2	67.2	68.6	**71.0**	51.3	**69.2**	53.2
BOAMERICA(846)	(2117)	6	37.4	33.9	47.2	**53.3**	42.8	49.6
BCLINTON(1900);	50.0	2	82.2	**87.6**	81.1	81.2	81.2	70.3
TBLAIR(1900)	(3800)	6	58.5	61.6	61.8	**71.4**	61.5	**72.3**
MEXICO(1500);	50.0	2	42.3	52.4	52.7	**54.5**	52.6	**54.5**
INDIA(1500)	(3000)	6	28.4	36.6	37.5	49.0	37.9	**52.4**
THANKS(817);	55.6	2	61.2	**65.3**	61.4	56.7	61.4	56.7
RCROWE(652)	(1469)	6	36.3	41.2	38.5	**52.0**	39.9	47.8
BAYER(1271);BOAMERICA(846);	43.2	3	69.7	**73.7**	57.1	54.7	55.1	54.7
JGRISHAM(828);	(2945)	6	31.5	38.4	32.7	**53.1**	32.8	**52.8**
BCLINTON(1900);TBLAIR(1900);	33.3	3	51.4	**56.4**	47.7	44.8	47.7	44.9
EBARAK(1900);	(5700)	6	**58.0**	54.1	43.8	48.1	43.7	48.1
MEXICO(1500);INDIA(1500);	33.3	3	**40.4**	**41.7**	38.1	36.5	38.2	37.4
CALIFORNIA(1500)	(4500)	6	31.5	**38.4**	32.7	**36.2**	32.8	**36.2**
THANKS(817);RCROWE(652);	35.4	4	42.7	**61.5**	42.9	38.5	42.7	37.6
BAYER(1271);BOAMERICA(846)	(3586)	6	47.0	**53.0**	43.9	34.0	43.5	34.6
BCLINTON(1900);TBLAIR(1900);	25.0	4	48.4	**52.3**	44.2	**50.1**	44.7	**51.4**
EBARAK(1900);VPUTIN(1900)	(7600)	6	**51.8**	47.8	43.4	**49.3**	44.4	**50.6**
MEXICO(1500);INDIA(1500);	25.0	4	**34.4**	**35.7**	29.2	27.4	29.2	27.1
CALIFORNIA(1500);PERU(1500)	(6000)	6	**31.3**	**32.0**	27.3	27.2	27.2	27.2

Table 2: Sense Assignment Matrix (2-way)

	TBlair	BClinton	
C0	784	50	834
C1	139	845	984
	923	895	1818

Table 3: Sense Assignment Matrix (3-way)

	BClinton	TBlair	EBarak	
C0	617	57	30	704
C1	65	613	558	1236
C2	215	262	356	833
	897	932	944	2773

would be achieved if all the contexts to be clustered were placed in a single cluster.

K is the number of clusters that the method will attempt to classify the contexts into. FSD are the experiments where a separate set of data is used as the feature selection data. TST are the experiments where the features are extracted from the test data. For FSD and TST experiments, the complete context was used to create the context vector to be clustered, whereas for FSD/S and TST/S in the order 2 experiments, only the five words on either side of the target name are averaged to form the context-vector.

For each name conflated sample we evaluate our methods by setting K to the exact number of clusters, and then for 6 clusters. The motivation for the higher value is to see how well the method performs when the exact number of clusters is unknown. Our belief is that with an artificially- high number specified, some of the resulting clusters will be nearly empty, and the overall results will still be reasonable. In addition, we have found that the precision of the clusters associated with the known names remains high, while the overall recall is reduced due to the clusters that can not be associated with a name.

To evaluate the performance of the clustering,

Table 4: Labels for Name Discrimination Clusters (found in Table 1)

Original Name	Type	Created Labels
CLUSTER 0: TONY BLAIR	Desc.	Britain, British Prime, Camp David, Middle East, Minister, New York, Prime, Prime Minister, U S, Yasser Arafat
	Disc.	Britain, British Prime, Middle East, Minister, Prime, Prime Minister
CLUSTER 1: BILL CLINTON	Desc.	Al Gore, Ariel Sharon, Camp David, George W, New York, U S, W Bush, White House, prime minister
	Disc.	Al Gore, Ariel Sharon, George W, W Bush
CLUSTER 2: EHUD BARAK	Desc.	Bill Clinton, Camp David, New York, President, U S, White House, Yasser Arafat, York Times, minister, prime minister
	Disc.	Bill Clinton, President, York Times, minister

a contingency matrix (e.g., Table 2 or 3) is constructed. The columns are re-arranged to maximize the sum of the cells along the main diagonal. This re-arranged matrix decides the sense that gets assigned to the cluster.

8 Discussion

The order 2 experiments show that limiting the scope in the test contexts (and thereby creating an averaged vector from a subset of the context) is more effective than using the entire context. This corresponds to the findings of (Pedersen et.al., 2005). The words closest to the target name are most likely to contain identifying information, whereas those that are further away may be more likely to introduce noise.

As the amount and the number of contexts to be clustered (and to be used for feature identification) increases, the order 1 context representation performs better. This is because in the larger samples of data it is more likely to find an exact match for a feature and thereby achieve overall better results. We believe that this is why the order 1 results are generally better for the 3-way and 4-way distinctions, as opposed to the 2-way distinctions. This observation is consistent with earlier findings by Purandare and Pedersen for general English text.

An example of a 2-way clustering is shown in Table 2, where Cluster 0 is assigned to Tony Blair, and Cluster 1 is for Bill Clinton. In this case the precision is 89.60 ((1629/1818)*100), whereas the recall is 85.69 ((1629/1818+83)*100). This suggests that there were 83 contexts that the clustering algorithm was unable to assign, and so they were not clustered and removed from the results.

Table 3 shows the contingency matrix for a 3-way ambiguity. The distribution of contexts in cluster 0 show that the single predominant sense in the cluster is Bill Clinton, but for cluster 1 though the number of contexts indicate clear demarcation between BClinton and TBlair, this distinction gets less clear between TBlair and EBarak. This suggests that perhaps the level of details in the New York Times regarding Bill Clinton and his activities may have been greater than that for the two non-US leaders, although we will continue to analyze results of this nature.

We can see from the labeling results shown in Table 4 that clustering performance affects the quality of cluster labels. Thus the quality of labels for cluster assigned to BClinton and TBlair are more suggestive of the underlying entity than are the labels for EBarak clusters.

9 Future Work

We wish to supplement our cluster labeling technique by using World Wide Web (WWW) based methods (like Google-Sets) for finding words related to the target name and other significant words in the context. This would open up a venue for large and multi-dimensional data. We are cautious though that we would have to deal with the problems of noisy data that WWW brings along with the good data. Another means of improving the clustering labeling will be using WordNet::Similarity to find the relatedness amongst the words from the cluster using the knowledge of WordNet as is also proposed by (McCarthy et.al., 2004).

Currently the number of clusters that the contexts should be grouped into has to be specified by the user. We wish to automate this process such that the clustering algorithm will automatically determine the optimal number of clusters. We are exploring a number of options, including the use of GAP statistic (Tibshirani et.al., 2000).

For the order 2 representation of the contexts there is considerable noise induced in the resulting context vector because of the averaging of all the word-vectors. Currently we reduce the noise in the averaged vector by limiting the word vectors to those associated with words that are located near the target name. We also plan to develop methods that select the words to be included in the averaged vector more carefully, with an emphasis on locating the most content rich words in the context.

Thus far we have tested our methods for one-to-many discrimination. This resolves cases where the same name is used by multiple different people. However, we will also test our techniques for the many-to-one kind ambiguity that occurs when the same person is referred by multiple names, e.g., President Bush, George Bush, Mr. Bush, and President George W. Bush.

Finally, we will also evaluate our method on real data. In particular, we will use the John Smith Corpus as compiled by Bagga and Baldwin, and the name data generated by Mann and Yarowsky for their experiments.

10 Conclusions

We have shown that word sense discrimination techniques can be extended to address the problem of name discrimination. The experiments with second order context representation work better with limited or localized scope. As the dimensionality of the ambiguity increases first order context representation out-performs second order representation. The labeling of clusters using the simple technique of significant bigram selection also shows encouraging results which highly depends on the performance of the clustering of contexts.

11 Acknowledgments

I would like to thank my advisor Dr. Ted Pedersen for his continual guidance and support.

I would also like to thank Dr. James Riehl, Dean of the College of Science and Engineering, and Dr. Carolyn Crouch, Director of Graduate Studies in Computer Science, for awarding funds to partially cover the expenses to attend the Student Research Workshop at ACL 2005.

I am also thankful to Dr. Regina Barzilay and the ACL Student Research Workshop organizers for awarding the travel grant.

This research has been supported by a National Science Foundation Faculty Early CAREER Development Award (#0092784) during the 2004-2005 academic year.

References

Amruta Purandare and Ted Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. *The Proceedings of the Conference on Computational Natural Language Learning*, pages 41-48. Boston, MA.

Gideon Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. *The Proceedings of the Conference on Computational Natural Language Learning*, pages 33-40. Edmonton, Canada.

Amit Bagga and Breck Baldwin. 1998. Entity–based cross–document co–referencing using the vector space model. *The Proceedings of the 17th international conference on Computational linguistics*, pages 79-85. Montreal, Quebec, Canada.

Patrick Pantel and Deepak Ravichandran. 2004. Automatically Labeling Semantic Classes. *The Proceedings of HLT-NAACL*, pages 321-328. Boston, MA.

Diana McCarthy, Rob Koeling, Julie Weeds and John Carroll. 2004. Finding Predominant Word Senses in Untagged Text. *The Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 279-286. Barcelona, Spain.

Robert Tibshirani, Guenther Walther and Trevor Hastie. 2000. Estimating the number of clusters in a dataset via the Gap statistic. *Journal of the Royal Statistics Society (Series B)*, 2000.

Ted Pedersen, Amruta Purandare and Anagha Kulkarni 2005. Name Discrimination by Clustering Similar Contexts. *The Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 226-237. Mexico City, Mexico.

Schütze H. 1998. Automatic Word Sense Discrimination *Computational Linguistics*, 24(1):97-124.

A Domain-Specific Statistical Surface Realizer

Jeffrey T. Russell
Center for the Study of Language and Information
Stanford University
jefe@stanford.edu

Abstract

We present a search-based approach to automatic surface realization given a corpus of domain sentences. Using heuristic search based on a statistical language model and a structure we introduce called an *inheritance table* we overgenerate a set of complete syntactic-semantic trees that are consistent with the given semantic structure and have high likelihood relative to the language model. These trees are then lexicalized, linearized, scored, and ranked. This model is being developed to generate real-time navigation instructions.

1 Introduction

The target application for this work is real-time, interactive navigation instructions. Good direction-givers respond actively to a driver's actions and questions, and express instructions relative to a large variety of landmarks, times, and distances. These traits require robust, real-time natural language generation. This can be broken into three steps: (1) generating a route plan, (2) reasoning about the route and the user to produce an abstract representation of individual instructions, and (3) realizing these instructions as sentences in natural language (in our case, English). We focus on the last of these steps: given a structure that represents the semantic content of a sentence, we want to produce an English sentence that expresses this content. According to the traditional division of content determination, sentence planning, and surface realization, our work is primarily concerned with surface realization, but also includes aspects of sentence planning. Our application requires robust flexibility within a restricted domain that is not well represented in the traditional corpora or tools. These requirements suggest using trainable stochastic generation.

A number of statistical surface realizers have been described, notably the FERGUS (Bangalore and Rambow, 2000) and HALogen systems (Langkilde-Geary, 2002), as well as experiments in (Ratnaparkhi, 2000). FERGUS (Flexible Empiricist/Rationalist Generation Using Syntax) takes as input a dependency tree whose nodes are marked with lexemes only. The generator automatically "supertags" each input node with a TAG tree, then produces a lattice of all possible linearizations consistent with the supertagged dependency tree. Finally it selects the most likely traversal of this lattice, conditioned on a domain-trained language model. The HALogen system is a broad-coverage generator that uses a combination of statistical and symbolic techniques. The input, a structure of feature-value pairs (see Section 3.1), is symbolically transformed into a forest of possible expressions, which are then ranked using a corpus-trained statistical language model. Ratnaparkhi also uses an overgeneration approach, using search to generate candidate sentences which are then scored and ranked. His paper outlines experiments with an n-gram model, a trained dependency grammar, and finally a hand-built grammar including content-driven conditions for applying rules. The last of these systems outperformed the n-gram and trained grammar in testing based on human judgments.

Proceedings of the ACL Student Research Workshop, pages 151–156,
Ann Arbor, Michigan, June 2005. ©2005 Association for Computational Linguistics

The basic idea of our system fits in the overgenerate-and-rank paradigm. Our approach is partly motivated by the idea of 'softening' Ratnaparkhi's third system, replacing the hand-built grammar rules with a combination of a trained statistical language model and a structure called an *inheritance table,* which captures long-run dependency information. This allows us to overgenerate based on rules that are sensitive to structured content without incurring the cost of designing such rules by hand.

2 Algorithm

We use dependency tree representations for both the semantics and syntax of a sentence; we introduce the *syntactic-semantic (SS) tree* to combine information from both of these structures. An SS tree is constructed by "attaching" some of the nodes of a sentence's semantic tree to the nodes of its syntactic tree, obeying two rules:

- **Each node in the semantic tree is attached to at most one node of the syntactic tree.**

- **Semantic and syntactic hierarchical orderings are consistent.** That is to say, if two semantic nodes x_1 and x_2 are attached to two syntactic nodes y_1 and y_2, respectively, then x_1 is a descendant of x_2 in the semantic tree if and only if y_1 is a descendant of y_2 in the syntactic tree.

The nodes of an SS tree are either unattached semantic or syntactic nodes, or else pairs of attached nodes. The SS tree's hierarchy is consistent with the hierarchies in the syntactic and semantic trees. We say that an SS tree T *satisfies* a semantic structure S if S is embedded in T. This serves as formalization of the idea of a sentence expressing a certain content.

2.1 Outline

The core of our method is a heuristic search of the space of possible SS trees. Our search goal is to find the N best complete SS trees that express the given semantic structure. We take 'best' here to be the trees which have the highest conditional likelihood given that they express the right semantic structure. If S is our semantic structure and LM is our statistical language model, we want to find syntactic trees T that maximize $P_{LM}(T|S)$.

In order to search the space of trees, we build up trees by expanding one node at a time. During the search, then, we deal with *incomplete trees*; that is, trees with some nodes not fully expanded. This means that we need a way to determine how promising an incomplete tree T is: i.e., how good the best complete trees are that can be built up by expanding T. As it turns out (Section 2.2), we can efficiently approximate the function[1] $P_{LM}(T|S)$ for an incomplete tree, and this function is a good heuristic for the maximum likelihood of a complete tree extended from T.

Here is an outline of the algorithm:

- Start with a root tree.
  - Take the top N trees and expand one node in each.
  - Score each expanded tree for $P_{LM}(T|S)$, and put in the search order accordingly.
  - Repeat until we find enough trees that satisfy S.
- Complete the trees.
- Linearize and lexicalize the trees.
- Rank the complete trees according to some scoring function.

2.2 Heuristic

Our search goal is to maximize $P_{LM}(T|S)$. (Henceforth we abbreviate P_{LM} as just P.) Ideally, then, we would at each step expand the incomplete tree that can be extended to the highest-likelihood complete tree, i.e. that has the highest value of $\max_{T'} P(T'|S)$ over all complete trees T' that extend T. We use the notation $T' > T$ when T' is a complete tree that extends an incomplete tree T, and the notation $T' \triangleright S$ when T' satisfies S. Then the "goodness" of a tree T is given by

$$\max_{T'>T} P(T'|S) = \max_{T'>T; T'\triangleright S} P(T')/P(S) \qquad (1)$$

[1] This probability is defined to be the sum of the probabilities $P_{LM}(T|T')P_{LM}(T'|S)$ for all complete trees T'

Since finding this maximum explicitly is not feasible, we use the heuristic $P(T|S)$. By Bayes' rule, $P(T|S) = P(S|T)P(T)/P(S)$, where $P(S)$ is a normalizing factor, $P(T)$ can be easily calculated using the language model (as the product of the probabilities of the node expansions that appear in T), and

$$P(S|T) = \sum_{T'} P(S|T')P(T'|T) = \sum_{T' \triangleright S} P(T'|T)$$

Since $P(T'|T) = P(T|T')P(T')/P(T)$, and since $P(T|T')$ is 1 if $T' > T$ and 0 otherwise, we have

$$P(T|S) = \frac{1}{P(S)} \sum_{T' \triangleright S} P(T|T')P(T')$$

$$= \frac{1}{P(S)} \sum_{T' > T; T' \triangleright S} P(T')$$

Together with Equation 1 this shows that $P(T|S) \geq \max_{T'>T} P(T'|S)$, since the maximum is one of the terms in the sum. This fact is analogous to showing that $P(T|S)$ is an admissible heuristic (in the sense of A* search).

We can see how to calculate $P(T|S)$ in practice by decomposing the structure of a tree T' such that $T' > T$ and $T' \triangleright S$. Since T' extends T, the top of T' is identical to T. The semantic tree S will have some of its nodes in T, and some in the part of T' that extends beyond T. Let $\alpha(S, T)$ be the set containing the highest nodes in S that are not in T. Each node $s \in \alpha(S, T)$ is the root node of a subtree in T'. Each of these subtrees can be considered separately.

First we consider how the these subtrees are joined to the nodes in T. The condition of consistent ordering requires that each node in $\alpha(S, T)$ be a descendant in T' of its parent in S, and moreover it should not be a descendant of any of its siblings in S. Let sib be a set of siblings in $\alpha(S, T)$, and let p be their semantic parent. Then p is the root node of a subtree of T, called T_p. We will designate the *T-set* of sib as the set of leaves of T_p that are not descended from any nodes in S below p–in particular, that are not descended from any other siblings of the nodes in sib. Then in T' all of the nodes in sib must descend from the T-set of sib. In other words, there is a set of subtrees of T' which are rooted at the nodes in the T-set of sib, and all of the nodes in sib appear in these subtrees such that none of them are descended from each other.

This analysis sets us up to rewrite $P(T|S)$ in terms of sums over these various subtrees. We use the notation $P(\{x_1, ..., x_k\} \rightarrow \{y_1, ..., y_l\})$ to denote the probability that the nodes $y_1, ..., y_l$ eventually descend from $x_1, ..., x_k$ without dominating each other; this probability is the sum of $P(T_1, ..., T_k)$ over all sets of trees $T_1 > x_1, ..., T_k > x_k$ such that each node $y_1, ..., y_l$ appears in some T_i and no y_i descends from any y_j. Then we can rewrite $P(T|S)$ as

$$\frac{P(T)}{P(S)} \prod_{sib} P(\text{T-set}(sib) \rightarrow sib) \prod_{x \in \alpha(S,T)} P(x \rightarrow S_x) \tag{2}$$

S_x denotes the subtree of S whose root node is x. $P(x \rightarrow S_x)$ is 1 if S_x contains only the node x, and otherwise is

$$P(x \rightarrow \text{children}_S(x)) \left(\prod_{y \in \text{children}_S(x)} P(y \rightarrow S_y) \right)$$

Rather than calculating the value of formula 2 exactly, we now introduce an approximation to our heuristic function. For sets X, Y, we approximate $P(X \rightarrow Y)$ with $\prod_{y \in Y} P(X \rightarrow y)$. This amounts to two simplifications: first, we drop the restriction that no node be descended from its semantic sibling; second, we assume that the probabilities of each node descending from X are independent from one another.

$P(X \rightarrow y)$ is the probability that at least one $x \in X$ has y as a descendant, i.e. $P(X \rightarrow y) = \text{AL1}_{x \in X} P(x \rightarrow y)$, where AL1 is the 'At-least-one' function.[2] This means that we can approximate $P(T|S)$ as

$$\frac{P(T)}{P(S)} \prod_{y \in \alpha(S,T)} \text{AL1}_{x \in \text{T-set}(y)} P(x \rightarrow y) P(y \rightarrow S_y) \tag{3}$$

[2] That is, given the probabilities of a set of events, the At-least-one function gives the probability of at least one of the events occuring. For independent events, $\text{AL1}\{\} = 0$ and $\text{AL1}\{p_1, ..., p_n\} = p_n + (1 - p_n)\text{AL1}\{p_1, ..., p_{n-1}\}$.

The calculation of $P(T|S)$ has been reduced to finding $P(x \rightarrow y)$ for individual nodes. These values are retrieved from the inheritance table, described below.

Note that when we expand a single node of an incomplete tree, only a few factors in Equation 3 change. Rather than recalculating each tree's score from scratch, then, by caching intermediate results we can recompute only the terms that change. This allows for efficient calculation of the heuristic function.

2.3 Inheritance Table

The inheritance table (IT) allows us to predict the potential descendants of an incomplete tree. For each pair of SS nodes x and y, the IT stores $P(x \rightarrow y)$, the probability that y will eventually appear as a descendant of x. The IT is precomputed once from the language model; the same IT is used for all queries.

We can compute the IT using an iterative process. Consider the transformation $\mathbf{T}$ that takes a distribution $Q(x \rightarrow y)$ to a new distribution $\mathbf{T}(Q)$ such that $\mathbf{T}(Q)(x \rightarrow y)$ is equal to 1 when $x = y$, and otherwise is equal to

$$\sum_{\zeta \in Exp(x)} P_{LM}(\zeta|x) \mathrm{AL1}_{z \in \zeta} Q(z \rightarrow y) \qquad (4)$$

Here $Exp(x)$ is the set of possible expansions of x, and $P_{LM}(\zeta|x)$ is the probability of the expansion ζ according to the language model.

The defining property of the IT's distribution P is that $\mathbf{T}(P) = P$. We can use this property to compute the table iteratively. Begin by setting $P_0(x \rightarrow y)$ to 1 when $x = y$ and 0 otherwise. Then at each step let $P_{k+1} = \mathbf{T}(P_k)$. When this process converges, the limiting function is the correct inheritance distribution.

2.4 Completing Trees

A final important issue is termination. Ordinarily, it would be sensible to remove a tree from the search order only when it is a goal state—that is, if it is a complete tree that satisfies S. However, this turns out to be not the best approach in this case due to a quirk of our heuristic. $P(T|S)$ has two non-constant factors, $P(S|T)$ and $P(T)$. Once all of the nodes in S appear in an incomplete tree T, $P(S|T) = 1$, and so it won't increase as the tree is expanded further. Moreover, with each node expanded, $P(T)$ decreases. This means that we are unlikely to make progress beyond the point where all of the semantic content appears in a tree.

An effective way to deal with this is to remove trees from the search order as soon as $P(S|T)$ reaches 1. When the search terminates by finding enough of these 'almost complete' trees, these trees are completed: we find the optimal complete trees by repeatedly expanding the N most likely almost-complete trees (ranked by $P(T)$) until sufficiently many complete trees are found.

3 Implementation

3.1 Representation

Our semantic representation is based on the HALogen input structure (Langkilde-Geary, 2002). The meaning of a sentence is represented by a tree whose nodes are each marked with a concept and a semantic role. For example, the meaning of the sentence "Turn left at the second traffic light" is represented by the following structure:

```
(maketurn
   :direction (left)
   :spatial-locating
     (trafficlight
        :modifier (second)))
```

The syntax model we use is statistical dependency grammar. As we outlined in Section 2, the semantic and syntactic structures are attached to one another in an SS tree. In order to accomodate the requirement that each semantic node is attached to no more than one syntactic node, collocations like "traffic light" or "John Hancock Tower", are treated as single syntactic nodes. It can also be convenient to extend this idea, treating phrases like "turn around" or "thank you very much" as atomic. In the case where a concept attaches to multi-word expression, but where it is inconvenient to treat the expression as a syntactic atom, we adopt the convention of attaching the concept to the hierarchically dominant word in the expression. For instance, the concept of turning can be attached to the expression "make a turn”; in this case we attach the concept to the word “make”, and not to “turn”.

The nodes of an SS tree are (word, part of speech, concept, semantic role) 4-tuples, where the concept and role are left empty for function words, and the word and part of speech are left empty for concepts with no direct syntactic correlate. Generally we omit the word itself from the tree in order to mitigate sparsity issues; these are added to the final full tree by a lexical choice module.

We use a domain-trained language model based on the same dependency structure as our syntactic-semantic representations. The currently implemented model calculates the probability of expansions given a parent node based on an explicit tabular representation of the distribution $P(\zeta|x)$ for each x. This language model is also used to score and rank generated sentences.

3.2 Corpus and Annotation

Training this language model requires an annotated corpus of in-domain text. Our main corpus comes from transcripts of direction-giving in a simulation context, collected using the “Wizard of Oz” set-up described in (Cheng et al., 2004). For development and testing, we extracted approximately 600 instructions, divided into training and test sets. The training set was used to train the language model used for search, the lexical choice module, and the scoring function. Both sets both underwent four partially-automated stages of annotation.

First we tag words with their part of speech, using the Brill tagger with manually modified lexicon and transformation rules for our domain (Brill, 1995). Second, the words are disambiguated and assigned a concept tag. For this we construct a domain ontology, which is used to automatically tag the unambiguous words and prompt for human disambiguation in the remaining cases. The third step is to assign semantic roles. This is accomplished by using a list of contextual rules, similar to the rules used by the Brill tagger. For example, the rule

> CON intersection PREV1OR2OR3WD at
> : spatial-locating

assigns the role “spatial-locating” to a word whose concept is “intersection” if the word “at” appears one, two, or three words before it. A segment of the corpus was automatically annotated using such rules, then a human annotater made corrections and added new rules, repeating these steps until the corpus was fully annotated with semantic roles.

After the first three stages, the sentence, “Turn left at the next intersection” is annotated as follows:

```
turn/VB/maketurn left/RB/
$leftright/direction at/IN the/
DT next/JJ/first/modifier
intersection/NN/intersection/
spatial-locating
```

The final annotation step is parsing. For this we use an approach similar to Pereira and Schabes’ grammar induction from partially bracketed text (Pereira and Schabes, 1992). First we annotate a segment of the corpus. Then we use the *inside-outside* algorithm to simultaneously train a dependency grammar and complete the annotation. We then manually correct a further segment of the annotation, and repeat until acceptable parses are obtained.

3.3 Rendering

Linearizing an SS tree amounts to deciding the order of the branches and whether each appears on the left or the right side of the head. We built this information into our language model, so a grammar rule for expanding a node includes full ordering information. This makes the linearization step trivial at the cost of adding sparsity to the language model.

Lexicalization could be relegated to the language model in the same way, by including lexemes in the representation of each node, but again this would incur sparsity costs. The other option is to delegate lexical choice to a separate module, which takes a SS tree and assigns a word to each node. We use a hybrid approach: content words are assigned using a lexical choice module, while most function words are included explicitly in the language model. The current lexical choice module simply assigns each unlabeled node the most likely word conditioned on its (POS, concept, role) triple, as observed in the training corpus.

4 Example

We take the semantic structure presented in Section 3.1 as an example generation query. The search stage terminates when 100 trees that embed this semantic structure have been found. The best-scoring sentence has the following lexicalized tree:

```
turn/VB/maketurn
  +left/RB/$leftright/direction
  +at/IN
     +traffic_light/NN/
      trafficlight/
      spatial-locating
        -the/DT
        +next/JJ/first/modifier
```

This is finally rendered thus:

turn left at the second traffic_light.

5 Preliminary Results

For initial testing, we separated the annotated corpus into a 565-sentence training set and a 57-sentence test set. We automatically extracted semantic structures from the test set, then used these structures as generation queries, returning only the highest-ranked sentence for each query. The generated results were then evaluated by three independent human annotaters along two dimensions: (1) Is the generated sentence grammatical? (2) Does the generated sentence have the same meaning as the original sentence?

For 11 of the 57 sentences (19%), the query extraction failed due to inadequate grammar coverage.[3] Of the 46 instances where a query was successfully extracted, 3 queries (7%) timed out without producing output. Averaging the annotaters' judgments, 1 generated sentence (2%) was ungrammatical, and 3 generated sentences (7%) had different meanings from their originals. 39 queries (85%) produced output that was both grammatical and faithful to the original sentence's meaning.

6 Future Work

Statistically-driven search offers a means of efficiently overgenerating sentences to express a given semantic structure. This is well-suited not only to our navigation domain, but also to other domains with a relatively small vocabulary but variable and complex content structure. Our implementation of the idea of this paper is under development in a number of directions.

A better option for **robust language modeling** is to use maximum entropy techniques to train a feature-based model. For instance, we can determine the probability of each child using such features as the POS, concept, and role of the parent and previous siblings. It may also be more effective to isolate **linear precedence** from the language model, introducing a non-trivial linearization step. Similarly, the **lexicalization module** can be improved on by using a more context-sensitive model.

Using only a tree-based **scoring function** is likely to produce inferior results to one that incorporates a linear score. A weighted average of the dependency score with an n-gram model would already offer improvement. To further improve fluency, these could also be combined with a scoring function that takes longer-range dependencies into account, as well as penalizing extraneous content.

References

Srinivas Bangalore and O. Rambow. 2000. Using TAG, a Tree Model, and a Language Model for Generation. *5th Int'l Workshop on Tree-Adjoining Grammars (TAG+)*, TALANA, Paris.

Eric Brill. 1995. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. *Computational Linguistics*, 21 (4).

Hua Cheng, H. Bratt, R. Mishra, E. Shriberg, S. Upson, J. Chen, F. Weng, S. Peters, L. Cavedon and J. Niekrasz. 2004. A Wizard Of OZ Framework for Collecting Spoken Human-Computer Dialogs. *Proc. 8th ICSLP*, Jeju Island, Korea.

Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. *Proc. 2nd INLG*, Harriman, NY.

Fernando Pereira and Y. Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. *Proc. 30th ACL*, p.128-135, Newark.

Adwait Ratnaparkhi. 2000. Trainable methods for surface natural language generation. *Proc. 1st NAACL*, Seattle.

[3] The corpus was partially annotated for parse data, the full parses being automatically generated from the domain-trained language model. It was at this step that query extraction sometimes failed.

Index