Australasian Language Technology Association Workshop 2010

Proceedings of the Workshop

Editors:
Nitin Indurkhya
Simon Zwarts

9-10 December 2010
University of Melbourne
Melbourne, Australia

Australasian Language Technology Association Workshop 2010 (ALTA 2010)
URL: http://www.alta.asn.au/events/alata2010

Sponsors:

Volume 8, 2010
ISSN: 1834-7037

ALTA 2010 Workshop Committees

Workshop Co-Chairs

- Nitin Indurkhya (University of New South Wales and eBay Research Labs)
- Simon Zwarts (Macquarie University)

Workshop Local Organisers

- David Martinez (University of Melbourne)
- Steven Bird (University of Melbourne)

Program Committee

- Achim Hoffmann (University of New South Wales)
- Adam Saulwick (DSTO)
- Alistair Knott (University of Otago, New Zealand)
- Andrea Schalley (Griffith University)
- Andrew Lampert (CSIRO)
- Ben Hachey (CMCRC)
- Caroline Gasperin (Universidade de Sao Paulo, Brasil)
- Cecile Paris (CSIRO)
- David M. W. Powers (Flinders University)
- David Martinez (University of Melbourne)
- Diego Molla Aliod (Macquarie University)
- Dominique Estival (University of Western Sydney)
- Dongqiang Yang (Flinders University)
- Eric Choi (NICTA)
- Francis Bond (Nanyang Technological University, Singapore)
- Jean-Yves Delort (CMCRC)
- Jette Viethen (Macquarie University)
- Kazunori Komatani (Nagoya University, Japan)
- Luiz Augusto Sangoi Pizzato (University of Sydney)
- Mark Dras (Macquarie University)
- Matthew Honnibal (University of Sydney)
- Menno van Zaanen (Tilburg University, The Netherlands)
- Nigel Collier (National Institute of Informatics, Japan)
- Rolf Schwitter (Macquarie University)
- Scott Nowson (Appen Pty Ltd)
- Son Bao Pham (Vietnam National University, Vietnam)
- Steven Bird (University of Melbourne)
- Tara McIntosh (NICTA)
- Timothy Baldwin (University of Melbourne)
- Wayne Wobcke (University of New South Wales)

Preface

This volume contains the papers accepted for presentation at the Australasian Language Technology Workshop (ALTA) 2010, held at the University of Melbourne, Melbourne, Australia on December 9-10, 2010. This is the eighth annual instalment of the ALTA workshop in its most-recent incarnation, and the continuation of an annual workshop series that has existed in various forms Down Under since the early 1990s.

The goals of the workshop are:

- to bring together the growing Language Technology (LT) community in Australia and New Zealand and encourage interactions;
- to encourage interactions and collaboration within the community and with the wider international LT community;
- to foster interaction between academic and industrial researchers, to encourage dissemination of research results;
- to provide a forum for the discussion of new and ongoing research and projects;
- to provide an opportunity for the broader artificial intelligence community to become aware of local LT research;
- and finally, to increase visibility of LT research in Australia, New Zealand and overseas.

This year's ALTA Workshop includes full papers (9 pages in length) as well as short papers (5 pages in length). We received a total of 19 submissions including 5 short papers. 12 papers including 2 short papers were selected by the program committee for publication in these proceedings.

Each paper in the 'reviewed papers' section was independently peer-reviewed by at least two members of an international program committee, in accordance with the DEST requirements for E1 conference publications. The review process was double-blind: Great care was exercised to avoid all conflicts of interest whenever an author also served as program committee/co-chair or the reviewer worked at the same institution as an author. Such conflicts of interest were resolved by transferring the reviewing task to other members of the program committee.

A key feature of this year's workshop is a special session of invited speakers from industry. In this 'industry session', which aims to bridge the gap between academia and industry, members of different companies talk about how they apply language technology (research) in their work. Another exciting feature of this year's workshop is the Language Technology Programming Competition. It is formatted as a "shared task": all participants compete to solve the same problem. The problem highlights an active area of research and programming in the area of language technology. Details of the shared task are published in the proceedings and also presented in a special session (along with the winner of the competition.)

We would like to thank all the authors who submitted papers to ALTA, the members of the program committee for the time and effort they put into the review process and to our invited speakers: Rodolfo Delmonte (University Ca'Foscari, Italy) and Casey Whitelaw (Google Inc, Sydney). We would also like to thank all the invited speakers for our industry session for making it such a success and hope that it will feature in future ALTA workshops as well.

Finally, we would like to thank our sponsors, NICTA and the University of Melbourne, for supporting the workshop.

Nitin Indurkhya and Simon Zwarts
Program Co-Chairs

ALTA 2010 Program

Thursday, 9th December 2010

9:00-10:00	Keynote: *Opinion Mining, Subjectivity and Factuality* by Rodolfo Delmonte (University Ca'Foscari, Italy)
10:00-10:30	Coffee break
10:30-10:40	ALTA Opening remarks
Session 1 - 10:40 - 12:45	
	Paper Presentations
10:40-11:05	Shunichi Ishihara *Variability and Consistency in the Idiosyncratic Selection of Fillers in Japanese Monologues: Gender Differences*
11:05-11:30	Michael Curtotti and Eric McCreath *Corpus Based Classification of Text in Australian Contracts*
11:30-11:55	Li Wang, Su Nam Kim and Timothy Baldwin *Thread-level Analysis over Technical User Forum Data*
11:55-12:20	Susan Howlett and Mark Dras *Dual-Path Phrase-Based Statistical Machine Translation*
12:20-12:45	Yue Li and David Martinez *Information Extraction of Multiple Categories from Pathology Reports*
12:45-2:00	Lunch
2:00-3:00	Keynote: *Language Technology: A View From The Trenches* by Casey Whitelaw (Google Inc, Sydney)
3:00-3:30	Coffee break
Session 2 - 3:30 - 5:30	
	Industry Session 1. BinaryPlex (Tim Bull) 2. NICTA (Wray Buntine) 3. Pacific Brands (Yuval Marom) 4. Lexxe (Hong Liang Qiao) 5. Digital Sonata (Vadim Berman) 6. CSIRO (Stephen Wan) 7. Atex (Geoff Wilson) 8. eBay (Nitin Indurkhya) 9. Appen (Scott Nowson) 10. Vicnet/State Library of Victoria (Andrew Cunningham) 11. DSTO (Adam Saulwick)
5:30-	**Informal Q&A followed by Conference Dinner (jointly with ADCS) at 6pm**

Friday, 10th December 2010

Session 3 - 9:00 - 10:40	
	Joint Session with ADCS including ALTA presentations:
	Marco Lui and Timothy Baldwin
	Classifying User Forum Participants: Separating the Gurus from the Hacks, and Other Tales of the Internet
	Alexandra Uitdenbogerd
	Fun with Filtering French
10:40-11:10	Coffee break
11:10-12:00	LT shared talk report
12:00-1:00	**ALTA AGM Meeting** Free Pizza for attendees at the end!!
1:00-2:00	Lunch
Session 4 - 2:00-3:15	
	Paper Presentations
2:00-2:25	Diego Molla
	A Corpus for Evidence Based Medicine Summarisation
2:25-2:50	Sze-Meng Jojo Wong and Mark Dras
	Parser Features for Sentence Grammatical Classification
2:50-3:15	Jette Viethen and Robert Dale
	Speaker-Dependent Variation in Content Selection for Referring Expression Generation
3:15-3:45	Coffee break
Session 5 - 3:45-4:35	
	Paper Presentations
3:45-4:10	Dominick Ng, Matthew Honnibal and James R. Curran
	Reranking a wide-coverage CCG *parser*
4:10-4:35	Simon Zwarts, Mark Johnson and Robert Dale
	Repurposing Corpora for Speech Repair Detection: Two Experiments

Contents

Non-reviewed papers

Opinion Mining, Subjectivity and Factuality

Rodolfo Delmonte
University Ca'Foscari
Italy
delmont@unive.it

Abstract

We assume that in order to properly capture opinion and sentiment expressed in a text or dialog any system needs a deep text processing approach. In particular, the idea that the task may be solved by the use of Information Retrieval tools like Bag of Words Approaches (BOWs) is totally flawed. BOWs approaches are sometimes also camouflaged by a keyword based Ontology matching and Concept search, based on such lexica as SentiWordNet, by simply stemming a text and using content words to match its entries and produce some result. Any search based on keywords and BOWs is fatally flawed by the impossibility to cope with such fundamental issues as the following ones:

- presence of **negation** at different levels of syntactic constituency;
- presence of **lexicalized negation** in the verb or in adverbs;
- presence of conditional, counterfactual subordinators;
- double negations with copulative verbs;
- presence of modals and other modality operators.

In order to cope with these linguistic elements we propose to build a Flat Logical Form (FLF) directly from a Dependency Structure representation augmented by indices and where anaphora resolution has operated pronoun-antecedent substitutions. We implemented these additions our the system called **venses** that we will show. The output of the system is an xml representation where each sentence of a text or dialog is a list of attribute-value pairs, like **polarity**, **attitute** and **factuality**. In order to produce this output, the system makes use of FLF and a vector of semantic attributes associated to the verb at propositional level and then memorized. Important notions required by the computation of opinion and sentiment are also the distinction of the semantic content of each proposition into two separate categories:

- Objective vs Subjective

This distinction is obtained by searching for **factivity** markers again at propositional level. In particular we take into account:

- tense;
- voice;
- mood;
- modality operators;
- modifiers and attributes adjuncts at sentence level;
- lexical type of the verb (in Levin's classes and also using WordNet classification).

Rodolfo Delmonte. 2010. Opinion Mining, Subjectivity and Factuality. In *Proceedings of Australasian Language Technology Association Workshop*, page 2

Language Technology: A View From The Trenches

Casey Whitelaw
Google Inc, Sydney
Australia
whitelaw@google.com

Abstract

Computers that can listen and talk? Helpers that read your books and answer complex questions? Natural language processing has always made big promises, and has left many people with a healthy dose of skepticism. The truth is that language technology has had a bigger impact than you may think, by working behind the scenes.

In this talk, I will discuss the changing role of NLP in applications, and the trends I have observed over the last 5 years with my work at Google. I will be focusing on the practicalities of applying NLP techniques to real-world problems, avoiding common pitfalls, and learning from our most valuable resource: our users.

Casey Whitelaw. 2010. Language Technology: A View From The Trenches. In *Proceedings of Australasian Language Technology Association Workshop*, page 3

Multilingual Language Identification: ALTW 2010 Shared Task Dataset

Timothy Baldwin and Marco Lui
NICTA VRL
Department of Computer Science and Software Engineering
University of Melbourne, VIC 3010, Australia
tb@ldwin.net, saffsd@gmail.com

Abstract

While there has traditionally been strong interest in the task of monolingual language identification, research on multilingual language identification is underrepresented in the literature, partly due to a lack of standardised datasets. This paper describes an artificially-generated dataset for multilingual language identification, as used in the 2010 Australasian Language Technology Workshop shared task.

1 Introduction

Language identification is traditionally defined as the task of determining the unique language a given document is authored in, under the assumption that all documents are monolingual (Baldwin and Lui, to appear). In contexts such as the web, however, multilingual documents are commonplace, suggesting the need for language identification research to move towards a more realistic task setting where a document can be authored in one or more languages (Hughes et al., 2006). This paper describes such a dataset, based around the task of multilingual language identification, where the task is to determine which one or two languages a given document is authored in. This dataset formed the basis of the 2010 Australasian Language Technology Workshop shared task.

Multilingual language identification is relevant in a number of contexts. "Word spotting" of foreign words in multilingual documents has been shown to improve parsing performance (Alex et al., 2007), and multilingual language identification is a first step in this direction. It can also be used as part of a linguistic corpus creation pipeline for low-density languages, e.g. to determine the language used in interlinear glossed text (IGT) embedded in language documentation (Xia et al., 2009; Xia and Lewis, 2009).

The ideal vehicle for multilingual language identification research would be a dataset genuinely representative of the true multilingualism of resources such as the web. Creating such a resource, however, would require: (a) a multilingual crawl without language bias; and (b) a large-scale document collection with gold-standard annotations over the full range of languages extant on the web, including sub-document extents for the individual languages contained in a document. While we would ultimately like to generate such a dataset for general usage, in this paper we describe a more modest effort to artificially generate a dataset for multilingual language identification purposes. Our basic approach is to: (1) select a language bias-preserving set of primary documents; (2) select a comparable document for each in a second language based on translation links; and (3) concatenate sections of the two documents together to form a single multilingual document. In this paper, we detail the methodology for generating the dataset, and outline baseline and benchmark results over the dataset to calibrate future efforts.

2 Dataset Synthesis

The dataset for the task was prepared from database exports of the various language Wikipedias provided by the WikiMedia Foundation.[1] The WikiMedia Foundation carries out an ongoing export of the databases of each of the language-specific Wikipedias, and makes these exports available for download. The exports that we utilized are dated between 9 June 2008 and 1 August 2008. We downloaded all the Wikipedias that exceeded 1000 articles, which at the time numbered 75 (as of October 2010, this number is now almost 200). Of these, the file for the Spanish (es) Wikipedia failed to download correctly,

[1] http://download.wikimedia.org/backup-index.html

Timothy Baldwin and Marco Lui. 2010. Multilingual Language Identification: ALTW 2010 Shared Task Data. In *Proceedings of Australasian Language Technology Association Workshop*, pages 4–7

Lang code	Language name	No. Docs 1°	No. Docs 2°
af	Afrikaans	9	1
an	Aragonese	8	1
ar	Arabic	71	24
ast	Asturian	5	2
az	Azerbaijani	8	2
be	Belarusian	10	0
bg	Bulgarian	57	39
bn	Bengali	24	6
bpy	Bishnupriya	8	10
br	Breton	8	3
bs	Bosnian	26	4
ca	Catalan	105	62
ceb	Cebuano	15	0
cs	Czech	80	37
cy	Welsh	12	4
da	Danish	72	27
de	German	747	1327
el	Modern Greek (1453-)	31	7
en	English	3330	3774
et	Estonian	52	7
eu	Basque	19	2
fa	Persian	53	12
fi	Finnish	154	88
fr	French	747	1084
gl	Galician	27	3
he	Hebrew	122	83
hi	Hindi	22	2
hr	Croatian	43	10
ht	Haitian	11	0
hu	Hungarian	82	38
id	Indonesian	95	31
io	Ido	4	0
is	Icelandic	23	3
it	Italian	384	505
ja	Japanese	442	552
jv	Javanese	8	1
ka	Georgian	25	8
ko	Korean	72	26
ku	Kurdish	11	0
la	Latin	21	1
lb	Luxembourgish	18	1
lt	Lithuanian	57	12
lv	Latvian	19	5
mk	Macedonian	16	5
mr	Marathi	22	1
ms	Malay (macrolanguage)	35	9
nap	Neapolitan	13	0
nds	Low German	9	1
new	Newari	33	4
nl	Dutch	330	419
nn	Norwegian Nynorsk	37	9
no	Norwegian	156	80
oc	Occitan (post 1500)	15	1
pl	Polish	335	340
pms	Piemontese	11	0
pt	Portuguese	413	410
ro	Romanian	92	63
ru	Russian	376	437
scn	Sicilian	23	0
sh	Serbo-Croatian	21	9
sk	Slovak	61	17
sl	Slovenian	52	7
sq	Albanian	18	0
su	Sundanese	11	0
sv	Swedish	220	136
ta	Tamil	11	5
te	Telugu	27	6
th	Thai	50	21
tl	Tagalog	11	0
tr	Turkish	111	34
uk	Ukrainian	106	41
vi	Vietnamese	54	16
wa	Walloon	13	0
zh	Chinese	181	125

Table 1: Composition of primary (1°) and secondary (2°) documents in the dataset for each language (based on ISO-639 language codes).

leaving us with data in 74 languages. All of the data is UTF-8 encoded, and the total volume of uncompressed data is almost 60GB.

For this task, we were interested in presenting a language identification challenge over largely bilingual documents. We assumed that Wikipedia documents were all monolingual, and that the language they were written in corresponded exactly to the Wikipedia they were located in. On the basis of these assumptions, we set out to build bilingual documents by combining portions of monolingual documents. Each document in our dataset is compiled from two source documents, which we will refer to as "primary" and "secondary". In addition to making our documents bilingual, we were interested in maintaining semantic linkage between the sections of the document in different languages. We did this by taking advantage of the fact that many Wikipedia documents contain links to a comparable document in another language. For example, the English Wikipedia document on *Natural language processing* contains a link to the equivalent document in a variety of languages, including the Italian *Elaborazione del linguaggio naturale* and French *Traitement automatique du langage naturel*. The links are of the form `[[<language-prefix>:<page title>]]`, and thus can easily be parsed with a regular expression. For purposes of elaboration, we shall refer to this kind of link as a language-link. It is important to note that the language-linked documents are not translations, they are comparable documents, on the same topic in different languages.

To construct each bilingual document, we first selected the language of the primary document via a roulette-wheel approach, weighted according to the relative distribution of the number of pages

for each language Wikipedia. From there, we randomly sampled a document (without replacement) from the primary language Wikipedia. We then selected a secondary document from the set of language-links in the primary document via the same routlette-wheel approach, again weighted by the global distribution of the languages present.

To each source document, we applied simple regular expression-based normalisation to remove redirects, language links and templates. We also replaced intra-wiki links with the anchor text of the link. We then chunked each of the two source documents into paragraphs by splitting on two consecutive newline characters. We select the first half of the paragraphs from the primary document and the second half of the paragraphs from the secondary document (rounding up in each case), and concatenate them together to form a single document. For example, if the primary document contained 5 paragraphs and the secondary contained 8 paragraphs, we would select the first 3 paragraphs from the primary document, and the last 4 paragraphs from the secondary document. If either of these sections falls below 1000 bytes, we reject this primary–secondary pair and start over.

3 Dataset Characteristics

The dataset contains 10000 documents, separated into three partitions: 8000 for training, 1000 for development and 1000 for test. All except three of the documents are multilingual. These three documents are caused by anomalies in the Wikipedia data, in that the primary document contained a language-link to a document in the same language; in two of these cases, the primary document contained the same content under different identifiers. As a result, the same secondary document was selected for both, resulting in two documents with identical content in the final dataset.

The language distributions of the primary and secondary document components are as detailed in Table 1.

In addition to the raw documents and language annotations, we have also made available an evaluation script. The full dataset is available from `http://www.csse.unimelb.edu.au/research/lt/resources/altw2010-langid/`.

Baseline	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu$	$\mathcal{R}_\mu$	$\mathcal{F}_\mu$
en	.011	.015	.012	**.701**	.350	**.467**
en+de	**.014**	**.030**	**.018**	.458	**.458**	.458

Table 2: Results for the different baseline strategies over the development documents

4 Baseline Results

As each document has two languages associated with it, three different baselines can be considered:

best-1 monolingual: the single most common language

best-2 monolingual: the two most common languages

best-1 multilingual: the most common language pair

The results for the different strategies are presented in Table 2, as trained over the training documents and evaluated over the development documents. In our case, the two most common languages are `en` followed by `de`, and it also happens that the most common language pair is `en-de`. As such, our latter two baselines are identical in behaviour, and are presented together in the final row of the table. Based on the evaluation scripts made available as part of the dataset, we evaluate the models using micro-averaged precision ($\mathcal{P}_\mu$), recall ($\mathcal{R}_\mu$) and F-score ($\mathcal{F}_\mu$), as well as macro-averaged precision ($\mathcal{P}_M$), recall ($\mathcal{R}_M$) and F-score ($\mathcal{F}_M$). The micro-averaged scores indicate the average performance *per document*, while the macro-averaged scores indicate the average performance *per language*.

5 Benchmark Results

To provide a minimal benchmark, we consider a prototype-based classifier based on skew divergence, with the usual mixing parameter $\alpha = 0.99$, based on the findings of Baldwin and Lui (to appear). The prototype is calculated as the arithmetic mean across all instances for each feature. We deal with the multiple-language labelling using three different methods:

single: a single prototype is learned for each language; any document containing the language is used in the calculation of this prototype.

Tokenisation	Multiclass	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu$	$\mathcal{R}_\mu$	$\mathcal{F}_\mu$
unigram	single	.440	.274	.295	.264	.132	.176
bigram	single	.540	.376	.413	.583	.291	.389
trigram	single	**.564**	.412	.453	.814	.407	.543
unigram	stratified	.412	.458	.414	.629	.622	.625
bigram	stratified	.460	.448	.435	.775	.768	.771
trigram	stratified	.497	.467	**.464**	**.833**	.826	**.829**
unigram	binarised	.115	**.786**	.155	.057	.878	.107
bigram	binarised	.171	.705	.221	.114	.885	.202
trigram	binarised	.227	.686	.292	.259	**.903**	.402

Table 3: Results for the benchmark methods over the development documents, for a nearest prototype learner in combination with different tokenisation and multiclass handling strategies

stratified: a single prototype is learned for each language pair.

binarised: a pair of prototypes is learned for each language, one from documents containing the language, and the other from documents that do not contain the language; a classification for each language is produced via this binarisation.

We combine these three strategies with three tokenisation strategies, based on byte unigrams, bigrams or trigrams.

We present results over the development documents in Table 3. In the shared task, the primary evaluation measure was micro-averaged F-score ($\mathcal{F}_\mu$), on the basis of which the best-performing benchmark method is nearest prototype with skew divergence on the basis of byte trigram tokenisation, and with stratified multiclass handling.

6 Conclusion

This paper has described a multilingual language identification dataset, as used in the 2010 Australasian Language Technology Workshop shared task. We outlined the methodology for constructing the dataset from Wikipedias for different languages, and detailed results for a series of baseline and benchmark methods.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. This research was supported in part by a Google Research Award.

References

Beatrice Alex, Amit Dubey, and Frank Keller. 2007. Using foreign inclusion detection to improve parsing performance. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2007 (EMNLP-CoNLL 2007)*, pages 151–160, Prague, Czech Republic.

Timothy Baldwin and Marco Lui. to appear. Language identification: The long and the short of the matter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, Los Angeles, USA.

Baden Hughes, Timothy Baldwin, Steven Bird, Jeremy Nicholson, and Andrew MacKinlay. 2006. Reconsidering language identification for written language resources. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 485–488, Genoa, Italy.

Fei Xia and William Lewis. 2009. Applying NLP technologies to the collection and enrichment of language data on the web to aid linguistic research. In *Proceedings of the EACL 2009 Workshop on Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education (LaTeCH – SHELT&R 2009)*, pages 51–59, Athens, Greece.

Fei Xia, William Lewis, and Hoifung Poon. 2009. Language ID in the context of harvesting language data off the web. In *Proceedings of the 12th Conference of the EACL (EACL 2009)*, pages 870–878, Athens, Greece.

Peer-reviewed papers

Variability and Consistency in the Idiosyncratic Selection of Fillers in Japanese Monologues: Gender Differences

Shunichi Ishihara
The Australian National University
Canberra, Australia
shunichi.ishihara@anu.edu.au

Abstract

This study is a linguistic study on idiosyncrasy using speaker classification technique as an analytical tool. The goals of this study are to find out 1) to what extent Japanese filler words (e.g. *um*, *you know* in English) carry individual idiosyncratic information; 2) if there are any differences in the degree/nature of idiosyncrasy between the sexes; and 3) what contributes to the identified gender differences, if there are any. Based purely on the individual selection of fillers, we report in this study that 1) speaker discrimination performance was better in the male (ca. 85% accuracy) than the female (ca. 75% accuracy) speakers by approximately 10%, and 2) the poorer performance of the female speakers was due to the larger within–speaker differences in the female speakers than the male speakers. That is, the selection of fillers by female speakers is more variable, speech by speech, than that by male speakers, even under similar conditions (e.g. same type of audience and the same degree of formality). We also discuss that the findings of the current study agree with the previously–reported differences between the sexes in language use.

1 Introduction

We intuitively know that different people talk/write differently, even when they try to convey the same message. We also know that people tend to use their individually selected preferred words despite the fact that in principle they can use any word at any time from the vocabulary built up over the course of their lives. This is due to the idiosyncratic choice of words, expressions and so forth. Every speaker of a given language has their own distinctive and individual version of the language—which is often referred to as *idiolect* (Halliday et al., 1964; Coulthard and Johnson, 2007). This idiolect manifests itself in various aspects of communication, such as the choice of words, expressions, or even grammar, morphology, semantics and discourse structure. The idiosyncratic nature of word selection between speakers/writers has been studied in different fields. For example, it has been used to understand speaking styles of political leaders (Slatcher et al., 2004), to identify the authors of literary works (Thisted and Efron, 1987), to detect plagiarism (Woolls, 2003) and to enhance the performance of automatic speaker recognition (Doddington, 2001). In the domain of text (in contrast to speech), it has been demonstrated that word category usage is very stable across time and writing topics (Pennebaker and King, 1999).

Besides the idiosyncrasies of individual speakers, men and women speak/write differently (Koppel et al., 2002). This has been well reported in various linguistic and non–linguistic aspects of speech (Lakoff, 1975; Coats, 1993). Particularly in terms of linguistic styles, it has been argued that women tend to be more stylistically flexible and varied than men in language use (Holmes, 1997; Holmes, 1998; Chambers, 1992). Thus, the current study investigates:

- 'to what extent we are idiosyncratic' in selecting certain words rather than others, keeping in mind that there may be some differences in the degree/nature of idiosyncrasy between the sexes; and
- if there are any differences between the sexes, 'what contributes to the identified gender differences' in these instances.

This study focuses on the use/selection of fillers in Japanese as several existing studies identify

Shunichi Ishihara. 2010. Variability and Consistency in the Idiosyncratic Selection of Fillers in Japanese Monologues: Gender Differences. In *Proceedings of Australasian Language Technology Association Workshop*, pages 9−17

subjectively (Furui et al., 2002; Sato, 2002; Yamane, 2002) and empirically (Ishihara, 2009) that preference of fillers exists across speakers. Fillers are unique to spoken language. They are a sound or a word (e.g. *um*, *you know*, *like*) which is uttered by a speaker to signal that he/she is thinking or hesitating. It is reported that 6% of the total number of words spoken in Japanese are fillers (NIJL, retrieved 2008). It is also reported that speakers' attributes, such as age and gender, affect the choices of fillers in Japanese and English (NIJL, retrieved 2008; Watanabe et al., 2006). Studies on speech corpora show that males tend to use fillers more frequently than females in English and Japanese (Shriberg, 1994; NIJL, retrieved 2008).

In order to answer the above research questions, we will conduct a series of speaker discrimination tests—separately between male and female speakers—based solely on fillers. The hypothesis is that the more consistent the individual speaker's selection of fillers is, and the more significantly fillers selected by one speaker differ from those selected by another, the more accurately speaker discrimination can be performed.

We demonstrate first of all that fillers bear the idiosyncratic information of speakers to the extent that the accuracy of the speaker discrimination based solely on fillers can be as high as approximately 85% for male speakers and approximately 75% for female speakers. As can be seen in this difference in accuracy between the sexes, we also report that the speaker discrimination performance is better in the male than the female speakers by approximately 10%. Four reasons can be speculated: 1) the idiosyncrasy was not well modelled for the females due to the fact that less female speakers were used in the current study; 2) the between–speaker difference is larger in the female speakers; 3) the within–speaker difference is larger in the female speakers or 4) any combination of the above three. Further investigation of the data revealed that the poorer performance of the female speaker discrimination compared to the male speaker discrimination is due to the tendency of the female speakers to have larger within–speaker differences than the male speakers. That is, the selection of fillers is more variable or less consistent across the non–contemporaneous speeches of the same speakers for female than male speakers even under very similar conditions.

2 Methodology

Two kinds of comparisons are involved in speaker discrimination tests. One is called *Same Speaker Comparison* (SS comparison) where two speech samples produced by the same speaker need to be correctly identified as the same speaker. The other is, *mutatis mutandis*, *Different Speaker Comparison* (DS comparison).

The series of speaker discrimination tests that we conducted can be categorised into two experiments: Experiments 1 and 2. Detailed procedures of Experiments 1 and 2 are explained in §4 and §5, respectively.

2.1 Database and Speakers

For speech data, we used the Corpus of Spontaneous Japanese (CSJ) (Maekawa et al., 2000), which contains recordings of various speaking styles such as sentence reading, monologue, and conversation. For this study we used only the monologues, categorised as Academic Presentation Speech (APS) or Simulated Public Speech (SPS). APS was mainly recorded live at academic presentations, most of which were 12-25 minutes long. For SPS, 10-12 minute mock speeches on everyday topics were recorded. We selected our speakers from this corpus based on three criteria: availability of multiple and non–contemporaneous recordings, spontaneity (e.g. not reading) of the speech, and speaking in standard modern Japanese. Spontaneity and standardness of the language were assessed on the basis of the rating the CSJ provides. Thus, only those speech samples which are high in spontaneity and uttered entirely in Standard Japanese were selected for this study. This gives us 416 speech samples (= 208 speakers: 132 male and 76 female speakers x 2 sessions).

2.2 Fillers

In CSJ, a filler tag is assigned to one of the pre–selected words given in Table 1 which have the function of 'filling up gaps in utterances'. Some of the words given in Table 1 can also be used as lexical words. If it is uncertain as to whether a given word is used as a lexical word or a filler, additional information is embedded in the tag indicating this uncertainty. In such cases, the word was removed from speaker discrimination tests.

In the selected speech samples, we observed 44

- a(-), i(-), u(-), e(-), o(-), n(-), to(-)†, ma(-)†
- u(-)n, a(-)(n)no(-)†, so(-)(n)no(-)†
- u(-)n(-)(t)to(-)†, a(-)(t)to(-)†, e(-)(t)to(-)†, n(-)(t)to(-)†
- one of the above + {∼desune(-), ∼ssune(-)}
- one of the above with † + {ne(-), sa(-)}

Table 1: Pre–selected fillers in CSJ. "-" stands for the prolongation of the preceding segment.

different filler words for the male speakers and 42 for the female speakers, which are listed in Table 2. As the previous studies which report that more fillers appear in formal speech than informal speech predict (Watanabe, 2009; Nishimura et al., 2010), a large number of fillers could be identified in the selected speech samples as the setting was relatively formal.

Although there are some ranking differences between the sexes in terms of frequency, it can be seen from Table 2 that very similar fillers are used across the sexes.

In order for the choice of filler words to be useful as a speaker classifier, it has to satisfy two criteria. First it has to be consistent within a speaker. The second criterion is the relative frequency of use of fillers compared to other speakers. In order to capture these characteristics, we have to model each speech in terms of the use of fillers, and it needs to be compared against another. We describe our method below.

2.3 Vector space model

Using the frequency counts of the identified fillers, each speech is modelled as a real–valued vector in this study. If n different fillers are used to represent a given speech S, the dimensionality of the vector is n. That is, S is represented as a vector of n dimensions ($\overrightarrow{S} = (F_1, F_2 \dots F_n)$, where F_i represents the i^{th} component of $\overrightarrow{S}$ and F_i is the frequency of the i^{th} filler). For example, if 5 fillers are used to represent a speech (X), and the frequency counts of these fillers are 3, 10, 4, 18 and 1 respectively, the speech X is represented as $\overrightarrow{X} = (3, 10, 4, 18, 1)$.

2.4 Term frequency inverse document frequency weighting

The usefulness of particular words is determined by their uniqueness as well as how frequently they are used. Different weights were given to different filler words depending on their uniqueness in the pooled data. The *tf* · *idf* (term frequency inverse document frequency) weight (Formula 1) is used to evaluate how unique a given filler word is in the population, and a weight is given to that filler to reflect its importance to the speaker discrimination (Manning and Schütze, 2001).

$$W_{i,j} = tf_{i,j} * log(\frac{N}{df_i}) \quad (1)$$

In Formula 1, term frequency ($tf_{i,j}$) is the number of occurrences of word *i* (W_i) in the document (or speech sample) *j* (d_j). Document frequency (df_i) is the number of documents (or speech samples) in the collection in which that word *i* (W_i) occurs. *N* is the total number of documents (or speech samples).

2.5 Cosine similarity measure

The difference between two speech samples, which are represented as vectors ($\overrightarrow{x}$, $\overrightarrow{y}$), is calculated based on the cosine similarity measure (Formula 2) (Manning and Schütze, 2001). This particular method was selected as the durations of the speech samples are all different, assuming that the direction of a vector should be constant if the speech sample is long enough.

$$\begin{aligned} diff(\overrightarrow{x}, \overrightarrow{y}) = cos(\overrightarrow{x}, \overrightarrow{y}) &= \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|} \\ &= \frac{\sum_{i=1}^{n} x_i * y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} * \sqrt{\sum_{i=1}^{n} y_i^2}} \end{aligned} \quad (2)$$

The range of the difference in two vectors (*diff*($\overrightarrow{x}$, $\overrightarrow{y}$)) is between 1.0 (=$cos(0°)$) for two vectors pointing in the same direction and 0.0 (=$cos(90°)$) for two orthogonal vectors.

Please note that in the experiments of this study (§4 and §5), the length of the vectors were standardised by only looking at the X most frequent fillers ($X = (5, 10, 15, 20, 25, 30, 35, 40)$) as cosine similarity measure requires vectors of equal length.

3 Method for Speaker Discrimination

In this study, the performance of speaker discrimination is assessed on the basis of the probability distribution functions (PDFs) of the difference for two contrastive hypotheses. One is the hypothesis that two speech samples were uttered by the same speaker (the same speaker (SS) hypothesis)

Male: (52588 fillers)						Female: (20575 fillers)					
F	%	F	%	F	%	F	%	F	%	F	%
e-	31.71	u-	1.08	nto	0.03	e-	24.24	e-to-	0.83	to-	0.15
ma	10.84	n-	1.00	to-	0.03	ano-	15.20	u	0.71	eto-	0.13
e	10.62	o	0.83	nto-	0.02	ano	12.86	a-no	0.67	n-to	0.07
ma-	7.85	etto	0.74	u-n	0.01	e	12.07	etto-	0.61	u-n	0.05
ano	6.41	i-	0.59	a-to	0.01	ma	8.27	un	0.45	nto	0.02
ano-	6.19	e-to-	0.57	n-to-	0.008	ma-	5.45	a-no-	0.42	n-to-	0.01
sono	3.31	i	0.35	nto-	0.008	sono	2.74	to	0.35	u-nto	0.005
e-to	3.17	eto	0.30	a-to-	0.006	n	2.45	e-tto-	0.32	nto-	0.005
a	2.52	etto-	0.24	ntto	0.002	a	2.37	o	0.29	ntto-	0.005
o-	2.09	to	0.22	n-tto-	0.002	e-to	2.17	eto	0.28	so-no-	0.005
n	2.07	e-tto-	0.21	u-nto	0.002	sono-	1.29	u-	0.28	unto	0.005
a-	2.01	a-no-	0.21	a-tto	0.002	e-tto	1.22	o-	0.28	so-no	0.005
e-tto	1.59	a-no	0.19	e-ttodesune	0.002	n-	1.13	i	0.25		
sono-	1.49	un	0.09	so-no-	0.002	a-	1.02	i-	0.16		
u	1.20	eto-	0.05			etto	0.88	at	0.15		

Table 2: Fillers and their frequencies (%) of occurrence given separately for the different sexes. F = fillers. "-" stands for the prolongation of the preceding vowel.

and the other is that two speech samples were uttered by different speakers (the different speaker (DS) hypothesis). These probabilities can be formulated as $P(E|H_{ss})$ and $P(E|H_{ds})$ respectively, where E is the difference, H_{ss} is the SS hypothesis and H_{ds} is the DS hypothesis. In this study, the PDF of the difference assuming the SS hypothesis is true is called the SS PDF (PDF_{ss}) and assuming the DS hypothesis is true the DS PDF (PDF_{ds}). Please note again that in this study, the difference of two speech samples refers to the cosine difference between the two vectors representing the two speech samples.

Each PDF was modelled using the kernel density function (`KernSmooth` library of R statistical package). Examples of PDF_{ss} and PDF_{ds}, which are based on all of the male speakers with the dimensions of 40, are given in Figure 1. As can be seen from Figure 1, those PDF_{ss} and PDF_{ds} do not conform to a normal distribution. This is the motivation of the use of the kernel density function in this study.

As can be seen from Figure 1, PDF_{ss} and PDF_{ds} were not always monotonic, resulting in more than a single crossing point, particular when the dimension of a vector is less than 5. Thus, the performance of the system with the length of a vector being less than 5 is not given.

These two PDFs also show the accuracy of this particular speaker discrimination system. If the crossing point (θ) of the PDF_{ss} and the PDF_{ds} is set as the threshold, we can estimate the performance of this particular speaker discrimination system from these PDFs. Area 1 in Figure 1—the area surrounded by the grey line (PDF_{ss}), the vertical dotted line of $x = \theta$ and the line of $y = 0$—is the predicted error for the SS comparisons, and Area 2 of Figure 1—the area which is surrounded

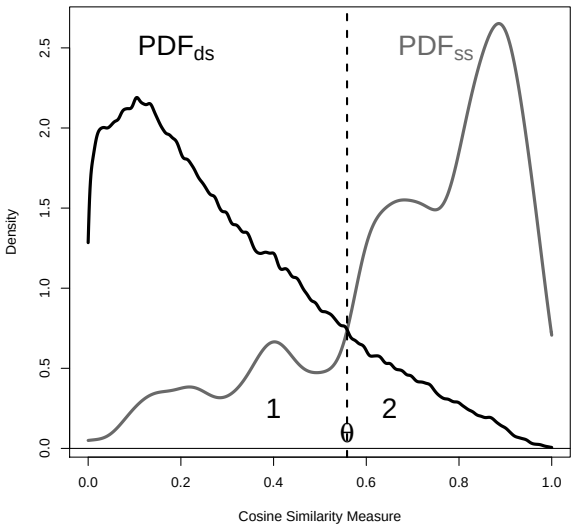


Figure 1: Examples of PDF_{ss} (the grey curve) and PDF_{ds} (the black curve). The vertical dashed line ($x = \theta$) is the crossing point of PDF_{ss} and PDF_{ds}.

by the black line (PDF_{ds}), the vertical dotted line of $x = \theta$ and the line of $y = 0$—is the predicted error for the DS comparisons. Therefore, the accuracy of the SS ($ACCURACY_{ss}$) and DS comparisons ($ACCURACY_{ds}$) can be calculated by Formulae 3 and 4, respectively.

$$ACCURACY_{ss} = \left(1 - \int_0^{\theta} PDF_{ss}(x)dx\right) * 100 \quad (3)$$

$$ACCURACY_{ds} = \left(1 - \int_{\theta}^{1} PDF_{ds}(x)dx\right) * 100 \quad (4)$$

The accuracy of a speaker classification system (both in SS and DS comparisons) was estimated in this study.

4 Experiment 1: Discrimination Performance and its Difference between the Sexes

In Experiment 1, a series of speaker discrimination tests were conducted separately for the male and the female speakers (132 male and 76 female speakers). Out of the 264 speech samples of the 132 male speakers, 132 SS comparisons and 34584 DS comparisons are possible. Likewise, for the female speakers, 76 SS comparisons and 11400 DS comparisons are possible.

The performance of a speaker classification system is assessed separately for the male and the female speakers as explained in §3, with different numbers of the dimensions of a spacial vector. The spacial vectors of 5, 10, 15, 20, 25, 30, 35 and 40 dimensions are used in Experiment 1. That is, for example, the spacial vector of 5 dimensions means that the frequency counts of the 5 most frequently used fillers are used to represent a speech sample. In Figure 2, the accuracy of a speaker classification system is plotted separately for the male (solid lines) and the female speakers (dotted lines) as a function of the different numbers of dimensions (= fillers). The grey and black lines represent the SS and the DS comparisons, respectively, in Figure 2.

Despite the fact that the techniques used in the speaker discrimination tests are standard and fairly simple, the performance of speaker discrimination is fairly good, particularly for the male speaker discrimination of which accuracy is as good as approximately 85%. It can be observed from Figure 2 that when 20 or more fillers are included in the vectors, 1) the performance of the SS and the DS comparisons becomes stable; 2) the speaker discrimination of the male speakers outperforms that of the female speakers by approximately 10% and 3) the performance of the SS and the DS comparisons becomes comparable. A trade–off between the performance in the SS comparison and that in the DS comparisons is evident if less than 20 fillers are used. The third point above is important that the comparable performance between the SS and the DS comparisons means that the result is well calibrated.

The fact that speaker discrimination performance peaks with half of the dimensions available is not surprising. The feature vectors were based on the frequencies of occurrence of a given filler word, and we first picked ones with higher frequency to be included in the feature. So vectors in the later orders have very low frequencies, such as 0. This means that the latter part of longer vectors tends to include very similar low numbers across speakers, not contributing as a strong unique feature of speakers.

In Experiment 2, we will look into what contributes to the difference in performance between the male and the female speakers.

5 Experiment 2: Why is Female Discrimination Worse?

In Experiment 1, it was demonstrated that the male speaker discrimination outperforms the female speaker discrimination by approximately 10%. Four reasons for this seem to be possible. The first possible reason (R1) is a simple technical and statistical reason. As the number of female speech samples (152 = 76 speakers x 2 sessions) is less than that of male speech samples (264 = 132 speakers x 2 sessions), the idiosyncratic use of fillers was not modelled as well for the female as for the male speakers, resulting in a poor performance for the female speaker discrimination. The second possible reason (R2) is that the between–speaker differences are smaller and less significant in the female than the male speakers. That is, the female speakers behave more uniformly than the male speakers, making the speaker discrimination of the female speakers more difficult. The third possible reason (R3) is that the within–speaker difference is larger in the female than the male speakers. That is, the female speakers are less consistent with their idiosyncrasy in selecting fillers than the male speakers, giving rise to the poorer perfor-

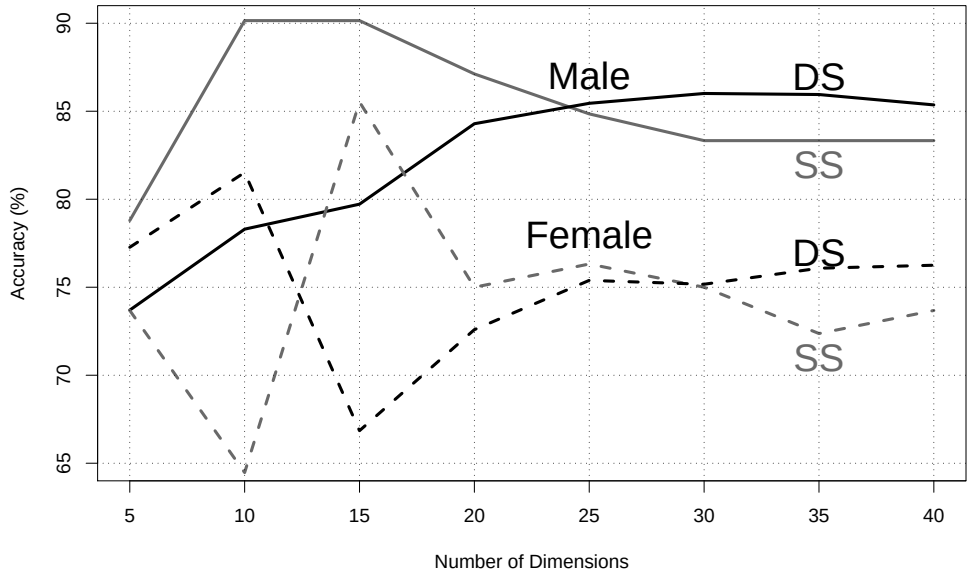


Figure 2: Speaker discrimination performance. The solid lines denote male speakers and the dashed lines denote female speakers. The black lines indicate DS comparisons and the grey lines indicate SS comparisons.

mance of the female speaker discrimination. The fourth possible reason (R4) includes any combination of the above three.

As a first step for identifying which is the true picture contributing to the difference in speaker discrimination performance between the male and female speakers, we need to conduct speaker discrimination tests under the same conditions for both the male and female speakers, equalising the number of speakers. Therefore, 3 male speaker groups, each of which consisted of 76 speakers, were randomly created from the 132 male speakers. Three different speaker discrimination tests were conducted separately using these 3 male speaker groups. If the speaker discrimination performance of these 3 groups of 76 male speakers is similar to that of the 132 male speakers obtained in Experiment 1, we can eliminate R1. In Experiment 2, the spatial vectors of 40 dimensions were used for the speaker discrimination tests. The results of these speaker discrimination tests are summarised in Table 3, together with those of the previous tests.

As can be seen from Table 3, the performance of the male speaker discrimination remains as accurate with only 76 male speakers as with 132 male speakers, the male speaker discrimination outperforming the female speaker discrimination. This result indicates that R1 can be eliminated as a possible reason.

In order to examine the validity of R2∼R4, the

Sex	Male				
n	76–1	76–2	76–3	**76–Ave.**	**132**
SS	88.2	89.5	77.6	**85.1**	**83.3**
DS	84.7	82.1	89.0	**85.3**	**85.3**

Sex	Female
n	**76**
SS	**73.7**
DS	**76.2**

Table 3: Comparison of speaker discrimination accuracies (%) under the same conditions for the male and the female speakers. n = number of speakers. The discrimination accuracies when 132 male speakers are pooled together are given as references. The numerals in bold are the values of most concern for the sex comparisons.

differences of paired speech samples that were calculated for the SS and the DS comparisons were scrutinised for the male and the female speakers. Table 4 contains the average differences of pairs of speech samples for the SS and the DS comparisons, which were calculated separately for the male and the female speakers. Let us remind the reader that the value of the cosine similarity measure becomes smaller if the difference of two speech samples is larger. It can be seen in Table 4 that for the DS comparisons, the male (0.29) and the female speakers (0.31) show very similar values, while for the SS comparisons, the average difference of compared speech samples is larger for

the female (0.62) than the male speakers (0.73).

Sex	Male			
n	76–1	76–2	76–3	**76–Ave.**
SS	0.75	0.70	0.75	**0.73**
Skew	-0.97	-0.65	-1.14	**-0.92**
DS	0.29	0.27	0.30	**0.29**
Skew	0.83	0.87	0.83	**0.84**

Sex	Female
n	**76**
SS	**0.62**
Skew	**-0.53**
DS	**0.31**
Skew	**0.81**

Table 4: The average differences of pairs of speech samples for SS and DS comparisons in cosine similarity measure and the degree of skewness for each PDF. n = number of speakers. The numerals in bold are the values of most concern for the sex comparisons.

The above gender difference, i.e. that the female speakers have greater differences than the male speakers for the SS comparisons, can also be seen from the different patterns observed between the PDF_{ss} of the female speakers and those of the male speakers. Figure 3 contains the PDF_{ss} and the PDF_{ds} plotted for the female speaker discrimination test (solid line) and those plotted for the 3 male speaker discrimination tests (dotted lines) conducted in Experiment 2.

Figure 3–1 shows that the PDF_{ds} of the female speakers (solid line) is very similar to those of the male speakers (dotted line)—with the PDF_{ds} of the male speakers being slightly more positively skewed (the average male skew: 0.84; the female skew: 0.81). On the other hand, as for the PDF_{ss} (Figure 3–2), the male speakers (dotted lines) show more negative skewness than the female speakers (solid line) (the average male skew: –0.92; the female skew: –0.53). Statistically speaking as well, three sets of two–sided two–sample Kolmogorov–Smirnov tests (Male 1 vs. Female; Male 2 vs. Female and Male 3 vs. Female) confirm that the distributional pattern is different between the male and female speakers in their PDF_{ss} ($p \leqq 0.04432$).

Thus, it can be concluded that the larger within–speaker difference in the female speakers than the male speakers—which is R3—contributed to the poorer performance in the female speaker discrimination than the male.

6 Discussion

We have demonstrated in Experiment 1 that Japanese fillers carry individual information to the extend that we can discriminate speakers with approximately 85% and 75% accuracy for the male and the female speakers, respectively. 75%∼85% accuracy is not too bad, but not so great as a speaker discrimination task. However, we would like to remind the reader that the techniques we employed are very simple.

In Experiment 2, it has been demonstrated that the male and the female speakers exhibit a very similar pattern for DS comparisons, whilst for SS comparisons they are different in that the female speakers generally have greater differences than the male speakers for compared speech samples. This indicates that the within–speaker difference is larger for the female speakers than the male speakers. In other words, the selection of fillers is more flexible and variable in the female speakers, even under fairly controlled and similar situations, whereas male speakers tend to be more consistent with their selection of fillers.

However, it is not clear at this stage if this is a general tendency observed in many languages or unique to Japanese. Furthermore, we do not know why female speakers are more variable in selecting fillers across non–contemporaneous occasions in comparison to male speakers.

Judging from what has been researched on gender differences in languages, the flexibility/variability of women's speech appears to be one of the universals (Holmes, 1998). Holmes (1997, p. 198) remarks that "women tend to use a wider range of linguistic variants than men, and that their usage varies according to identical contextual factors". A very similar statement can be found in Chambers (1992, p. 199) that "... they [women] command a wider range of linguistic variants ... they have the linguistic flexibility to alter their speech as social circumstances warrant." The flexibility/variability of women's speech in linguistic styles has been empirically supported by various studies (Nichols, 1983; Ide, 1982; Escure, 1991). In speech perception as well, it has been reported that females are more sensitive to variations in speech styles (Wiley and Eskilson, 1985).

Thus, the result of the current study, which demonstrated the differences between males and

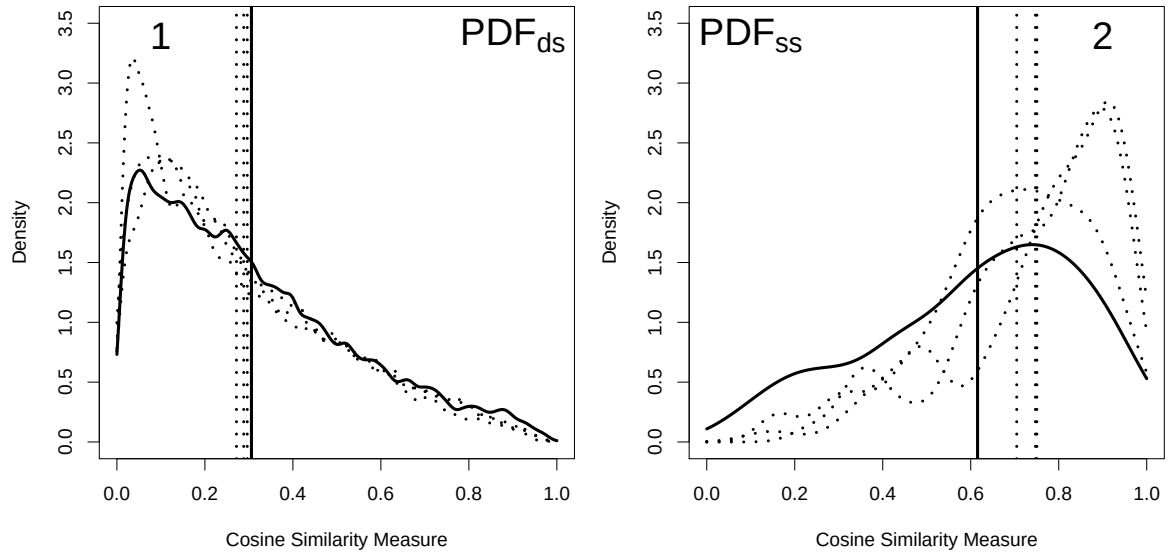


Figure 3: Differences between male and female in PDFs. Panels 1 and 2 are for PDF_{ds} and PDF_{ss}, respectively. The vertical lines are average cosine similarity values. The solid lines are used for the female speakers and the dotted lines are for the three groups of the male speakers.

females in their linguistic styles, contributes further evidence supporting the above assertion, namely that women are more variable and flexible in their linguistic usages. Yet, from the findings of this paper, it is difficult to explain linguistically why females behave in this way.

7 Conclusions

In this study, we have first demonstrated that Japanese fillers carry the idiosyncratic information of speakers. We have shown that the speaker discrimination performance is more effective in the male than the female speakers by approximately 10%. We also have demonstrated that the poorer performance of the female speaker discrimination compared to the male speaker discrimination is due to the tendency for female speakers to have larger within–speaker differences than male speakers. That is, the selection of fillers is more variable, or less consistent, for female than male speakers even under very similar conditions. We have also discussed that the result of the current study conforms to the previously reported differences between males and females in their speech; that women's linguistic use of their language is more variable and flexible in comparison to males.

8 Future Research

Japanese data was used for this study. Thus it is interesting to see if we can recognise the same sex–difference in other languages.

This study is part of a large study on forensic voice comparison (FVC). In FVC, the strength of evidence (or likelihood ratio) is equally important to the discriminability of the system. Therefore, it is interesting to see what sort of strength of evidence can be obtained from the idiosyncratic selection of fillers. Furthermore, FVC usually uses acoustic features, such as Mel–frequency cepstrum coefficients, formant–patterns, fundamental frequency (f0) and so on. The feature used in the current study is a non–acoustic feature (or a text–based feature) which is completely independent from the acoustic features. It thus has potential to make a significant contribution to improving the accuracy of speaker classification systems (Shriberg and Stolcke, 2008) by combining the non–acoustic feature of the current study and usual acoustic features. As a next step, therefore, we intend to extend this study by combining this feature with other, more conventional speaker classification features, such as formants or f0 related features.

Acknowledgements

This study was financially supported by the International Association for Forensic Phonetics and Acoustics and the College of Asia and the Pacific, the Australian National University. The author thanks anonymous reviewers for their valuable comments.

References

M. A. K. Halliday, A. McIntosh and P. Strevens. 1964. *The Linguistic Sciences and Language Teaching*. London: Longman.

M. Coulthard and A. Johnson. 2007. *An Introduction to Forensic Linguistics: Language in Evidence*. London, New York: Routledge.

R. Slatcher, C. Chunga, J. Pennebakera and L. Stone. 2004. Winning words: Individual differences in linguistic style among U.S. presidential and vice presidential candidates. *Journal of Research in Personality*, 41(1):63–75.

R. Thisted and B. Efron. 1987. Did Shakespeare write a newly–discovered poem? *Biometrika*, 74(3):445–455.

D. Woolls. 2003. Better tools for the trade and how to use them. *Forensic Linguistics. The International Journal of Speech, Language and the Law*, 10(1):102–112.

G. Doddington. 2001. Speaker recognition based on idiolectal differences between speakers, *Proceedings of the Eurospeech*, Aalborg, Denmark, September 2001.

J. W. Pennebaker and L. A. King. 1999. Linguistics styles: Language use as an individual difference. *Journal of Personality and Social Psychology*, 77(6):1296–1312.

M. Koppel, S. Argamon and A. R. Shimoni. 2002. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17(4):401–412.

R. Lakoff. 1975. *Language and Women's Place*. New York: Harper and Row.

J. Coats. 1993. *Women, Men and Language*. New York: Longman.

J. Holmes. 1997. Women, language and identity. *Journal of Sociolinguistics*, 1/2:195–223.

J. Holmes. 1998. Women's talk: The question of sociolinguistic universals. In, J. Coates (ed.) *Language and Gender: A Reader*. Oxford, 461–483.

J. K. Chambers. 1992. Linguistic correlates of gender and sex. *English World–Wide*, 13:173–218.

S. Furui, K. Maekawa and H. Isahara. 2002. Intermediate results and perspectives of the project 'Spontaneous Speech: Corpus and Processing Technology', *Proceedings of the 2nd Spontaneous Speech Science and Technology Workshop*, 1–6.

Y. Sato. 2002. *'UN' and 'SO' in Japanese Casual Conversation between Native Speakers: The Use of Fillers*. MA thesis, Nagoya University.

C. Yamane. 2002. *Fillers in Japanese Conversation*. Tokyo: Kuroshio publisher.

S. Ishihara. 2009. How diverse are we? An empirical analysis of individual differences in the use of fillers, *The 11th International Pragmatics Conference*, Unpublished paper.

The National Institute of Japanese Language. Homepage of the Corpus of Spontaneous Japanese. http://www.kokken.go.jp/katsudo/seika/corpus/public/6_1.html, retrieved in August, 2008.

M. Watanabe, Y. Den, K. Hirose, S. Miwa and N. Minematsu. 2006. Factors affecting speakers' choice of fillers in Japanese presentations, *Proceedings of Interspeech 2006*, 1256–1259.

E. E. Shriberg. 1994. *Preliminaries to a Theory of Speech Disfluencies*, PhD thesis, University of California, Berkeley.

K. Maekawa, H. Koiso, S. Furui, and H. Isahara. 2000. Spontaneous speech corpus of Japanese, *The Second International Conference of Language Resources and Evaluation (LREC2000)*, Athens, 2000, 947–952.

M. Watanabe. 2009. *Features and Roles of Filled Pauses in Speech Communication, A Corpus–Based Study of Spontaneous Speech*. Tokyo: Hitsuji Shobo Publishing.

R. Nishimura, N. Kitaoka and S. Nakagawa. 2010. Analysis of factors to make prosodic change in spoken dialog. *Journal of Phonetic Society of Japan*, 13(3):66–84.

C. D. Manning and H. Schütze. 2001. *Foundations of Statistical Natural Language Processing*.

P. C. Nichols. 1983. Linguistic options and choices for black women in the rural South. In, B. Thorne, C. Kramarae and N. Henley (eds.) *Language, Gender and Society*. Cambridge: Cambridge University Press, 335–358.

S. Ide. 1982. Japanese sociolinguistics: Politeness and women's language. *Lingua*, 57:357–385.

G. Escure. 1991. Gender roles and linguistic variation in the Belizean Creole community. In, J. Cheshire (ed.) *English around the World: Sociolinguistic Perspectives*. Cambridge: Cambridge University Press, 595–609.

M. G. Wiley and A. Eskilson. 1985. Speech styles, gender stereotypes, and corporate success: What if women talk more like men? *Sex Roles*, 12(9/10):993–1007.

E. Shriberg and A. Stolcke. 2008. The case for automatic higher–level features in forensic speaker recognition, *Proceedings of Interspeech 2008*, 1509–1512.

Corpus Based Classification of Text in Australian Contracts

Michael Curtotti
Australian National University
Canberra, Australia
michael.curtotti@anu.edu.au

Eric Mccreath
Australian National University
Canberra, Australia
eric.mccreath@anu.edu.au

Abstract

Written contracts are a fundamental framework for commercial and cooperative transactions and relationships. Limited research has been published on the application of machine learning and natural language processing (NLP) to contracts. In this paper we report the classification of components of contract texts using machine learning and hand-coded methods. Authors studying a range of domains have found that combining machine learning and rule based approaches increases accuracy of machine learning. We find similar results which suggest the utility of considering leveraging hand coded classification rules for machine learning. We attained an average accuracy of 83.48% on a multi-class labelling task on 20 contracts combining machine learning and rule based approaches, increasing performance over machine learning alone.

1 Introduction

Contracts govern economic transactions and commercial or organisational relationships from trivial purchases to major national infrastructure projects. Any large organisation (whether private or public) must unavoidably invest significant resources in developing and concluding contracts, as the contracts it enters define its legal relations with organisations and individuals with which it interacts. As one author has noted, contracts are an integral part of any business enterprise and it is "difficult to overstate their importance to the business world" (Khoury and Yamouni, 2007, p16). In addition, drafting contracts represents a major economic activity for the legal industry.

Ambiguity is an unresolved problem in natural language processing. It is also an important problem affecting the drafting and operation of contracts. The costs of ambiguity in a contract can be enormous. The presence of ambiguity increases the risk of litigation, project failure and loss of commercial relationships. Instances of litigated ambiguity are used in drafting manuals as instructional tales for the unwary drafter (Aitken and Butt, 2004, p37).

Our future research goal is to investigate approaches that will help identify such ambiguity, however, a precursor for addressing the problem of ambiguity is accurate labelling or classification of the constituent parts of contract documents.[1] In this paper we report work on classifying 'lines' in contracts into 32 classes. We compare three different classification approaches in order to determine which approach may provide the best performance with respect to this task: supervised machine learning; a hand-crafted rule based tagger; or combining hand-crafted rules based tagging with machine learning.

2 Background

Natural language processing involves applying a pipeline of transformations to text. A typical first step is to segment the text into its sentences, after which further processing is applied (Jurafsky and Martin, 2009, p 69). This works well in usual English prose, but is not necessarily well suited to contracts because of the heavy use of layout to embody structure and semantics. In a contract, a sentence may occur within a single paragraph of text, or may be spread over several line breaks due to the use of sub-paragraphing as an aid to comprehension. Such sub-paragraphs may embody disjunctive or conjunctive conditions associated with

[1] This task can also be thought of as a document segmentation task, although we here consider it as a classification exercise.

Michael Curtotti and Eric McCreath. 2010. Michael Curtotti and Eric McCreath. In *Proceedings of Australasian Language Technology Association Workshop*, pages 18−26

a rule. A 'clause'[2] is the typical structural unit which embodies a legal rule or set of rules. Sometimes clauses are organised into sub-clauses and clauses and sub-clauses may be numbered in hierarchical fashion (e.g. 1, 1.1, (a), (i), (A))(Aitken and Butt, 2004, pp 23, 27). Line breaks often occur immediately before data such as headings, clauses, sub-clauses, party names, dates or execution blocks. Being able to label a line as to its character, key content or function thus enriches available information for later NLP. The line is thus a valuable starting point for applying NLP.

3 Related Work

3.1 Classification and Segmentation of Contracts and Legislative Documents

There would appear to be little published work on the application of natural language processing (NLP) or machine learning specifically for the classification of text in legal contracts. An exception to this is (Indukuri and Krishna, 2010) who apply a support vector machine over feature vectors extracted from contract clauses. These feature vectors are comprised of n-grams up to size n = 4. They classify contract sentences as clauses or non-clauses and then sub-classify clauses as concerned with payment terms or not. Indukuri and Krishna note the lack of published work on classification of sentences in contract documents for workflow and monitoring purposes.

While also limited, there has been some work on segmentation or classification of legislative corpora using machine learning. (Mencía, 2009) reports 100% precision and recall in machine learning on classification of legislation into articles, sections and parts (although to a task for which he is able to craft an equally accurate regular expression based segmenter). (Bacci et al., 2009) automatically classify plain text legislative documents for the purpose of XML mark up. (Francesconi, 2006), on whose work Bacci et al. build, describes four separate modules for dealing with plain text legacy content which they are seeking to convert to XML tagged text. (Hasan et al., 2008) carry out segmentation of Spanish legislative bulletins using the table of contents as an aid to segmentation. This work, in the legislative field, provides a parallel application domain for the work undertaken here in respect of contracts, although it is generally true that legislation will tend to conform to far tighter stylistic rules, because of the central control of legislative drafting.

3.2 Combining Machine Learning and Hand Coded Approaches

A number of researchers have combined hand-coded approaches and machine learning to produce better overall results.

(Kipp, 2006) augments handcrafted rules with machine learnt rules for gesture generation.

(McCreath and Kay, 2003) used such a combined approach to improve performance on categorizing emails.

(Park and Zhang, 2003) address the problem of text chunking for the Korean language, combining hand-crafted rules with a memory based machine learning method. A rule based method is used to determine whether an instance may constitute an exceptional case, with machine learning used to correct classifications assigned by the rule based method. They report up to a 2.83% improvement in F-scores over memory-based learning alone.

(Rochery et al., 2002) report the use of hand-crafted rules to enhance the accuracy of machine learning utilising boosting. They apply the method to a multi-labelling classification problem where they are seeking to classify spoken utterances. They attribute increased accuracy to the fact that the rules they develop are not represented in their data at all or do not appear sufficiently to have statistical impact during the training process. They note that the combination has the greatest effect when there are less then 100 examples, with the benefit of the combination decreasing as the number of training examples increases. The method therefore has direct applicability to reducing the data requirement for machine learning (a costly aspect of supervised machine learning).

(Takahashi et al., 2005) utilise a combination of SVM based learning and hand-crafted rules for the task of classifying occupations from social surveys. They effect a combination by using rule derived labels as features for the machine learner. Again the work occurs in a multi-class labelling context (with around 200 occupation codes). They find that SVM approaches are superior to rule-based approaches alone but that combining rule-based and SVM learning produces the best results. They also examine the effect of the number of training examples, noting that the differential ben-

[2]The term 'clause' here refers to a legal rule or rules appearing in a contract, rather than the linguistic entity.

efits of combining methods reduces with increasing data.

In our study we use class labels assigned by a hand-coded tagger as additional features for machine learning to obtain an increase in accuracy in the domain of text classification.

The work reported above, and our own paper, come from diverse domains and illustrate the application of combined rule/ML approaches using a variety of machine learning algorithms.

This suggests the wide utility of using hand-coded rules to improve accuracy of machine learning. Symetrically we may say that machine learning may potentially be used to improve the accuracy of hand-coded rules. Here we essentially follow a hybrid model of development that examines each method separately as well as in combination.

4 Data

4.1 Australian Contract Corpus

The data which forms the basis of this study is a corpus of 256 contracts ('the Australian Contract Corpus') which has been compiled from the web using the Google Australia web search: `http://www.google.com.au`.[3] The corpus is constituted of 1043364 tokens (words and punctuation), 42910 sentences, and a vocabulary of 14217. The lexical diversity of this corpus is 73.39.[4] As far as we are aware there is no published work based on a corpus of Australian contract language, however space does not permit a fuller discussion of the design and analysis of the corpus.

4.2 Data Representation

For the purpose of this study we randomly selected 30 contracts from the Australian Contract Corpus to produce data sets for application of machine learning. The 30 contracts were divided into three sets: a training set (Set A)[5], a second set (Set B)[6] (primarily used as a test set, although sometimes combined with Set A to form a larger training set) and an additional test set (Set C)[7], each of 10 contracts. Set C was added to increase both available data and the number of contracts on which testing could be carried out.

To simplify processing all contracts were converted into text files as a preprocessing step.[8] Also we removed material that typically does not appear in contracts (primarily guideline notes for drafters).

For purposes of our study each 'line' or 'paragraph' in a contract constituted a data point. Sets A, B and C provided 1825, 2157 and 2231 lines/data points, respectively.

To classify our data for machine learning we first employed our hand-coded tagger to produce a labelled data set constituted of a line and primary and secondary labels according to our classification system (see below). The classifications applied by the hand-coded tagger were then manually corrected to remove any labelling errors. Both for the reason that hand tagging data from scratch is more error prone than correcting previously machine tagged data, and because of the high labour in manual tagging, machine assisted tagging is a standard method used in corpus creation (Biber et al., 1998, p 262), ((ed) Christopher S. Butler, 1992, p 131).

4.3 Feature Selection

Features were then extracted from our labelled lines using a hand-coded feature extractor to extract up to 40 features to be used for machine learning, with the primary class label earlier applied to each line serving as the target for machine learning.[9] Features for machine learning were selected based on assumed relevance to the intended classification.[10] While a bag of word representa-

[3]The search phrase we employed was: "clause party agreement" , limiting the search to "pages from Australia" and the filetype to ".doc". The collection of the corpus was undertaken in the period 6 - 24 December 2009. Each document was visually inspected by one of the authors to verify that it was an Australian contract and documents were added to the corpus in order of their appearance in Google search results. We have made available the list of URLs of the documents that make up the corpus at: `http://cs.anu.edu.au/people/Michael.Curtotti/`. We also make code and data referred to in this paper available at the same location.

[4]Lexical diversity refers to the ratio of the total number of tokens in the corpus to the total number of types of tokens i.e. tokens/vocabulary. The statistics and information provided here were extracted using the NLTK. (Bird et al., 2009).

[5]Contracts numbered in our corpus: 55, 74, 77, 91, 94, 144, 174, 185, 208 and 213

[6]Contracts 11, 12, 164, 193, 196, 199, 249, 59, 64, 9

[7]Contracts 200, 254, 175, 180, 120, 102, 207, 127, 75 and 79

[8]Notably, this step does have the downside of discarding valuable style and layout information found in the word document format.

[9]In some instances the assignment of class is not entirely disjunctive and implies a priority ordering of potential classifications. For instance a clause may contain address details, and be classified as such rather than as a clause. The feature extractor removed secondary labels.

[10]The full list of features is as follows: 1. relative position of a line within the contract; 2. line end punctuation; 3. num-

tion with automatic feature extraction was trialed, this was found to result in lower accuracy than a manually selected (but small) number of features. Traditionally, 'feature engineering' or 'selection' in text classification has focussed on 'bag of words' or proximate features such as n-grams or noun phrases (Scott and Matwin, 1999). Our selection of features might be regarded as 'hybrid' in that they implicitly encode domain knowledge and 'rules'. For example, we expect a definition line to contain the widely used word 'means' as in 'x means y'. We would expect clause headings to display a different pattern of parts of speech occurrence, as compared to clauses themselves. We also consider features such as the relationship between the particular line/datapoint and its neighbouring lines (e.g. is the preceding line long or short), or the document as a whole (e.g. relative placement of the line within the document).

Features used in the related work carried out by others on contracts and legislation include: n-grams (Indukuri), bag of word together with capitalisation and character patterns (Mencia), the use of sequential word and token data as a HMM to predict state where state is the classification of the portion of text (Bacci/Francesconi) and the use of word terms identifying titles and lists of indexed terms (Hasan).

4.4 Classes

We adopted 32 classes for our classification scheme. These classes represented significant structural elements (such as clause headings or content lines) or sometimes key contract meta data (for instance the parties, the date on which an agreement is made, email addresses, ABNs etc).[11] Of particular interest to us, was an ability to accurately identify clausematter (the primary location of legal rules of a contract) and clause headings (which in some contracts effectively mark boundaries between major rule sets).

5 Experimental Evaluation

Analysis of our data sets was carried out using two major methods: analysis using the training Set A measuring peformance on test Set B (i.e. testing for accuracy of classification of lines in Set B); and analysis using training Set A where each contract in test Sets B and C was used individually as a test set (testing for accuracy of classification of lines of each individual contract). The former testing approach is more widely used but the latter is more consistent with our intended end application: that is, developing an ability to accurately classify lines in a previously unseen contract as an individual document.

Set A consisted of 1825 lines of data, Set B consisted of 2157 lines of data and Set C consisted of 2231 lines of data.

5.1 Tools

The following tools were used to carry out machine learning tasks reported in this paper:

1. the *python* programming language was used to develop a hand coded line tagger, a feature extractor and code for evaluation of the performance of the hand coded tagger;

2. the Natural Language Toolkit (NLTK)(Bird et al., 2009) was used in corpus development and in the hand coded tagger for application of parts of speech tags; and

3. the Weka data mining software(Hall et al., 2009) was used to carry out machine learning.

bering at the beginning of the line; 4. line length in characters; 5. line length in tokens; 6. number of nouns; 7. number of adjectives; 8. number of verbs; 9. number of prepositions; 10. number of coordinating conjunctions and cardinal numbers; 11. number of modal verbs; 12. number of personal pronouns, possessives; 13. number of adjectives; 14. whether must occurs; 15. whether may occurs; 16. whether shall occurs; 17. whether means occurs;18. the position of the word means in the line; 19. whether the word 'include' occurs; 20. the position of the word 'includes' in the line; 21. whether the phase 'has the same meaning' occurs; 22. whether the line begins with a capital letter followed by a stop or a space; 23. do the letters 'abn' or 'acn' appear; 24. does the word address appear near the beginning of the line; 25. does the word 'contact' appear near the beginning of the line; 26. does the word 'email' appear near the beginning of the line; 27. does the word 'fax' appear near the beginning of the line; 28. does the word 'note' appear at the beginning of the line; 29. does the word 'phone' appear at the beginning of the line; 30. do the terms 'web' or 'www' appear near the beginning of the line; 31-36. token lengths of 3 lines before and after the data line; 37. the tag applied by the hand coded tagger to the data line; 38. whether the contract from which the line comes has clause headings; 39. whether the contract from which the line comes has clause sub-headings; 40. whether the contract has a schedule.

[11] The following class labels were applied: CLAUSEMATTER, CLAUSEHEAD, DEF, CONTENTSHEAD, BLANK, DROSS, AND, BETWEEN, DATEMADELINE, PARTIESHEAD, PARTYLINE, RECITALHEAD, RECITALLINE, HEAD, CONTENTLINE, PRELIM, OPERATIVEPROVISIONLINE, NUMBEREDLINE, SCHEDULEHEAD, SCHEDULEITEM, EXECUTIONBLOCK, NOTE, ABNLINE, ADDRESSLINE, EMAILLINE, WEBLINE, FAXLINE, PH-LINE, REF-LINE, SCHOTHERMATTER, CONTACTOFFICER, TITLELINE.

5.2 Hand-Coded Tagger

Our hand-coded tagger follows a pipeline methodology. A contract is separated into a list of lines. In the first phase, each line is passed through a series of "if-then-else" routines (essentially utilising regular expressions) to assign primary and secondary labels to the line. This produces a contract with enhanced information content, in the form of a tuple of labels and line text. This output is then fed through a series of methods, progressively bootstrapping improvements to labelling, based on the information available when the method is applied. For instance content lines will be identified, these are then used in later methods to seek to identify clause headings. In addition, the tagger extracts contract meta data such as: a list of definitions, a list of clause headings, where the beginning of operative provisions occurs, where the schedule begins. Such features may or may not occur in a contract. In addition the tagger also provided a method for hand correcting tagging which was used to create our data set for supervised machine learning purposes.

Development of the tagger relied heavily on domain knowledge to identify logical tests for label application. For instance the occurrence of numbering at the beginning a line was used to identify clause headings, or the presence of a common text pattern which occur in definitions i.e. (the pattern *["word[1-3]] means [...]"* was used as a criterion for identifying a line as a definition).

	$\overline{x}$	s
On Set B Contracts	86.27%	NA
Contract x Contract	82.84%	10.85%

Table 1: Accuracy of Line Tagger.

The first row of Table 1 reports accuracy of the hand coded tagger when applied to Set B as the test set (the % accuracy of tagging of individual lines in Set B is reported). The second row of Table 1 reports average accuracy with which the lines in each of 20 contracts in Sets B and C are tagged (standard deviation is also reported). The high variance of the results at a contract by contract level suggests a need for future work to identify if issues such as contract 'type' may account for the differences (the lowest level of accuracy was around 50% and the highest around 95%).

Table 2 provides measures of precision, recall and F-measure attained by the hand coded tagger on Set B for key data items: clause headings, clauses themselves and definitions.[12] These results on key measures are sufficient for our initial purposes and will allow us to focus subsequent work on addressing ambiguity in the substantive rules found in a contract.

	Precision	Recall	F-Measure
CL.HEAD	0.98	0.83	0.89
CL.MATTER	0.91	0.91	0.91
DEF	1.00	0.66	0.80

Table 2: Performance results for Hand-Coded Line Tagger

5.3 Supervised Machine Learning

Column 1 of Table 3 reports the accuracy of various supervised machine learning algorithms with Set A as training set and Set B as test set. Of the algorithms applied, Random Forest (implemented using 100 trees and 30 random features) performed best. No ML method performed better than the hand coded tagger tested on the same test set (see row 1 of Table 1).

5.4 Machine Learning vs Hand-coded methods

Given our ultimate focus on developing applied tools, we are primarily concerned to identify the best methods to maximise the accuracy of our classification task. Whether this proves to be machine learning methods or hand-coded methods or a combination of the two is immaterial from this viewpoint (other than in regard of development costs).

Also it is useful to consider whether methods and insights from one method may assist in the other. For instance, hand-coded rules that successfully labelled lines directly suggested 'features' to be extracted for machine learning purposes. Conversely the identification of 'features' for machine learning can be regarded as a programming abstraction where the algorithm is 'under the hood' but human intervention is required in the form of feature selection for 'learning' to occur (implicitly encoding of rules in the feature set). In deriving our results we used a confusion matrix as a tool to assist in identifying areas where the hand-

[12] Precision is $\frac{TP}{TP+FP}$, Recall is $\frac{TP}{TP+FN}$, and F-Measure is $\frac{2PR}{P+R}$, where TP is true positives, FP is false positives, FN is false negatives, P is precision, and R is recall.

coded tagger required improvement and as a common baseline for performance. In certain forms of machine learning, the resulting machine learning model can be represented as code in "if, then, else" form (as in the output of a decision list or decision tree learner). We may thus conceive of machine learning and hand coding of rules as different 'programming paradigms' - a perspective that encourages us to apply insights across the boundary between them.

Further, we wished to explore the relative accuracy of machine learning vs hand coding for this particular task. Such an exploration is valuable from a cost benefit point of view. Supervised machine learning (where we have focused our attention) requires the production of human annotated data (which is costly to produce) as well as feature testing and refinement. Also machine learning may be computationally intensive. Production of an accurate hand-coded tagger requires expert knowledge and considerable testing and code refinement in order to achieve a high level of accuracy (which again is resource intensive). We did not undertake quantitative data collection in respect of time employed in the two methods, particularly as we followed an iterative development model that switched between rule based and machine learning focussed work and as we were searching for optimal outcomes. Nonetheless we anecdotally report that each method presented significant demands in terms of development time.

In the results reported above we note that no machine learning approach alone exceeded the performance of the hand coded tagger when considering performance on test Set B as a whole. This is the result also found at contract by contract level (see below Table 4).

5.5 Leveraging Hand-Coded Labels for Machine Learning

As noted above we also wished to explore whether utilising the output from the hand coded tagger as input features for machine learning might improve the accuracy of machine learning alone. A converse question that could be framed is whether the rule based tagger's performance could be improved by combination with machine learning.

We carried out two trials that bear on these questions:

1. machine learning alone and machine learning enhanced with rule based input on an entire test set (Set B); and
2. machine learning alone, rule alone and machine learning enhanced with rule based input at contract by contract level.

ML Algorithm	ML Alone	ML + Rule
Naive Bayes	82.38%	84.75%
SMO (SVM)	82.80%	85.40%
Cl. via Regression	83.91%	87.07%
Decision Tree	82.01%	87.02%
Bagging	83.54%	87.76%
Majority Vote	84.42%	87.11%
Random Forest	85.12%	86.69%

Table 3: Comparative Performance ML alone and ML + Rule on train and test set

Column 2 of Table 3 shows the improvement in performance of various machine learning methods when tags applied by the hand coded tagger were provided as input features. All methods show improvement with addition of the rule derived feature on Set B as a whole.[13]

We report below a comparison of accuracy of machine learning, rule based tagging and a combination of both in respect of Decision Trees. Decision trees offer the advantage that the output is easily interpretable as a rule set for tagging.

Method	$\overline{x}$	s
ML Alone	79.12%	10.71%
Rule Alone	82.84%	10.85%
ML + Rule	83.48%	9.83%

Table 4: Tagging Accuracy Mean and Standard Deviation - Contract by Contract

Although the results reported in Table 4 are different (as different contracts are included in the tested contracts)[14] average accuracy continued to show a differential between machine learning alone, and machine learning leveraged with hand rules, as described above. On this combination ML + Rule also outperformed the rule based tagger alone, but by a very narrow margin. In 17 of 20 cases ML + Rule improved or did not worsen performance of ML alone. In 15 of 20 cases, ML +

[13]The training set for the results reported in Table 3 was Set A. The test set was Set B.

[14]The training set was Set A. Sets B and C provided 20 individual contracts (effectively 20 test sets) for testing.

rule was no worse or outperformed the hand coded tagger alone. Notably variation is quite high as compared to the mean. Testing for statistical significance we can reject a null hypothesis that the mean accuracy for ML + Rule is the same as for ML alone. We cannot reject a null hypothesis that ML + Rule is the same as Rule alone.[15] We note also that this investigation was only carried out in respect of the decision tree learner.

5.5.1 Is it more effective to increase data or use rule based features?

To further explore the question of how much effort might be saved by combining rule based approaches with machine learning alone we tested the effect of adding random noise to the feature set provided by the hand coded tagger. We found that we had to randomly flip over 45% of these tags before they ceased to impact positively on classification accuracy. This suggests that even poorly prepared hand coded rules can improve performance.[16]

We also tested the effect of incrementally increasing the amount of available training data to explore any decrease in differential in performance. Testing using Set C as the test set and taking training data from Sets A and B, we progressively increased training data by increments of 500 instances of data. Differential performance decreases progressively, starting around 5% and reducing to 2.5% as more training data is made available.[17] (See Table 5.) Extrapolating from these figures we would expect at a minimum that training data would have to be doubled to remove differential accuracy. Relatively little effort in developing rule based features may substitute for considerable work in creating additional supervised data.

[15]We gratefully acknowledge the assistance of Bob Forrester of the ANU Statistical Consulting Unit in deriving this result, undertaken utilising a generalised linear model to undertake model fitting adjusting for different file (contract) types. Predictions of estimated mean proportions were: ML Alone 0.7760 (s.e. 0.007181), Rule alone 0.8106 (s.e. 0.006775), ML+ Rule 0.8181 (s.e. 0.006677).

[16]Using a Decision Tree, ML + Rule accuracy on test Set B with training Set A, was 87.0% with zero noise, 86.9% with 5% noise, 84.7% with 10% noise, 85.2% with 20% noise, 84.4% with 30% noise, 83.9% with 40% noise, 82.8% with 45% noise, 82.4% with 50% noise. Decision Tree accuracy on ML alone on this training and test set was 82.0%.

[17]The data has a negative correlation of -0.70

Data	0.5	1K	1.5	2K	2.5	3K	3.5	4K
Diff.	5.1	3.4	5	4.2	3.3	3.1	3.9	2.6

Table 5: Difference in % accuracy of ML alone and ML with rule based feature with increasing training data.

5.5.2 Decision tree output

An examination of the decision tree produced after adding the output of the hand-coded tagger to machine learning is of interest.

```
LTlabel = #CLAUSEMATTER#
| tknLngth <= 4
|| nounNum <= 1: #CLAUSEMATTER# (6.0/2.0)
|| nounNum > 1
||| linePos. <= 0.383929: #PRELIM# (2.0)
||| linePos. > 0.383929: #CLAUSEHEAD# (2.0)
| tknLngth > 4: #CLAUSEMATTER# (508.0/7.0)
: :
: :
LTlabel = #PH-LINE#
| prepNum <= 1: #PH-LINE# (5.0)
| prepNum > 1: #CLAUSEMATTER# (2.0)
LTlabel = #EMAILLINE#: #EMAILLINE# (3.0)
LTlabel = #ABNLINE#
| preLine2 <= 0: #TITLELINE# (2.0/1.0)
| preLine2 > 0: #ABNLINE# (3.0)
```

Figure 1: Part of the decision tree learnt when the labels from the hand-coded are provided as an attribute.

Using only the machine learning feature set, the size of the pruned tree is 199 with 108 leaves. After addition of labels provided by our hand-coded tagger the size of the decision tree is reduced to 105 with 69 leaves. Figure 1 shows part of this decision tree and as illustrated, the decision tree generally begins with the label assigned by the hand-coded tagger 'LTlabel' and amends the label (where necessary) utilizing other features available to the machine learner. Some labels (e.g. EMAILLINE) were simply adopted without change. In the case of the CLAUSEMATTER label, the machine learner used the number of tokens in the line ('tknLngth'), number of nouns in the line ('nounNum') and the relative position of the line in the contract ('linePos.') to re-assign the class of the line. The significance of the use of labels assigned by the hand-coded tagger is that the decision-tree identifies that attribute as providing the most significant information gain (that is reduction in entropy) and therefore partitions on that feature (Callan, 2003, p245). This remains true in this example irrespective of the class to which

the data is being assigned.[18] It is obvious that the best feature for assigning the correct label for an item of data is the correct or probably correct label (if one can obtain it without undue endeavour). From a practical viewpoint some effort directing to hand-crafting code to extract appropriate class labels may thus assist machine learning.

6 Conclusions and Future Work

In terms of the overall performance of the methods described in this paper the closest point of comparison is the work reported by Indukuri and Krishna (Indukuri and Krishna, 2010). While comparisons using different methods, software and data are of dubious validity, we may note their broad similarity. With a training set of 10 contracts and a test set of 20 contracts, the contract-by-contract average F-measure for the clausematter class was 0.8632. Also in 14 contracts of the 20 tested the F-measure for the clausematter class was above 0.85. With a training set of 10 contracts (Set A) and a test set of 10 contracts (Set B) (as a single test set) the F-measure for the clausematter class was 0.921.[19] Using 4-grams as features, Indukuri and Krishna report an accuracy of 83.48% on the task of classification clauses from non-clauses, applied to 73 sentences to be so classified. Whereas the task in that case was a binary classification task over one contract, we report results on multi-class classification over test sets of up to 20 contracts.[20]

The work we have undertaken and related work reviewed in this paper (Section 3) suggests the potentially broad utility of combining rule based and machine learning methods. One potential hybrid classification method in its simplest formulation may look something like this:

1. determine the required or anticipated level of accuracy for the intended application;

2. develop a simple code rule set for the classification task (i.e. with minimal expenditure of resources);

3. if classification accuracy is sufficient machine learning would not be required, if not proceed as usual in development of machine learning using the output of hand coded rules as an input to machine learning.

Such a development model does not avoid the development of a supervised training set, as such a set is required both to assess the accuracy of the rule based tagger as well as the machine learner (should development proceed to that stage). However it may reduce the amount of data required to attain a desired level of accuracy. The method could of course be applied iteratively, if the accuracy level is insufficient after a first iteration.

Future work to be explored includes the effective "typing" of contracts (particularly in light of the high variance of performance on individual contracts). For instance, some contracts use clause headings, others do not. Some use schedules, others do not, etc. Such type information may further assist in the classification task (the accuracy of which varies considerably depending on the contract to which it is applied). In typical preprocessing, documents in corpora are converted to plain text. This results in the loss of layout and hierarchical information found in word documents. The preservation of such information for use in classification tasks may improve accuracy.

The preliminary work here is also a precursor to work on identification of ambiguity and the development of practical tools to assist in contract drafting. In separate work, we intend also to describe and analyse the contract corpus that is referred to here as the basis of this work.

[18] The Weka decision tree implementation, "J.48", is based on the Quinlan's C4.5(Quinlan, 1986).

[19] The results reported here are derived using a random forest algorithm and using the hand coded tag as an input feature for classification.

[20] Coincidentally the 83.48% figure is the same as our average result over all classes for 20 contracts.

References

J. K. Aitken and P. Butt. 2004. *Piesse The Elements of Drafting*. Lawbook Co, 10th edition.

L. Bacci, P. Spinosa, C. Marchetti, R. Battistoni, I. Florence, I. Senate, and I. Rome. 2009. Automatic mark-up of legislative documents and its application to parallel text generation. In *Proc. of LOAIT Workshop*, pages 45–54.

D. Biber, S. Conrad, and R. Reppen. 1998. *Corpus linguistics: Investigating language structure and use*. Cambridge University Press.

Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Rob Callan. 2003. *Artificial Intelligence*. Palgrave Macmillan.

(ed) Christopher S. Butler. 1992. *Computers and written texts*. Blackwell.

E. Francesconi. 2006. The Norme in Rete Project: Standards and Tools for Italian Legislation. *International Journal Legal Information*, 34:358.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software. *SIGKDD Explorations*, 11(1).

I. Hasan, J. Parapar, and R. Blanco. 2008. Segmentation of legislative documents using a domain-specific lexicon. In *Proceedings of the 19th International Conference on Database and Expert Systems Application*, pages 665–669.

Kishore Varma Indukuri and P. Radha Krishna. 2010. Mining e-contract documents to classify clauses. In *COMPUTE '10: Proceedings of the Third Annual ACM Bangalore Conference*, pages 1–5, New York, NY, USA. ACM.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, 2nd edition.

D. Khoury and Y. S. Yamouni. 2007. *Understanding Contract Law*. Lexis Nexis Butterworths, 7th edition.

M. Kipp. 2006. Creativity meets automation: Combining nonverbal action authoring with rules and machine learning. In *Intelligent Virtual Agents*, pages 230–242. Springer.

Eric McCreath and Judy Kay. 2003. Iems: Helping Users Manage Email. *User Modeling 2003*, pages 146–146.

Eneldo Loza Mencía. 2009. Segmentation of legal documents. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, pages 88–97. ACM.

Seong-Bae Park and Byoung-Tak Zhang. 2003. Text chunking by combining hand-crafted rules and memory-based learning. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 497–504.

J.R. Quinlan. 1986. Induction of decision trees. *Machine Learning*, 1:81–106.

M. Rochery, R. Schapire, M. Rahim, N. Gupta, G. Riccardi, S. Bangalore, H. Alshawi, and S. Douglas. 2002. Combining prior knowledge and boosting for call classification in spoken language dialogue. In *Proceedings of the IEEE International Conference of Acoustics and Speech and Signal Processing*.

S. Scott and S. Matwin. 1999. Feature Engineering for Text Classification. In *Machine learning: proceedings of the Sixteenth International Conference (ICML'99), Bled, Slovenia, June 27-30, 1999*, page 379. Morgan Kaufmann Pub.

K. Takahashi, H. Takamura, and M. Okumura. 2005. Automatic occupation coding with combination of machine learning and hand-crafted rules. *Advances in Knowledge Discovery and Data Mining*, pages 269–279.

Thread-level Analysis over Technical User Forum Data

Li Wang, Su Nam Kim and Timothy Baldwin

NICTA VRL
Department of Computer Science and Software Engineering
University of Melbourne
VIC 3010 Australia

li.wang.d@gmail.com, sunamkim@gmail.com, tb@ldwin.net

Abstract

This research focuses on improving information access over troubleshooting-oriented technical user forums via thread-level analysis. We describe a modular task formulation and novel dataset, and go on to describe a series of preliminary classification experiments over the data. We find that a class composition strategy achieves the best results, surpassing multiclass classification approaches.

HTML Input Code - CNET Coding & scripting Forums

User A Post 1	**HTML Input Code** ...Please can someone tell me how to create an input box that asks the user to enter their ID, and then allows them to press go. ...
User B Post 2	**Re: html input code** Part 1: create a form with a text field. ...
User C Post 3	**asp.net c# video** I've prepared for you video.link click ...
User A Post 4	**Thank You!** I have Microsoft Visual Studio 6, what program should I do this in? ...
User D Post 5	**A little more help** ...You would simply do it this way: ...

Figure 1: An extract from a real-world thread

1 Introduction

Online forums and discussion boards are online platforms for people to hold discussions in particular domains. They are widely used in various areas such as customer support, community development, interactive reporting and online education. Forums provide one of the primary avenues for online users to share information on the Internet. Users post their questions or problems onto online forums and get possible solutions from other users. Through this simple mechanism, great volumes of data with customised answers to highly specialised domain-specific questions are created on a daily basis. However, it is not an easy job to extract the information latent in the threads.

The aim of our research is to help users to more easily access existing information in forums which relate to their questions, by text mining troubleshooting-oriented, computer-related technical user forum data (Baldwin et al., 2010). An example thread from a real-world forum is shown in Figure 1, which is made up of 4 posts with 3 distinct participants.

Our proposed strategy is to model the "content structure" of forum threads by analysing requests for information and provision of solutions in the thread data. We devise an ontology of *problem sources* and *solution types* with which to analyse individual threads, paving the way for users to spell out the general nature of their support need in their queries. The main contributions of this paper are: (1) designing a modular thread-level class set; (2) constructing and publishing an annotated dataset; and (3) performing preliminary thread-level experiments over the dataset.

2 Related Work

There is very little work that is specifically targeted at the thread-level analysis of web user forum data. The most closely-related work is that performed by Baldwin et al. (2007), and our thread class set was created based on this original work.

Another research line that relates to the thread classification is discussion summarisation. For example, technical online IRC (Internet Relay Chat) discussions are summarised and segmented in Zhou and Hovy (2005)'s research. The message segments are then clustered to find the most relevant information to users using machine learning models. There has also been work on email summarisation, concentrating primarily on summarising and organising email archives by extracting overview sentences to help users find the most useful email threads (Nenkova and Bagga, 2004;

Li Wang, Su Nam Kim and Timothy Baldwin. 2010. Thread-level Analysis over Technical User Forum Data. In *Proceedings of Australasian Language Technology Association Workshop*, pages 27−31

Rambow et al., 2004; Wan and McKeown, 2004).

3 Class Definition

The proposed thread class set is made up of two orthogonal Basic Class sets (BASIC), and a Miscellaneous Class set (MISC). The first BASIC class set is Problem Source (PROBLEM), which concerns the targets and sources of the problem described in threads. It contains 6 basic classes: *Operating System* (*OS*), *Hardware*, *Software*, *Media*, *Network* and *Programming*. The second BASIC class set is Solution Type (SOLUTION), which describes the types of the solution presented in threads in the form of 4 classes: *Documentation*, *Install*, *Search* and *Support*. The MISC class set includes two classes: *Other* and *Spam*. A detailed description of each class in the thread class set is presented in Table 1.

A given thread is labelled either with one class label from each of the two BASIC class sets (i.e. two class labels in total), or alternatively one class label from the MISC class set. For example, the thread from Figure 1 would be labelled as *Programming*/*Documentation*. Therefore, when doing the actual annotation, we used the ALLCLASS class set containing 26 classes in total, i.e. the cross product of the two BASIC class sets plus *Other* and *Spam*.

It should be noted that while the design of our class set is specific to computer-related technical user forum threads, the idea of the two orthogonal BASIC class sets, namely PROBLEM and SOLUTION, can be applied to troubleshooting-oriented forum threads from other domains. This is because most troubleshooting-oriented forum threads present one or more problems (i.e. PROBLEM), and imply possible solution types (i.e. SOLUTION), even if the thread is unresolved.

4 Data Collection

This research focuses exclusively on data from CNET forums.[1] Firstly, 1000 threads were crawled from CNET forums using SiteScraper.[2] We only collected threads that contained 2 to 16 posts, as threads containing only 1 post have no answers and cannot provide solutions, and long threads tend to be more discussion-oriented and/or contain multiple sub-threads.

[1] http://forums.cnet.com

[2] http://sitescraper.googlecode.com/

The crawled threads were then preprocessed. Only the title and sub-forum information of each thread, and the body, title, and author information of each post were preserved. Finally, we randomly selected 500 threads from 4 sub-forums of the CNET forums: Operating Systems, Software, Hardware, and Web Development.

Two annotators performed a pilot annotation using a seed set of 150 threads and a dedicated web annotation tool. The κ value for the pilot annotation (indicating the relative agreement between the two annotators) was 0.43. The annotators sat down together to go over every thread where there were disagreements, and discussed the disagreements based on the class descriptions. Then, the two annotators annotated 327 new threads, achieving a more respectable κ value of 0.74. The annotators furthermore met again to resolve any disagreements in the labelling of the 327 threads. Most of the disagreements arose from confusion between Hardware and Media in the PROBLEM set, and Documentation and Support in the SOLUTION set.

5 Experimental Methodology

We carried out preliminary experiments over the annotated data, focusing on the implications of the modular class design for thread classification.

As our feature representation, we firstly removed all punctuation in the threads and normalised the threads to lower case. Then, we lemmatised the threads using the GENIA Tagger (Tsuruoka et al., 2005), and removed stopwords.[3] Based on the preprocessed threads, we used a bag-of-words term frequency representation, concatenating all posts in the thread into a single meta-document and thereby treating the task as a document categorisation task.

All of our experiments were carried out using Hydrat (Lui and Baldwin, 2009), a classifier comparison framework. Hydrat integrates several machine learning software packages including BSVM (Hsu and Lin, 2006), weka (Hall et al., 2009) and MALLET (McCallum, 2002), in addition to native implementations of a number of more basic learners. In our experiments, we tried a range of machine learning models including Support Vector Machines (SVM), multinominal Naive Bayes (NB), and instance-based learners

[3] Using the stop word list from InfoMap (http://infomap-nlp.sourceforge.net/).

Class Category		Description
PROBLEM:	*OS*	Operating system
	Hardware	Core computer components, including core external components (e.g. a keyboard)
	Software	Software-related issues, including applications and programming tools
	Media	Hardware which is either a non-standard external component or peripheral device
	Network	Network issues (e.g. connection speed, and installing a physical network)
	Programming	Coding and design issues relating to programming
SOLUTION:	*Documentation*	How to use a certain function, select a computer/component, or perform a task
	Install	How to install a component
	Search	Search for a particular component (e.g. a software package)
	Support	How to fix a problem with a computer or component
MISC:	*Other*	Troubleshooting-related, but the problem source is not included in the PROBLEM set
	Spam	The thread is not troubleshooting-related

Table 1: The components of the thread class set

(NN). A majority-class model (ZEROR) was used as the baseline.

The class set was represented in three ways, based on its two orthogonal components: (1) all 26 multiclasses (ALLCLASS); (2) only the PROBLEM class sub-set, the *Other* class and the *Spam* class, comprising 8 classes in total (PROBLEM); and (3) only the SOLUTION class sub-set, the *Other* class and the *Spam* class, comprising 6 classes in total (SOLUTION). By combining the outputs of classifiers based on the PROBLEM and SOLUTION class sub-sets (i.e. class composition), it is possible to construct full ALLCLASS classes, and we additionally compare the single-pass multiclass classification strategy with multi-pass class composition.

All experiments were carried out based on stratified 10-fold cross-validation. The results were evaluated via both micro-statistics and macro-statistics. Micro-statistics describe average performance *per instance* (i.e. thread), as represented in the micro-averaged precision ($\mathcal{P}_\mu$), recall ($\mathcal{R}_\mu$) and F-score ($\mathcal{F}_\mu$). Macro-statistics, on the other hand, describe average performance *per class*, as represented in the macro-averaged precision ($\mathcal{P}_M$), recall ($\mathcal{R}_M$) and F-score ($\mathcal{F}_M$). It should be noted that the $\mathcal{P}_\mu$, $\mathcal{R}_\mu$ and $\mathcal{F}_\mu$ are always the same, as the prediction per document is always unique. Moreover, because cross-validation is used, the averaged $\mathcal{F}_M$ is not necessarily the harmonic mean of the averaged $\mathcal{P}_M$ and $\mathcal{R}_M$.

Because we were more interested in the classification effectiveness per thread, the micro-averaged F-score ($\mathcal{F}_\mu$) was used as our primary evaluation method. We also tested the statistical significance of the results using randomised estimation with $p < 0.05$ (Yeh, 2000).

Class Space	Learner	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ALLCLASS	ZEROR	.006	.018	.009	.038
	SVM	.268	**.248**	**.246**	**.382**
	NB	**.306**	.211	.182	.333
PROBLEM	ZEROR	.038	.142	.060	.266
	SVM	.564	**.485**	**.500**	.661
	NB	**.574**	.483	.481	**.691**
SOLUTION	ZEROR	.122	.168	.140	.304
	SVM	.500	**.387**	**.413**	**.575**
	NB	**.513**	.270	.246	.520

Table 2: The performance of different learners over ALLCLASS, PROBLEM and SOLUTION

6 Results and Evaluation

We performed a series of experiments by applying the learners described in Section 5 over the three class sets (i.e. ALLCLASS, PROBLEM and SOLUTION). Because NN performed significantly below the other two learners in all experiments, we only present results from SVM and NB (along with baseline ZEROR). The performance of different learners over ALLCLASS, PROBLEM and SOLUTION is shown in Table 2. For each class space, the best result for each column is presented in **boldface**.

There are several things to note in the results presented in Table 2. First, we can see that the majority class (ZEROR) results are quite poor, especially for ALLCLASS. This is due to the effects of cross-validation, in learning the majority class from the training data in each fold, but due to relative class uniformity, often finding that this is not the majority class in the test data. Second, SVM has relatively strong performance over all three tasks, especially in ALLCLASS and SOLUTION with the best $\mathcal{F}_\mu$ scores. This is not surprising, because it is often reported that SVMs have superior performance in document categorisation tasks (Yang and Liu, 1999; Joachims, 1998). However,

PROBLEM	SOLUTION	ALLCLASS Results			
Learner	Learner	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
SVM	SVM	.345	**.313**	.314	.434
NB	SVM	**.379**	.310	**.316**	**.443**
SVM	NB	.278	.259	.229	.398
NB	NB	.268	.247	.206	.398

Table 3: Results for class composition of the separate predictions from the PROBLEM and SOLUTION classifiers

it is interesting to note that NB produces the best $\mathcal{F}_\mu$ (i.e. 0.691) in the PROBLEM task. Although this figure is not significantly better than the $\mathcal{F}_\mu$ (i.e. 0.661) from SVM, it still may imply that we should optimise our methodology over each subtask.

As is explained in Section 5, the main purpose of the experiments is to examine whether the modular class set (i.e. PROBLEM and SOLUTION as two orthogonal components of the overall ALLCLASS) has the potential to benefit the classification task for ALLCLASS. One simple way is to explore class composition. To be specific, as PROBLEM and SOLUTION represent orthogonal components of ALLCLASS, it is possible to perform classification separately over PROBLEM and SOLUTION, and compose predictions via a combined class set, to form the ALLCLASS class set. For example, if a given thread is predicted to have a PROBLEM class of *Hardware* and a SOLUTION class of *Documentation*, we can compose the two predictions into the *Hardware/Documentation* class. In order to map the results back onto the ALLCLASS class set cleanly, we used the combined class set, where the combination of *Other* from either PROBLEM or SOLUTION with any other class from the second class set produces an overall classification of *Other* in the ALLCLASS set, and the combination of *Spam/Spam* is treated as *Spam*.

The combined results for the ALLCLASS class set are presented in Table 3, with the best outcome for each column once again indicated in **boldface**. From the results we can see that the composition of NB for PROBLEM and SVM for SOLUTION yields the best $\mathcal{F}_M$ (i.e. 0.316) and $\mathcal{F}_\mu$ (i.e. 0.443), significantly improving over the best ALLCLASS results from Table 2 (0.246 and 0.382 respectively). It would therefore appear to be the case that class composition is effective in boosting overall classification performance.

7 Conclusion

This research is aimed towards improving information access over troubleshooting-oriented technical user forum data, focusing on automated thread-level analysis of the problem sources and solution types. As first steps in this direction, we designed a modular thread-level class set, annotated 327 threads, and performed thread classification over the data. We proposed a class composition strategy by first performing classification separately over the PROBLEM and SOLUTION class sets, and composing the predictions into an overall thread classification. This approach gives us the best classification performance overall, with an $\mathcal{F}_\mu$ of 0.443, well above the best result from doing the ALLCLASS classification directly.

Much more work could be done in terms of feature engineering. This could include new features such as author name/profile and the number of posts in the thread. We also speculate that noise in the threads, such as typos and incorrect casing/punctuation, reduced overall performance, suggest that text normalisation may help boost our classifiers. Additionally, because of the promising results produced by the class composition strategy and the innate structure of our thread class set, we could consider more sophisticated hierarchical classification methods (Dekel et al., 2004; Tsochantaridis et al., 2005). We leave these for future work.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

Timothy Baldwin, David Martinez, and Richard B. Penman. 2007. Automatic thread classification for Linux user forum information access. In *Proceedings of the 12th Australasian Document Computing Symposium (ADCS 2007)*, pages 72–79, Melbourne, Australia.

Timothy Baldwin, David Martinez, Richard Penman, Su Nam Kim, Marco Lui, Li Wang, and Andrew MacKinlay. 2010. Intelligent Linux information access by data mining: the ILIAD project. In *Proceedings of the NAACL 2010 Workshop on Computational Linguistics in a World of Social Media: #SocialMedia*, pages 15–16, Los Angeles, USA.

Ofer Dekel, Joseph Keshet, and Yoram Singer. 2004. Large margin hierarchical classification. In *Pro-*

ceedings of the 21st International Conference on Machine Learning (ICML 2004), Banff, Canada.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Chih-Wei Hsu and Chih-Jen Lin. 2006. BSVM. `http://www.csie.ntu.edu.tw/~cjlin/bsvm/`.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, pages 137–142, Chemnitz, Germany.

Marco Lui and Timothy Baldwin. 2009. hydrat. `http://hydrat.googlecode.com`. Retrieved on 25/10/2010.

Andrew Kachites McCallum. 2002. MALLET: A machine learning for language toolkit. `http://mallet.cs.umass.edu/`.

Ani Nenkova and Amit Bagga. 2004. Facilitating email thread access by extractive summary generation. In Nicolas Nicolov, Kalina Bontcheva, Galia Angelova, and Ruslan Mitkov, editors, *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, pages 287–294. John Benjamins, Amsterdam, Netherlands.

Owen Rambow, Lokesh Shrestha, John Chen, and Chirsty Lauridsen. 2004. Summarizing email threads. In *Proceedings of the 4th International Conference on Human Language Technology Research and 5th Annual Meeting of the NAACL (HLT-NAACL 2004)*, pages 105–108, Boston, USA.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(Sep):1453–1484.

Yoshimasa Tsuruoka, Yuka Tateishi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun'ichi Tsujii. 2005. Developing a robust part-of-speech tagger for biomedical text. In *Proceedings of the Advances in Informatics - 10th Panhellenic Conference on Informatics, LNCS 3746*, pages 382–392, Volos, Greece.

Stephen Wan and Kathy McKeown. 2004. Generating overview summaries of ongoing email thread discussions. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 549–555, Geneva, Switzerland.

Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 42–49, Berkeley, USA.

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 947–953, Saarbrücken, Germany.

Liang Zhou and Eduard Hovy. 2005. Digesting Virtual "Geek" Culture: The Summarization of Technical Internet Relay Chats. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 298–305, Ann Arbor, USA.

Dual-Path Phrase-Based Statistical Machine Translation

Susan Howlett and **Mark Dras**
Centre for Language Technology
Macquarie University
Sydney, Australia
{susan.howlett,mark.dras}@mq.edu.au

Abstract

Preceding a phrase-based statistical machine translation (PSMT) system by a syntactically-informed reordering preprocessing step has been shown to improve overall translation performance compared to a baseline PSMT system. However, the improvement is not seen for every sentence. We use a lattice input to a PSMT system in order to translate simultaneously across both original and reordered versions of a sentence, and include a number of confidence features to support the system in choosing on a sentence-by-sentence basis whether to use the reordering process. In German-to-English translation, our best system achieves a BLEU score of 21.39, an improvement of 0.62.

1 Introduction

Machine translation (MT) is the automatic translation of text from one human language to another. Statistical MT accomplishes this through a probabilistic model of the translation process.

In phrase-based statistical MT (PSMT), translation proceeds by dividing a sentence into sequences of adjacent words called *phrases*, then translating each phrase and reordering the phrases according to a *distortion model*. The distortion model may be lexicalised but does not typically incorporate information about the syntactic structure of the sentence. As such, although PSMT has been very successful, it suffers from the lack of a principled mechanism for handling long-distance reordering phenomena due to word order differences between languages.

One method for addressing this difficulty is the *reordering-as-preprocessing* approach, exemplified by Collins et al. (2005) and Xia and McCord (2004), where PSMT is coupled with a preprocessing step that reorders input sentences to more closely parallel the target language word order. Although this leads to improved performance overall, Collins et al. (2005) show that the reordering-as-preprocessing system does not consistently provide better translations than the PSMT baseline on a sentence-by-sentence basis.

One possible reason could be errors in the parse or the consequent reordering. Chiang et al. (2009) used features indicating problematic use of syntax to improve performance within hierarchical and syntax-based translation. In this work, we want to see whether syntax-related features can help choose between original and reordered sentence translations in PSMT.

We use as our starting point the PSMT system Moses (Koehn et al., 2007). In order to use features within the system's log-linear model to assess the reliability of syntax, it is necessary to input both variants simultaneously. To do this, we adapt in a novel way the *lattice input* of Moses; we refer to this new system as *dual-path PSMT* (§3). We then augment the model with a number of *confidence features* to enable it to evaluate which of the two paths is more likely to yield the best translation (§3.2). We reimplement the Collins et al. (2005) reordering preprocessing step and conduct some preliminary experiments in German-to-English translation (§4).

Our results (§5) do not replicate the finding of Collins et al. (2005) that the preprocessing step produces better translation results overall. However, results for our dual-path PSMT system do show an improvement, with our plain system achieving a BLEU score (Papineni et al., 2002) of 21.39, an increase of 0.62 over the baseline. We therefore conclude that a syntactically-informed reordering preprocessing step is inconsistently of use in PSMT, and that enabling the system to choose when to use the reordering leads to improved translation performance.

Susan Howlett and Mark Dras. 2010. Dual-Path Phrase-Based Statistical Machine Translation. In *Proceedings of Australasian Language Technology Association Workshop*, pages 32−40

2 Background and Related Work

2.1 Phrase-based statistical MT

Phrase-based statistical MT (PSMT) systems such as Marcu and Wong (2002) and Koehn et al. (2007) have set the standard for statistical MT for many years. While successful, these systems make limited use of linguistic information about the syntax of the languages which, intuitively, would seem to be useful. Much research is now focused on how to incorporate syntax into statistical MT, for example by using linguistic parse trees on one or both sides of the translation (e.g. Yamada and Knight (2001), Quirk et al. (2005)) or by incorporating only select aspects of syntactic structure, such as recursive structure (Chiang, 2007) or discontinuous phrases (Galley and Manning, 2010).

One area where this lack of syntactic information is felt is the *distortion model* of the PSMT system. This component governs the relative movement of phrases and is the primary means of dealing with word order differences in translation. Common options are distance-based models (where movement is penalised proportionally to the distance moved) and lexicalised models (where probability of movement is conditioned upon the phrase being moved). Without syntactic information here, PSMT systems lack a principled way to manage long-distance movements, leading to difficulty in language pairs where this is needed, such as English and Japanese or German.

2.2 Reordering-as-preprocessing

The *reordering-as-preprocessing* approach addresses the PSMT reordering difficulty by removing word order differences prior to translation. This is done with a preprocessing step where the input sentence is parsed and a reordered alternative created on the basis of the resulting parse tree.

Our work builds on the reordering-as-preprocessing approach of Collins et al. (2005). Working with German-to-English translation, Collins et al. (2005) parse input sentences with a constituent-structure parser and apply six hand-crafted rules to reorder the German text toward English word order. These rules target the placement of non-finite and finite verbs, subjects, particles and negation. The authors demonstrate a statistically significant improvement in BLEU score over the baseline PSMT system.

Many other reordering-as-preprocessing systems exist. Xia and McCord (2004) present a system for French–English translation that, instead of using hand-crafted reordering rules, automatically learns reordering patterns from the corpus. Automatically-acquired rules may be noisier and less intuitive than hand-crafted rules but the approach has the advantage of being more easily extended to new language pairs. Other examples of systems include Wang et al. (2007) (manual rules, Chinese-to-English), Habash (2007) (automatic rules, Arabic-to-English) and Popović and Ney (2006) (manual rules, Spanish/English-to-Spanish/English/German).

Despite the success of the reordering-as-preprocessing approach overall, Collins et al. (2005) found that in a human evaluation on 100 sentences, there were still several cases in which the baseline system translation was preferred over that produced with the reordering. The authors note this finding but do not analyse it further.

2.3 Features for improved translation

Zwarts and Dras (2008) explore the Collins et al. (2005) finding by examining whether machine learning techniques can be used to predict, on a sentence-by-sentence basis, whether the translation of the reordered sentence is to be preferred over the alternative. For features, they use sentence length, parse probability from the Collins parser and unlinked fragment count from the Link Grammar parser on the English side of the translation. The authors find that, when used on the source side (in English-to-Dutch translation), these features provide no significant improvement in BLEU score, while as target-side features (in Dutch-to-English translation) they improve the BLEU score by 1.7 points over and above the 1.3 point improvement from reordering.

Our work has some similarities to that of Zwarts and Dras (2008) but uses the log-linear model of the translation system itself to include features, rather than a separate classifier that does not permit interaction between the confidence features and features used during translation.

This idea of using linguistic features to improve statistical MT has appeared in a number of recent papers. Chiang et al. (2009) demonstrate an improvement in hierarchical PSMT and syntax-based (string-to-tree) statistical MT through the addition of features pinpointing possible errors in the translation, for example the number of occurrences of a particular grammar production rule, or

non-terminal in a rule. Xiong et al. (2010) derive features from the Link Grammar parser, in combination with word posterior probabilities, to detect MT errors (in order to subsequently improve translation quality). Unlike Chiang et al. (2009), we work with PSMT and use features that consider the parse tree as a whole or aspects of the reordering process itself. Unlike Xiong et al. (2010), we use these features directly in translation.

2.4 Reordering lattices

Our work also bears some similarity to recent work using a *reordering lattice* or *reordering forest* as input to the translation system. Examples include Ge (2010), Dyer and Resnik (2010) and Zhang et al. (2007). In these systems, the input structure simultaneously represents many possible reorderings of a sentence, which are produced from a single parse and capture all possible combinations of individual reordering choices.

Like these systems, we use a complex input structure to translate across multiple variations of the sentence simultaneously, and choose between the resulting translations within the translation model itself. However, like Zwarts and Dras (2008), we consider only two possibilities: the original sentence and the sentence with a particular reordering process applied in full.

Our work also differs from these lattice-based systems in that we preserve and incorporate the standard distortion model of the PSMT system in a way that the above systems cannot. Where the lattice-based systems aim to overcome the weaknesses of the distortion model by replacing it with the lattice-creating reordering process, we view the two components as complementary. This has an interesting consequence, which we introduce in §3.1 and discuss further in §5.

Similar to our work and the work in the previous section, Ge (2010) includes features to assess reordering options, based on the structure of the resulting tree, for example which nonterminal appears as the first child and the size of jump required to reach the nonterminal used as the next child. In addition to the difference with the distortion model mentioned above, our work differs in that Ge (2010) focuses on finding the best reordering using syntactic features plus a few surface and POS-tag features as a way of "guarding against parsing errors", whereas we also look at using features to represent confidence in a parse.

3 Dual-Path PSMT

In this paper, we develop a *dual-path* PSMT system. §3.1 introduces the lattice input format, by which we provide the system with two variants of the input sentence: the original and the reordered alternative produced by the preprocessing step. §3.2 outlines the confidence features that we include in the translation model to help the system choose between the two alternatives.

Our system is built upon the PSMT system Moses (Koehn et al., 2007). For reordering, we use the Berkeley parser (Petrov et al., 2006) and the rules given by Collins et al. (2005), but any reordering preprocessing step could equally be used. Further details of our systems are given in §4.

3.1 Word Lattices

A *word lattice* is a structure used to efficiently represent multiple sentences simultaneously. This is of use, for instance, in translating the output of a speech recognition system, where there is some uncertainty about the words of the sentence. The lattice is a directed, acyclic graph with one source and one sink node. Each path from source to sink corresponds to one of the set of sentences being represented. An example is given in Figure 1.

There exists an approximation to the word lattice structure, called a *confusion network*. In this case, every path of the lattice passes through every node, and epsilon transitions are allowed. While a confusion network is in general more space-efficient than a lattice, its paths may include more sentences than the original set (Koehn, 2010).

In the word lattice, each edge is accompanied by a *transition probability*; in speech recognition output this represents the probability that the edge is the correct interpretation of the next part of the signal, with the probabilities on all of the transitions out of a single node summing to one. The probability of one path in the lattice (and therefore one sentence in the set) is the product of the probabilities on the transitions that make up the path. This path probability becomes one feature in the system's translation model.

Moses can be used with word lattice input. Typically, training proceeds as in the baseline case, extracting and scoring phrase pairs from plain parallel sentence data. The lattice structure first appears in tuning, where Moses determines the weight to assign to the transition probability feature, which is then used to translate lattices during evaluation.

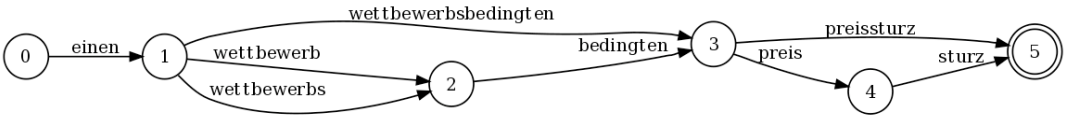


Figure 1: A example of a word lattice for decomposing German compounds, from Koehn (2010).

Our system In our dual-path system, we use a word lattice to simultaneously represent original and reordered versions of an input sentence. Representing only two versions of the sentence, our lattice contains only two paths. We do not wish to introduce any additional paths that are combinations of these two, since doing so sensibly is nontrivial, so we do not use the confusion network approximation.

As mentioned in §3, we use our lattice input in conjunction with the PSMT distortion model. Commonly, Moses is used with a lexicalised distortion model, which is trained along with the phrase table from the training data. In that case, the distortion model for the original sentences will not be appropriate for the reordered sentences, and vice versa. We therefore need separate distortion information for the two paths of the lattice. To accomplish this, we use disjoint vocabularies on the two paths, which we create by prepending every token in the original sentences with `1_` and every token in the reordered sentences with `2_`.

This has two consequences. First, since Moses translates unknown vocabulary items by copying them directly to the output, we must ensure that the prefix is removed before this copying occurs. Second, by introducing disjoint vocabularies for the two paths, we inhibit any sharing of edges between them. This in turn inhibits any sharing of nodes since, like in confusion networks, this would increase the number of paths. Our lattice is therefore degenerate, with the two paths sharing only the source and sink nodes. An example of our lattice input is shown in Figure 2.

This complete separation of the two paths means that extending the dual-path system to a multi-path system is likely to rapidly encounter efficiency issues. It is, however, a possibility; we discuss a number of options in §5.

Finally, when specifying the lattice, nodes must be given in topographical order. We number the nodes of each path alternately since for efficiency reasons Moses imposes a limit on the length of edges in the lattice.

Note about training As in the typical case, our lattice is used only during tuning and evaluation. For training, we construct a parallel corpus twice the size of the original corpus, containing the original sentences (prefixed with `1_`) plus their translations, and the reordered versions of these sentences (prefixed with `2_`) plus their translations (identical to those of the first half). Training takes place on this one corpus, giving one phrase table containing translation candidates for both paths.

Using one vocabulary The need for disjoint vocabularies is a consequence of the lexicalised distortion model; with a simple distance-based model this would not be necessary. In that case, we could construct a lattice identical to Figure 2 but with the `1_`/`2_` prefixes removed (which may then allow some compression of the lattice), and then proceed as in the disjoint-vocabulary system. In this case, it may also be necessary to create separate phrase tables from the two halves of the training data.

3.2 Confidence Features

When translating lattice input, Moses creates translation candidates for all (in our case, both) paths in the lattice, and the ultimate translation will be one option from one path. The probability of a path (obtained by multiplying the probabilities of its component edges) is used as a feature for every translation candidate for that path. We can therefore use the transition probabilities to influence the system to choose the best translation of one path over the best translation of the other path.

Usefully, Moses treats the transition probabilities like any other feature, so in fact the values on the transitions need not fall in the range $[0, 1]$ and the values on transitions out of a single node need not sum to one. Further, Moses allows multiple such features, so each edge in the lattice may be associated with n values and thus each path may have n scores, each produced by multiplying the corresponding values on its component edges.[1]

[1]Moses actually operates with log-probabilities. After the lattice is read in, a log transformation is applied to every value

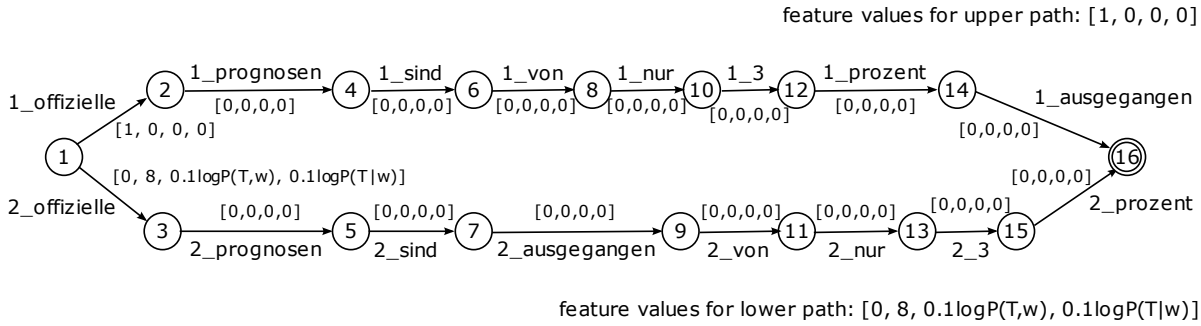


Figure 2: Example input to our dual-path system, illustrating disjoint vocabularies, node ordering and four feature values (both per edge and combined for path totals), for the sentence *Offizielle Prognosen sind von nur 3 Prozent ausgegangen* ('Official forecasts predicted just 3 percent'), adapted from the tuning data.

We first outline the features that we use, and then discuss how they are incorporated into the translation model.

3.2.1 Feature set

Reordering indicator feature This feature is intended to differentiate the two paths of the lattice. Its function is described more fully in §3.2.2.

Sentence length Intuitively, longer sentences are more likely to be poorly parsed or poorly reordered, so as a feature we include the number of tokens in the original sentence.

Parse probability (2 features) The reordering is based on an automatically-produced parse. A low parse probability may indicate that the sentence was difficult to parse and the reordering may therefore be unreliable. To capture this, we include features based on the joint and conditional probabilities of the parse tree: $P(T, w)$ and $P(T|w)$. Both are included as it is not obvious whether one or the other will be the more useful.

The Berkeley parser can provide both $\log P(T, w)$ and $\log P(T|w)$ with the parse of the sentence. To avoid underflow issues, we actually use $0.1 \times \log P(T, w)$ and $0.1 \times \log P(T|w)$.

Rule application counts (6 features) We include counts of the number of times each of the six Collins et al. (2005) reordering rules was used for the sentence, since particular rules may be more likely to cause problems in the reordering. We anticipate this feature would be more useful in a system with automatically-acquired reordering rules.

on every edge. In the discussion that follows and in Figure 2 we specify the feature values as those obtained after the log transformation.

POS counts (4 features) These features count the number of nouns, verbs, prepositions and conjunctions in the sentence. Higher counts may indicate more complex sentences that require more reordering or are more likely to be poorly reordered.

Jump sizes (10 features) Each time a reordering rule fires, a sequence of a tokens is moved past a sequence of b tokens. We call this a jump of size $\max(a, b)$ (as it may be either a sequence of a tokens jumping over a sequence of b tokens, or vice versa). These jump sizes are binned, and the number of jumps observed in each bin for the sentence becomes one feature in the model.

To choose the bins, we examined the jumps that occur when reordering the smaller of our parallel training corpora.[2] (Recall that the lattices and therefore these confidence features do not appear in the system until the tuning stage.) We observed jumps from 1 to 94 tokens in size, with the number of jumps observed decreasing roughly exponentially as the size of the jump increased.

Although the reordering process is intended to produce long jumps, we expect that jumps above a certain size are more likely to be due to erroneous bracketing in the parse. However, we are not sure at which point a jump becomes too large, and we may wish to apply progressively stronger penalties for longer jumps. We therefore prefer a larger number of narrower bins over fewer, wider bins. We chose the bins listed in Table 1.

Table 1 also lists the number of times a jump of this size was seen in the training set, and the number of times we therefore expect to see a jump of this size in the tuning set, where the weights for each bin feature will be set.

[2] news-commentary10.de-en.de

Bin	# in training	Expected # in tuning
1–2	104,296	2,133
3–5	89,383	1,828
6–8	46,695	955
9–12	27,484	562
13–16	10,935	234
17–20	4,739	97
21–25	2,610	53
26–30	1,115	23
31–35	498	10
36+	455	9

Table 1: Jumps seen in the training set of 100,269 sentence pairs, and the expected number for each bin in the tuning set of 2,051 sentence pairs, if the two corpora have a similar distribution.

3.2.2 Inclusion in the translation model

Recall that our input lattice consists of two disjoint paths. The only choice occurs at the first node of the lattice, where there are two options; all other nodes have precisely one outgoing transition.

Of the two transitions out of the first node of the lattice, one is for the original sentence and the other for the reordered variant. On the original sentence transition, we put the value 1 for the indicator feature and 0 for the remaining 23 confidence features. On the reordered sentence transition, we put 0 for the indicator feature and for all other features the value as specified in §3.2.1.

For all the other transitions in the lattice, where there is no branching, we put the value 0 for all 24 confidence features. This means that these edges do not contribute to the scores for each path, and so the overall feature values for each path are the same as those of its first edge. The numbers beside each edge in Figure 2 demonstrate the result with the first four features. Note that all translation candidates for a given path will share the same feature values; the features distinguish between paths, not between candidates from one path.

Moses uses a maximum entropy model to score translation candidates. As such, the final score of a translation candidate is the weighted sum of all of the feature values of the translation:

$$\text{score} = \exp \sum_{i=1}^{n} \lambda_i h_i(e, f) \qquad (1)$$

This automatically includes the lattice transition features. Assume without loss of generality that the confidence features above are features $1, \ldots, m$, with feature 1 being the reordering indicator feature. Therefore the part of this sum that is due to the confidence features will be $\sum_{i=1}^{m} \lambda_i h_i(e, f)$. This will simplify to λ_1 for candidate translations of the original sentence and $\sum_{i=2}^{m} \lambda_i h_i(e, f)$ for candidate translations of the reordered sentence. Hence we expect the features excluding the indicator feature to collectively indicate the extent to which the reordering may be trusted, with λ_1 controlling the general preference to use the original or the reordered sentence.[3]

The current version of Moses uses minimum error rate training (MERT) to set the weights λ_i. This procedure is limited in the number of features that it can efficiently process. Further extending the feature set would require a different optimisation procedure, such as MIRA, as discussed by Arun and Koehn (2007).

4 Experiments

4.1 Models and Evaluation

Our baseline PSMT system is Moses (Koehn et al., 2007), repository revision 3590.[4] We run all of our experiments using the Moses Experiment Management System; configuration files and scripts to reproduce our experiments are available online.[5]

For the reordering preprocessing step we reimplement the Collins et al. (2005) rules and use this to recreate the Collins et al. (2005) reordering-as-preprocessing system as our second baseline.

We use the Berkeley parser (Petrov et al., 2006), repository revision 14,[6] to provide the parse trees for the reordering process. Since the German parsing model provided on the parser website does not include the function labels needed by the Collins et al. (2005) rules, we trained a new parsing model on the Tiger corpus (version 1). The reordering script and parsing model, along with details of how the parsing model was trained, are available online with the configuration files above.

We compare four systems on German-to-English translation: the Moses baseline (MOSES), the Collins et al. (2005) baseline (REORDER), the lattice system with just the reordering indicator feature (LATTICE), and the lattice system with all

[3] It is possible that in practice the imbalance in number of non-zero features between the two paths could cause the system some difficulty in assigning the weights for each feature. In future it would be interesting to investigate this possibility by introducing extra features to balance the two paths.

[4] https://mosesdecoder.svn.sourceforge.net/svnroot/mosesdecoder/trunk/

[5] http://www.showlett.id.au/. Some minor changes to Moses were implemented to work with our job scheduling software; full details are also available here.

[6] http://berkeleyparser.googlecode.com/svn/trunk/

Purpose	File set	# Sents
Training	europarl-v5.de-en	1,540,549
	news-commentary10.de-en	100,269
Tuning	news-test2008	2,051
Test	newstest2009	2,525
LM	europarl-v5.en	1,843,035
	news-commentary10.en	125,879
	news.en.shuffled	48,653,884

Table 2: Corpora used in our experiments and their sizes. The first four are parallel corpora (size: number of sentence pairs); the last three are monolingual corpora for the language model (size: number of sentences).

confidence features (+FEATURES). We do not explore different subsets of the features here.

For evaluation we use the standard BLEU metric (Papineni et al., 2002), which measures n-gram overlap between the candidate translation and the given reference translation.

4.2 Approximate Oracle

To get an idea of a rough upper bound, we implement the approximate oracle outlined in Zwarts and Dras (2008). For every sentence, the approximate oracle compares the outputs of MOSES and REORDER with the reference translation and chooses the output that it expects will contribute to a higher BLEU score overall. The oracle chooses the candidate that shares the highest number of 4-grams with the reference; if the two have the same 4-gram overlap, it chooses the one that shares the highest number of trigrams with the reference, and so on down to unigrams. If still identical, it chooses the original.

Note that the output of the oracle for some sentence will be identical to the output of one or the other baseline system; for the lattice system this is not necessarily the case since the system is tuned separately with a different number of features.

4.3 Data

For data we use the corpora provided for the 2010 Workshop on Statistical Machine Translation[7] translation task. The number of sentence pairs in each corpus are given in Table 2.

We trained 5-gram language models with SRILM (Stolcke, 2002) using the three language model files listed in Table 2. For convenience, the news.en.shuffled corpus was split into eight smaller files, seven containing 6,100,000 lines and the last containing the remainder. One language model was produced for each file or subfile, giving a total of ten models. The final language model was produced by interpolation between these ten, with weights assigned based on the tuning corpus.

[7] http://www.statmt.org/wmt10/

System	BLEU
MOSES	20.77
REORDER	20.04
Approx oracle	22.45
LATTICE	21.39
+FEATURES	21.10

Table 3: BLEU scores for every system

5 Results and Discussion

Table 3 gives the BLEU score for each of our four systems and the approximate oracle. We note that these numbers are lower than those reported by Collins et al. (2005). However, this is most likely due to differences in the training and testing data; our results are roughly in line with the numbers reported in the Euromatrix project for this test set.[8]

Interestingly, our reimplementation of the Collins et al. (2005) baseline does not outperform the plain PSMT baseline. Possible explanations include variability due to differences in training data, noisier parser output in our system, or differing interpretation of the description of the reordering rules. It may also be that the inconsistency of improvement noted by Collins et al. (2005) is the cause; sometimes the reordering produces better results and sometimes the baseline, with the effect just by chance favouring the baseline here. To explore this, we look at the approximate oracle.

In our experiment, the oracle preferred the baseline output in 848 cases and the reordered in 1,070 cases. 215 sentences were identical between the two systems, while in 392 cases the sentences differed but had equal numbers of n-gram overlaps. The BLEU score for the oracle is higher than that of both baselines; from this and the distribution of the oracle's choices, we conclude that the difference between our findings and those of Collins et al. (2005) is at least partly due to the inconsistency that they identified. It is especially interesting to note that the reordered system's translations are preferred by the oracle more often even though its overall performance is lower.

Turning now to the results of our systems, we see that simply giving the system the option of

[8] http://matrix.statmt.org/

both reordered and non-reordered versions of the sentence (LATTICE) produces an improved translation performance overall. While the addition of our confidence features (+FEATURES) leaves the performance roughly unchanged, the gap between LATTICE and the approximate oracle implies that this is due to the choice of features, and that a feature set may yet be found that will improve performance over the plain LATTICE system.

In light of the Zwarts and Dras (2008) finding that source-side features in the classifier do not help translation performance, our negative result with +FEATURES may appear unsurprising, as all of our features may be classified as source-side. However, we note that there remains a considerable feature space to explore. Note that if we were to include the equivalent of their target-side features into our system, they would appear as language model features, rather than features on the input lattice. Thus it seems that in fact Zwarts and Dras (2008) address two distinct problems—adding syntactic information to the translation process, and adding it to the language model.

5.1 Extensions and Future Work

Given our dual-path system, an obvious question is whether a multi-path extension is possible. Since the disjoint vocabularies inhibit compression of the lattice, extending the input to a multi-path lattice is likely to rapidly encounter efficiency issues. However, some possibilities exist.

One option would be to include paths that represent different reorderings of the sentence based on the same parse. For example, the original sentence could be compared with reorderings created by different reordering-as-preprocessing approaches. In this instance, it would be advisable to use a different vocabulary (by using a different token prefix) for each path, as each reordering is likely to require a different lexicalised distortion model.

In the case where these reordered alternatives are all possible combinations of parts of one reordering process, our system approaches the work described in §2.4, and in fact those systems will probably be more suitable as the preprocessing takes over the role of the PSMT distortion model.

Alternatively, the multiple options could be created by the same preprocessor but based on different parses, say the n best parses returned by one parser, or the output of n different parsers with comparable outputs. This extension would be quite different from the lattice-based systems in §2.4, which are all based on a single parse.

For future systems, we would like to replace the Collins et al. (2005) reordering rules with a set of automatically-extracted reordering rules (as in Xia and McCord (2004)) so that we may more easily explore the usefulness of our system and confidence features in new language pairs with a variety of reordering requirements.

The next major phase of this work is to extend and explore the feature space. This entails examining subsets of confidence features to establish which are the most useful indicators of reliable reordering, and possibly replacing the MERT tuning process with another algorithm, such as MIRA, to handle a greater quantity of features. In addition, we wish to explore more fully our negative result with the reimplementation of the Collins et al. (2005) system, to investigate the effect of balancing features in the lattice, and to examine the variability of the BLEU scores for each system.

6 Conclusion

We adapt the lattice input to the PSMT system Moses to create a system that can simultaneously translate a sentence and its reordered variant produced by a syntactically-informed preprocessing step. We find that providing the system with this choice results in improved translation performance, achieving a BLEU score of 21.39, 0.62 higher than the baseline.

We then augment the translation model of our system with a number of features to express our confidence in the reordering. While these features do not yield further improvement, a rough upper bound provided by our approximate oracle suggests that other features may still be found to guide the system in choosing whether or not to use the syntactically-informed reordering.

While our reordering step is a reimplementation of the Collins et al. (2005) system, contrary to their findings we do not see an improvement using the reordering step alone. This provides evidence against the idea that reordering improves translation performance absolutely. However, our success with the lattice system highlights the fact that it *is* useful for some sentences, and that syntactic confidence features may provide a mechanism for identifying which sentences, thus incorporating syntactic information into phrase-based statistical machine translation in a useful way.

Acknowledgements

We would like to thank the anonymous reviewers, Chris Dyer and Mark Johnson for their helpful comments, and Christian Hardmeier and Simon Zwarts for implementation assistance.

References

Abhishek Arun and Philipp Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. In *Proceedings of the MT Summit XI*, pages 15–20.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 218–226.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531–540.

Chris Dyer and Philip Resnik. 2010. Context-free reordering, finite-state translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 858–866.

Michel Galley and Christopher D. Manning. 2010. Accurate non-hierarchical phrase-based translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 966–974.

Niyu Ge. 2010. A direct syntax-driven reordering model for phrase-based machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 849–857.

Nizar Habash. 2007. Syntactic preprocessing for statistical machine translation. In *Proceedings of the MT Summit XI*, pages 215–222.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180.

Philipp Koehn, 2010. *Moses: Statistical Machine Translation System*. University of Edinburgh. http://www.statmt.org/moses/manual/manual.pdf Downloaded 13 July 2010.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–139.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440.

Maja Popović and Hermann Ney. 2006. POS-based word reorderings for statistical machine translation. In *Proceedings of LREC 2006*, pages 1278–1283.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 271–279.

Andreas Stolcke. 2002. Srilm — an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 901–904.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 737–745.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 508–514.

Deyi Xiong, Min Zhang, and Haizhou Li. 2010. Error detection for statistical machine translation using linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 604–611.

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*, pages 523–530.

Yuqi Zhang, Richard Zens, and Hermann Ney. 2007. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 1–8.

Simon Zwarts and Mark Dras. 2008. Choosing the right translation: A syntactically informed classification approach. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1153–1160.

Information Extraction of Multiple Categories from Pathology Reports

Yue Li
NICTA VRL
Melbourne, Australia
y.li30@ugrad.unimelb.edu.au

David Martinez
NICTA VRL
Melbourne, Australia
davidm@csse.unimelb.edu.au

Abstract

Pathology reports are used to store information about cells and tissues of a patient, and they are crucial to monitor the health of individuals and population groups. In this work we present an evaluation of supervised text classification models for the prediction of relevant categories in pathology reports. Our aim is to integrate automatic classifiers to improve the current workflow of medical experts, and we implement and evaluate different machine learning approaches for a large number of categories. Our results show that we are able to predict nominal categories with high average f-score (81.3%), and we can improve over the majority class baseline by relying on *Naive Bayes* and feature selection. We also find that the classification of numeric categories is harder, and deeper analysis would be required to predict these labels.

1 Introduction

A pathology report is the summary of the analysis of cells and tissues under a microscope, and it may also contain information of the studied specimen as it looks to the naked eye. Pathology reports play an important role in cancer diagnosis and staging (describing the extent of cancer within the body, especially whether it has spread). These reports are usually written by the pathologist in natural language, and then the relevant parts are transcribed into structured form by a different person to be stored in a database.

The use of structured information can help share the data between institutions, and can also be used to find patterns in the data. For this reason, some recent initiatives are exploring better ways to manage pathology reports. For instance, the Department of Health and Ageing of Australia is funding the project Structured Pathology Reporting of Cancer since 2008 to develop standard reporting protocols for cancer reports[1]. Another way to promote the creation of structured data is to use standard terminologies, such as SNOMED CT[2], which is a large collection of medical terminology covering most areas of clinical information such as diseases, findings, procedures, microorganisms, pharmaceuticals etc. The National E-Health Transition Authority (NEHTA) has recently launched an adapted terminology (SNOMED CT-AU) to be used by the Australian health sector[3].

These initiatives will help to increase the repositories of structured data, but they will not be a substitute to the flexibility of natural language. The relevant fields in structured reports change over time as different clinical tests are made available, and it is difficult to design a specific form to cover all the possible cases that will be observed in the pathology analysis. Clinicians need time to learn the different standards, and they prefer the flexibility of free text to record their analyses and conclusions. Ideally their natural language input would be used to automatically extract the structured data that different protocols demand.

This scenario is promising for text mining research, because tools that can perform well in this space are likely to make an impact in the way health information is stored and used. Our goal in this work is to explore this area, and develop and evaluate a text mining tool that aims to work in a real hospital setting, by predicting pieces of information to populate a database. Specifically, we focus on a system for the Royal Melbourne Hospital, where pathology reports of cancer patients are

[1] http://www.rcpa.edu.au/Publications/StructuredReporting.htm

[2] http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html

[3] http://www.nehta.gov.au/media-centre/nehta-news/571-snomed-ct

Yue Li and David Martinez. 2010. Information Extraction of Multiple Categories from Pathology Reports. In *Proceedings of Australasian Language Technology Association Workshop*, pages 41−48

kept in natural language, and an electronic form is manually filled with the most relevant information. We would like to predict the classes automatically in order to facilitate the process.

Our aim is to build a generic approach for different prediction categories, involving heterogeneous classes with a large set of possible values (e.g. the class "Tumour site" has 11 different values in our data, for instance "Sigmoid Colon"). We will rely on the available document-level annotations of pathology reports to build our classifiers using Machine Learning (ML) algorithms. Annotated data is difficult to obtain in this domain, and there are few works evaluating the performance of supervised classifiers for pathology reports, as we will see in Section 2. In this work we will explore how far we can get with existing annotations, and simple lexical features that can be extracted without external knowledge sources.

Thus, we present an extensive set of experiments to evaluate the ability of different models and methods to perform class-predictions over pathology reports. The problem will involve predicting nominal and numeric classes, and we test models that perform sentence-level and document-level classification. Our main challenges in this project will be the sparseness of the data, the coarseness of the annotations (document-level categories only), and the high number of heterogeneous categories. In the future, the tool resulting from this work will be integrated in the hospital workflow, and it will work interactively with the user, making predictions and allowing corrections. We will store all user interactions to continually add training data to our classifiers. We will also highlight the relevant parts of the text as predicted by our learning models, by using feature selection.

2 Related work

Related work in text mining from pathology reports has mainly relied on domain-specific lexicons and rules (Dunham et al., 1978; Schadow and McDonald, 2003; Xu et al., 2004; Hanauer et al., 2007; Coden et al., 2009; Nguyen et al., 2010); although there has been some work using ML (Nguyen et al., 2007; McCowan et al., 2007). The earliest work in this area was perfomed by Dunham et al. (1978), who built morphosyntactic rules, synonym expansion, and handcrafted rules in order to extract terms from the Systematized Nomenclature of Pathology (SNOP), which was an earlier version of the SNOMED CT terminology collection. More recent works have used SNOMED CT as the target terminology to map the raw text into. Hanauer et al. (2007) relied on custom-made lists containing approximately 2,500 terms and phrases, and 800 SNOMED codes. Their method was based on looking up relevant phrases in order to discriminate the documents of interest.

Other works have developed their own set of relevant classes instead of relying on SNOMED. This is interesting when the focus is on a specific subdomain, and this is the approach that we explored in our work. Schadow and McDonald (2003) relied on a subset of UMLS[4] (Unified Medical Language System) as target concept inventory for information extraction from surgical pathology reports. They applied a regular expression-based parser with good performance, but they also found that their target terminology was too extensive, and this caused false positives. Xu et al. (2004) also targeted surgical pathology reports, and they used a restricted set of 12 classes, referred as "types of findings". This is similar to our approach, and some of their classes are part of our relevant classes as well (e.g. "number of positive nodes"); however they do not provide the performance for each class separately, which makes comparison unfeasible. Regarding the methodology, their system is based on hand-crafted rules, and relies on a domain-specific lexicon. Our motivation is different, and we rely on ML to infer the knowledge from coarse-grained annotation for a larger set of classes.

Also in the area of information extraction from pathology reports, recent work from the Australian e-Health Research Centre[5] explored the extraction of staging information of lung cancer using Support Vector Machines (Nguyen et al., 2007). Their initial experiments showed the difficulty of the primary tumour stage detection (T), with a top accuracy of 64%. In a follow-up paper they explored richer annotation, and a combination of ML and rule-based post-processing (McCowan et al., 2007). They performed fine-grained annotation of stage details for each sentence in order to build their system, and they observed improvements over a coarse-grained (document-level) multiclass classifier. However, the authors explain that the

[4] http://www.nlm.nih.gov/research/umls

[5] http://aehrc.com/

annotation cost is high, and in their latest work they rely heavily on the SNOMED CT concepts and relationships to identify the relevant entities (Nguyen et al., 2010). They argue that this approach is more portable than fine-grained annotation, although it still requires involvement from the experts, and there is a loss in accuracy with respect to their best ML approach. These three papers evaluate their system in the prediction of staging classes (T, N, and M), which are not explicit in our dataset.

Another relevant work on this area was conducted by Coden et al. (2009), where the authors defined an extensive knowledge model for pathology reports. Their model was linked to hand-built inference rules built to process unseen data. They reported high performance over 9 target classes for a hand-annotated 300-report dataset. This system seeks to build a strong representation of the domain by relying on human experts, and its portability to a different dataset or class-set could be problematic. The classes they evaluate on are not present in our dataset.

Currently there is no dataset of pathology reports that is freely available for research, and different groups have built their own corpora. Pathology reports contain sensitive material, and even after de-identification it is not easy to make them widely available. However, initiatives as the NLP challenges leaded by the Informatics for Integrating Biology and the Bedside (i2b2)[6] illustrate that there is growing interest on text mining from clinical data, and show that the research community can collaboratively create corpora for experimentation. In 2010 they organised their fourth challenge, focused on the extraction of medical problems, tests, and treatments from patient discharge summaries[7]. Previous challenges have also focused on discharge summaries and narrative patient records for different information extraction categories. Although this data is different to pathology reports, the initiative is interesting for the future of text mining from pathology reports.

3 Experimental setting

In this section we first describe the dataset and categories we will work on, and then introduce the models and classifiers we applied. Finally we explain our feature set, and our evaluation methodology.

[6] i2b2 is a NIH-funded National Center for Biomedical Computing (NCBC), for more information see https://www.i2b2.org/about/index.html

[7] https://www.i2b2.org/NLP/Relations/

Category	Unique Values	Highest	Lowest
CAV	21	40	0
Distal Distance	48	150	0
Nodes Examined	36	73	0
Nodes Positive	12	15	0
Polyps Number	13	43	0
Radial Distance	9	80	0
Tumour Length	36	110	0
Tumour depth	22	40	0
Tumour width	35	75	0

Table 1: List of numeric categories, with the number of unique values and the full range.

3.1 Dataset

For our analysis we rely on a corpus of 203 de-identified clinical records from the Royal Melbourne Hospital. These records were first written in natural language, and then structured information about 36 fields of interest was introduced to the Colorectal Cancer Database of the hospital. The written records tend to be brief, usually covering a single page, and semantically dense. Each report contains three sections describing different parts of the intervention: macroscopic description, microscopic description, and diagnosis. All sections contain relevant information for the database.

There are two types of fields (which will be the target categories of our work), depending on the type of values they take: numeric and nominal. Numeric categories are those that take only numeric values, and they are listed in Table 1. We also show the number of different values they can take, and their value range. We can see that most categories exhibit a large number of unique values. The remaining 27 categories are nominal, and the list is shown in Table 2, where we also provide the number of unique values, and the most frequent value in the corpus for each category. Some of the categories are linked to a large number of values (e.g. *Colon Adherent To* and *Tumour Site*). During pre-processing we observed that the database had some inconsistencies, and a normalisation step was required with collaboration of the experts. For nominal categories this involved mapping empty values, "0", and "?" into the class "N/A"; and for numeric categories we mapped empty values into "zero".

The manual annotation is provided at document

Category	Unique Values	Most Frequent
Anastomosis Method	3	*Staple*
Anastomosis Type	4	*End-End*
Biopsy Confirmed Mata	3	*No*
Colon Adherent	3	*No*
Colon Adherent To	14	*N/A*
Differentiation	8	*Moderate*
Inflammatory Infiltrate	4	*Not reported*
Liver	3	*N/A*
Lympho Invasion	4	*No*
MLH1	4	*Not done*
MSH2	4	*Not done*
MSH6	4	*Not done*
MSI	4	*Not done*
Margins Distal	3	*Not involved*
Margins Radial	4	*N/A*
Microscopic Type	5	*Adenocarcinoma*
Mucinous	4	*Not reported*
Necrosis	4	*Not reported*
Other Meta	3	*N/A*
Pathologic Response	4	*N/A*
Peritoneal	3	*N/A*
Polyps	3	*No*
Polyps Type	6	*N/A*
Primary Tumour Rectum	4	*N/A*
Ressected Meta	3	*No*
Staging ACPS	6	*B*
Tumour Site	11	*Sigmoid Colon*

Table 2: List of nominal categories, with the number of unique values, and most frequent class.

level, and for numeric categories we automatically produce fine-grained annotation by looking up the goldstandard mentions in the text. We try to match both the string representation and the numbers, and only numbers different to zero are identified. After this automatic process, each sentence has individual annotations for each of the target categories, and this information is used to build sentence-level classifiers. Because the process is automatic, some matches will be missed, but our hypothesis is that the noisy annotation will be useful for the document-level evaluation.

3.2 Models

Our goal is to build document classifiers for each of the 36 categories with minimal hand tuning. We follow different strategies for nominal and numeric categories. For nominal categories we observed that the information can be given at different points in the document, and we decided to build a multiclass classifier for each category. This method makes a single prediction based on the class annotations in training data.

For numeric categories the information tends to be contained in a single sentence, and instead of using the full document, we relied on the sentence-level annotation that we obtained automatically. In this case the target values would be the different numeric values seen in the goldstandard. The first step is to build sentence classifiers for each class, by using the sentence-level annotations. Note that only numbers different to zero are detected, and the zero label is assigned only in cases where the sentence classifiers fail to identify any number. After the model identifies the positive sentences, the numeric values are extracted, and the number closest to the median of the class (in training data) is assigned. In the cases where no positive sentences are identified the number zero is assigned.

3.3 Classifiers

Each of our models is tested with a suite of classifiers provided by the Weka toolkit (Witten and Frank, 2005). We chose a set of classifiers that has been widely used in the text mining literature in order to compare their performances over our dataset:

- Naive Bayes (*Naive Bayes*): A simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) to obtain the conditional probability of each class given the features in the context. It assumes independence of the features, which in real cases can be a strong (naive) assumption.

- Support Vector Machines (*SVM*): They map feature vectors into a high-dimensional space and construct a classifier by searching for the hyperplane in that space that gives the greatest separation between the classes.

- AdaBoost (*AdaBoost*): This is a meta-learning algorithm where an underlaying classifier is used to update a distribution of weights that indicates the importance of the training examples. Adaboost is an adaptive algorithm, and the prediction hits and misses in each iteration are used to build the final weight distribution for the model.

We use the default parameter settings of Weka (version 3-6-2) for each of the classifiers. As underlaying classifier for *AdaBoost* we rely on simple Decision Stumps (one-level decision trees).

We also explore the contribution of feature selection to the classification performance. We apply a correlation-based feature subset selection method, which considers the individual predictive

Category	*Majority Class*			*Naive Bayes*			*SVM*			*AdaBoost*		
	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc
Tumour site	3.9	19.7	6.5	23.3	40.4	27.7	28.5	38.4	**32.2**	12.4	34.0	18.1
Staging ACPS	12.9	36.0	19.0	39.2	43.8	36.7	44.1	48.3	**45.1**	33.2	49.3	37.5
Anastomosis type	22.4	47.3	30.4	46.6	52.7	44.9	53.5	59.1	**55.0**	32.7	50.2	39.3
Colon adherent	23.3	48.3	31.4	63.2	67.0	62.0	67.4	70.4	**67.0**	47.8	58.6	51.4
Lympho invasion	23.8	48.8	32.0	48.4	51.2	42.9	53.4	55.7	**53.5**	32.3	51.2	39.6
Polyps	27.8	52.7	36.4	69.6	72.9	69.3	83.1	84.2	**83.3**	74.9	77.8	74.3
Colon adherent to	28.3	53.2	37.0	59.9	70.0	63.9	62.6	73.4	**67.4**	40.6	47.3	43.7
Margins radial	31.0	55.7	39.8	65.4	73.4	**68.6**	65.2	70.9	67.4	59.9	67.5	62.0
MlH1	31.5	56.2	40.4	41.4	50.7	45.4	46.7	49.8	**46.9**	35.6	58.1	43.4
MSH6	31.5	56.2	40.4	41.5	50.7	45.5	43.0	48.8	**45.7**	35.0	57.6	42.7
MSH2	31.5	56.2	40.4	41.5	50.7	45.5	43.0	48.8	**45.7**	35.0	57.6	42.7
MSI	32.7	57.1	41.6	44.5	53.7	**48.4**	45.8	50.7	47.9	32.7	57.1	41.6
Mucinous	34.9	59.1	43.9	42.6	58.6	46.7	70.2	72.4	**68.8**	57.8	73.4	64.5
Anastomosis method	41.6	64.5	50.6	57.6	70.4	62.0	62.4	71.9	**65.8**	56.9	69.5	60.4
Necrosis	42.9	65.5	51.9	64.5	74.4	68.7	73.5	77.3	**74.6**	62.8	69.5	62.7
Polyps type	49.6	70.4	58.2	49.6	70.4	58.2	66.9	72.9	**63.4**	49.6	70.4	58.2
Differentiation	51.0	71.4	59.5	51.2	70.9	59.5	70.3	78.3	72.1	66.0	80.8	**72.7**
Inflammatory infiltrate	51.0	71.4	59.5	66.5	72.4	62.6	68.4	76.8	**70.9**	68.9	70.9	66.2
Liver	53.2	72.9	61.5	53.0	68.0	59.5	59.5	64.5	**61.7**	52.8	70.9	60.5
Other meta	53.2	72.9	61.5	53.0	68.0	59.5	59.5	64.5	**61.7**	53.1	71.4	60.9
Primary tumour rectum	53.2	72.9	61.5	56.5	72.4	62.7	70.9	80.3	**75.1**	67.5	73.9	70.4
Peritoneal	53.2	72.9	61.5	53.0	68.0	59.5	59.5	64.5	**61.7**	52.7	70.4	60.3
Ressected meta	63.7	79.8	70.8	68.6	79.8	72.1	70.4	79.3	**73.2**	66.2	77.8	70.8
Margins distal	76.0	87.2	81.2	76.0	87.2	81.2	86.1	92.1	**89.0**	86.1	92.1	**89.0**
Biopsy confirmed mata	80.4	89.7	84.8	80.4	89.7	84.8	85.0	89.7	**86.5**	80.3	88.7	84.3
Pathologic response	81.3	90.1	**85.5**	81.3	90.1	**85.5**	81.3	90.1	**85.5**	81.3	90.1	**85.5**
Microscopic type	87.6	93.6	**90.5**	87.6	93.6	**90.5**	88.0	93.1	**90.5**	87.6	93.6	**90.5**
Macro-average	43.5	63.8	51.0	56.5	67.1	59.8	63.3	69.1	**65.1**	54.1	67.8	59.0

Table 3: Performances of multiclass document classifiers for nominal categories without feature selection. Results sorted by baseline f-score performance. Best f-score per category is given in bold.

ability of each feature and the redundancy of each subset (Hall, 1999). We relied on Weka's implementation of this technique, and used Best-First search, with a cache-size of one element, and 5 levels of backtracking.

3.4 Features

Pathology reports tend to be short and dense, and the selection of words tries to precisely specify the relevant pieces of information. For this reason we rely on a bag-of-words (BOW) approach for our feature representation, without any lemmatisation. We built a simple tokeniser based on regular expressions to separate words, numbers, and punctuation. We also use regular expressions to convert the textual mentions of numbers into their numeric representation. Finally, we include the binary feature "NUMBER" to indicate whether there is a numeric reference in the text.

3.5 Evaluation

In order to evaluate the different models and classifiers we use precision, recall, and f-score by micro-averaging the results over the different class values. The macro-averaged scores over all categories are also provided to compare different systems. 10-fold cross-validation is used in all our experiments.

As a baseline we rely on the *Majority Class* classifier, which assigns the most frequent class from training data to all test instances. In case of ties the value is chosen randomly among those tied.

4 Results

We first present our result over the nominal categories, and then show the performances over numeric categories.

4.1 Nominal categories

Our first experiment applies the multiclass document classifier to nominal categories. The results are given in Table 3. We can see that the best performance is achieved by *SVM*, with a large improvement over the majority class baseline. *Naive Bayes* and *AdaBoost* also perform above the baseline, and attain similar results. However, a maximum f-score of 65.1% seems insufficient to be of use for an application. Regarding the different categories, as expected these with lowest baseline

Category	*Naive Bayes*			*SVM*		
	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc
Tumour site	53.0	54.2	**51.1**	43.9	44.8	43.8
Staging ACPS	71.1	71.9	**70.7**	58.0	59.1	58.3
Anastomosis type	74.1	71.9	**71.3**	69.4	69.5	69.4
MSH6	75.7	70.9	**71.4**	65.3	65.0	64.7
MSH2	75.7	70.9	**71.4**	65.3	65.0	64.7
Colon adherent to	68.9	74.9	**71.6**	64.4	70.4	66.5
MSI	77.6	72.9	**73.3**	71.5	72.4	71.3
Lympho invasion	74.3	75.4	**74.4**	69.8	70.9	70.3
MlH1	78.1	75.4	**75.2**	71.2	71.4	70.6
Anastomosis method	77.8	78.8	**77.8**	75.1	76.4	75.5
Margins radial	79.1	79.8	**78.5**	76.8	76.4	75.5
Colon adherent	78.2	80.3	78.8	80.0	80.3	**79.7**
Other meta	83.6	83.7	**83.7**	75.6	77.3	76.4
Peritoneal	83.9	84.2	**84.0**	79.0	80.3	78.6
Inflammatory infiltrate	82.9	86.2	**84.0**	83.3	84.2	82.9
Polyps type	84.8	85.2	**84.4**	80.5	82.3	81.2
Necrosis	84.6	85.7	**85.1**	79.9	81.3	80.5
Mucinous	84.5	86.7	**85.4**	82.6	83.7	82.8
Liver	86.9	86.2	**86.4**	76.1	76.8	76.4
Primary tumour rectum	87.8	86.2	**86.6**	84.4	84.7	83.8
Ressected meta	89.8	90.1	**89.7**	86.6	86.7	86.2
Margins distal	90.0	92.6	**90.3**	88.4	91.6	89.5
Polyps	90.3	91.6	**90.9**	90.1	90.1	90.1
Differentiation	91.8	92.6	**91.9**	90.0	92.1	90.7
Biopsy confirmed mata	93.8	94.1	**93.9**	94.3	94.6	**93.9**
Microscopic type	96.6	96.6	**96.4**	94.2	95.6	94.6
Pathologic response	97.7	97.5	**97.5**	91.4	93.1	91.9
Macro-average	81.9	82.1	**81.3**	77.3	78.4	77.4

Table 4: Performances of multiclass document classifiers for nominal categories using feature selection. Results sorted by baseline f-score performance. Best f-score per category is given in bold.

performance are the ones most benefited from our classifier, and the categories with highest baseline score are the only ones that do not get any improvement.

Our next experiment applies feature selection over the initial classifiers. The results are given in Table 4 for *Naive Bayes* and *SVM*[8]. We can see that the scenario changes when we add feature selection, with *Naive Bayes* achieving the highest performance in all cases. The performance for the hardest category (which is again *Tumour site*) raises to above 50% f-score, clearly beating the baseline. The highest-performing category is now *Pathologic response*, and *Naive Bayes* almost reaches perfect scores over this category, improving the baseline again. The macro-averaged results show that our best classifier is able to reach an f-score of 81.3% over the 27 nominal categories, with an improvement of 30.3% over the majority class baseline.

[8] *AdaBoost* obtains the same results with and without feature selection.

4.2 Numeric categories

In this section we present the results of our numeric classifiers in Table 5. In this case the results of *Naive Bayes* are worse than the baseline, and *AdaBoost* and SVM only achieve small improvements. One of the reasons for the low performance seems to be the strong bias of the categories towards the majority value. On these conditions, the baseline obtains the best result for 6 of the 9 categories. The macro-averaged performances show that the performance is insufficient for a real application.

For our next experiment we applied feature selection to the numeric classifiers, and the results are presented in Table 6. We can see that the overall performance goes down when applying feature selection, and the main cause for this seems the low number of features that are left for each instance.

5 Discussion

Our results over nominal categories show that our classifiers can achieve high performance (above 80% f-score in average) by relying on feature

Category	*Majority Class*			*Naive Bayes*			*SVM*			*AdaBoost*		
	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc
Nodes examined	7.4	7.4	7.4	83.1	58.1	**68.4**	82.1	58.6	68.4	81.8	57.6	67.6
Tumour length	42.9	42.9	**42.9**	41.7	24.6	31.0	50.3	37.0	42.6	44.9	39.4	42.0
Tumour width	47.8	47.8	**47.8**	41.4	26.1	32.0	51.8	42.4	46.6	46.5	42.9	44.6
Distal distance	52.2	52.2	52.2	63.3	34.0	44.2	70.1	53.2	**60.5**	52.0	51.7	51.8
Polyps number	62.1	62.1	**62.1**	48.9	43.4	46.0	57.0	54.2	55.6	58.9	58.6	58.8
Nodes positive	64.0	64.0	64.0	66.5	58.6	62.3	79.0	75.9	**77.4**	67.5	67.5	67.5
Tumour depth	70.0	70.0	**70.0**	70.1	49.8	58.2	74.7	61.1	67.2	70.0	70.0	**70.0**
Cav	72.4	72.4	**72.4**	72.1	71.4	71.8	73.1	69.5	71.2	72.4	72.4	**72.4**
Radial distance	94.1	94.1	**94.1**	94.1	94.1	**94.1**	95.4	92.1	93.7	94.1	94.1	**94.1**
Macro-average	57.0	57.0	57.0	64.6	51.1	56.4	70.4	60.4	**64.8**	65.3	61.6	63.2

Table 5: Performances for numeric categories without feature selection. Results sorted by baseline f-score performance. Best f-score per category is given in bold.

Category	*Majority Class*			*Naive Bayes*			*SVM*			*AdaBoost*		
	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc	Prec.	Rec.	F-sc
Nodes examined	7.4	7.4	7.4	35.3	32.0	**33.6**	22.7	21.7	22.2	26.7	25.1	25.9
Tumour length	42.9	42.9	**42.9**	39.4	31.0	34.7	42.3	37.9	40.0	43.7	40.9	42.2
Tumour width	47.8	47.8	47.8	53.5	49.3	51.3	53.3	51.7	**52.5**	49.0	48.8	48.9
Distal distance	52.2	52.2	**52.2**	53.8	31.5	39.8	52.5	51.7	52.1	52.2	52.2	**52.2**
Polyps number	62.1	62.1	62.1	52.8	51.2	52.0	64.4	64.0	**64.2**	60.2	59.6	59.9
Nodes positive	64.0	64.0	64.0	69.0	67.0	**68.0**	68.5	67.5	68.0	68.2	67.5	67.8
Tumour depth	70.0	70.0	70.0	70.2	62.6	66.1	70.8	70.4	**70.6**	70.0	70.0	70.0
Cav	72.4	72.4	**72.4**	71.1	65.5	68.2	72.4	72.4	**72.4**	72.4	72.4	**72.4**
Radial distance	94.1	94.1	**94.1**	94.1	94.1	**94.1**	94.1	94.1	**94.1**	94.1	94.1	**94.1**
Macro-average	57.0	57.0	57.0	59.9	53.8	56.4	60.1	59.1	**59.6**	59.6	58.9	59.3

Table 6: Performances for numeric categories with feature selection. Results sorted by baseline f-score performance. Best f-score per category is given in bold.

selection. These results have been attained using BOW features, and this indicates that pathology reports tend to use similar lexical elements to refer to the relevant classes. The results show promise to incorporate an extraction prototype into the medical workflow for nominal classes, which would aid the collection of structured information, and benefit from the interaction with the user.

One of the most interesting findings has been the effect of the feature selection step to achieve high performance. Apart from the increment of the f-score, feature selection would allow us to highlight the relevant terms in the document, and present them to the user for a better interaction.

Regarding the results for numeric categories, our strategy has not been successful, and the increments over the majority class baseline have been small. The baseline for these categories is higher than for nominal categories, and there is a strong bias towards the "zero" value. We observed that the main difficulty was to discriminate between "zero" and other classes, and a 2-step classifier would have been a better option to build upon. Our results over numeric categories also indicate that the generic BOW approach successfully evaluated over nominal categories may not be enough, and deeper analysis of the feature space may be required for these categories.

6 Conclusion

We have presented the results of a set of supervised text classification systems over different prediction categories in the domain of pathology records. Our results show that we are able to predict nominal labels with high average f-score (81.3%) and improve the majority class baseline by relying on Naive Bayes and feature selection. These results are positive for the integration of automatic aids in the medical workflow, and they illustrate that pathology reports contain repetitive lexical items that can be captured by a bag-of-words model. Our experiments also show that this is not the case for numeric labels, and richer features would be required in order to improve the baselines.

For future work one of our goals is to improve numeric classifiers by adding an initial classifier that identifies zero-valued instances before looking for the final value. We observed that

lexical items expressing negation may be relevant for this category (e.g. "No positive nodes were found"), we plan to incorporate the negation-classifier Negex (Chapman et al., 2001) to the feature extraction.

Finally, we want to combine our classifiers with a user interface that will allow clinicians to upload structured information into the database with the help of automatic predictions. The users will be able to copy the pathology reports, and the database fields will be pre-filled with the categories from the predictors. We will also highlight the top features from the selection process, and the user will be able to correct the automatic predictions before saving. All interactions will be kept and used to improve our classifiers.

Acknowledgments

We would like to thank Henry Gasko and his colleagues at the Royal Melbourne Hospital for providing anonymised pathology reports for our research.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

Wendy W. Chapman, Will Bridewellb, Paul Hanburya, Gregory F. Cooperb, and Bruce G. Buchananb. 2001. A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of Biomedical Informatics*, 34(5):301–310, October.

Anni Coden, Guergana Savova, Igor Sominsky, Michael Tanenblatt, James Masanz, Karin Schuler, James Cooper, Wei Guan, and Piet C de Groen. 2009. Automatically extracting cancer disease characteristics from pathology reports into a disease knowledge representation model. *Journal of Biomedical Informatics*, 42:937–949.

G. S. Dunham, M. G. Pacak, and A. W. Pratt. 1978. Automatic indexing of pathology data. *Journal of the American Society for Information Science*, 29(2):81–90, Mar.

Mark Hall. 1999. *Correlation-based Feature Subset Selection for Machine Learning*. Ph.D. thesis, Department of Computer Science, University of Waikato, New Zealand.

David A Hanauer, Gretchen Miela, Arul M Chinnaiyan, Alfred E Chang, and Douglas W Blayney. 2007. The registry case finding engine: an automated tool to identify cancer cases from unstructured, free-text pathology reports and clinical notes. *Journal of the American College of Surgeons*, 205(5):690–697, Nov.

Iain A McCowan, Darren C Moore, Anthony N Nguyen, Rayleen V Bowman, Belinda E Clarke, Edwina E Duhig, and Mary-Jane Fry. 2007. Collection of cancer stage data by classifying free-text medical reports. *Journal of the American Medical Informatics Association (JAMIA)*, 14:736–745.

Anthony Nguyen, Darren Moore, Iain McCowan, and Mary-Jane Courage. 2007. Multi-class classification of cancer stages from free-text histology reports using support vector machines. *Proceedings of the IEEE Engineering in Medicine and Biology Society Conference*, 2007:5140–5143.

Anthony N Nguyen, Michael J Lawley, David P Hansen, Rayleen V Bowman, Belinda E Clarke, Edwina E Duhig, and Shoni Colquist. 2010. Symbolic rule-based classification of lung cancer stages from free-text pathology reports. *Journal of the American Medical Informatics Association (JAMIA)*, 17:440–445.

Gunther Schadow and Clement J McDonald. 2003. Extracting structured information from free text pathology reports. *AMIA Annual Symposium Proceedings*, pages 584–588.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, USA.

Hua Xu, Kristin Anderson, Victor R Grann, and Carol Friedman. 2004. Facilitating cancer research using natural language processing of pathology reports. *Studies in health technology and informatics*, 107(Pt 1):565–572.

Classifying User Forum Participants: Separating the Gurus from the Hacks, and Other Tales of the Internet

Marco Lui and Timothy Baldwin
NICTA VRL
Department of Computer Science and Software Engineering
University of Melbourne, VIC 3010, Australia
saffsd@gmail.com, tb@ldwin.net

Abstract

This paper introduces a novel user classification task in the context of web user forums. We present a definition of four basic user characteristics and an annotated dataset. We outline a series of approaches for predicting user characteristics, utilising aggregated post features and user/thread network analysis in a supervised learning context. Using the proposed feature sets, we achieve results above both a naive baseline and a bag-of-words approach, for all four of our basic user characteristics. In all cases, our best-performing classifier is statistically indistinct from an upper bound based on the inter-annotator agreement for the task.

1 Introduction

The most natural form of communication is through dialogue, and in the Internet age this manifests itself via modalities such as forums and mailing lists. What these systems have in common is that they are a textual representation of a *threaded discourse*. The Internet is full of communities which engage in innumerable discourses, generating massive quantities of data in the process. This data is rich in information, and with the help of computers we are able to archive it, index it, query it and retrieve it. In theory, this would allow people to take a question to an online community, search its archives for the same or similar questions, follow up on the contents of prior discussion and find an answer. However, in practice, search forum accessibility tends to be limited at best, prompting recent interest in information access for user forums (Cong et al., 2008; Elsas and Carbonell, 2009; Seo et al., 2009).

One problem with current approaches to accessing forum data is that they tend not to take into account the structure of the discourse itself, or other characteristics of the forum or forum participants. The bag-of-words (BOW) model common in information retrieval (IR) and text categorisation discards all contextual information. However, even in IR it has long been known that much more information than simple term occurrence is available. In the modern era of web search, for example, extensive use is made of link structure (Brin and Page, 1998), anchor text, document zones, and a plethora of other document (and query, click stream and user) features (Manning et al., 2008).

The natural question to ask at this point is, *What additional structure can we extract from web forum data?* Previous work has been done in extracting useful information from various dimensions of web forums, such as the post-level structure (Kim et al., 2010). One dimension that has received relatively little attention is how we can use information about the identity of the participants to extract useful information from a web forum. In this work we will examine how we can utilize such *user-level* structure to improve performance over a user classification task.

We have used the term *threaded discourse* to describe online data that represents a record of messages exchanged between a group of participants. In this work, we examine data from LinuxQuestions, a popular Internet forum for Linux-related troubleshooting. Aside from a limited set of features specific to the Linux-related troubleshooting domain, however, our techniques are domain-inspecific and expected to generalize to any data that can be interpreted as a threaded discourse.

This work is part of ILIAD (Baldwin et al., 2010), an ongoing effort to improve information access in linux forums. Our contribution to the project is techniques to identify characteristics of forum users, building on earlier work in the space (Lui, 2009). The problem that we face here is two-fold: Firstly, there is no established ontology for

Marco Lui and Timothy Baldwin. 2010. Classifying User Forum Participants: Separating the Gurus from the Hacks, and Other Tales of the Internet. In *Proceedings of Australasian Language Technology Association Workshop*, pages 49−57

characteristics of forum users. To address this, we have designed a set of attributes that we expect to be helpful in improving information access over forum data. Secondly, in order to exploit user characteristics we would need to evaluate a large number of users. This quantity of data would be much too large to be processed manually. We therefore apply supervised machine learning techniques to allow us to effectively discover the characteristics of a large number of forum users in an automated fashion.

2 Related Work

Lui and Baldwin (2009b) showed that user-level structure is useful in predicting percieved quality of forum posts. The data they evaluate over is extracted from Nabble, where the ratings provided by users are interpeted as the gold-standard for a correct classification. The task was originally proposed by Weimer et al. (2007) and further explored by Weimer and Gurevych (2007). In both cases, the authors focus on heuristic post-level features, which are used to predict perceived quality of posts using a supervised machine learning approach. Lui and Baldwin (2009b) showed that features based on user-level structure outperformed the benchmark set by Weimer and Gurevych (2007) on a closely-related task, by using user-level structure to inform a post-level classification task. We build on this work by utilizing the user-level structure to perform our novel user-level classification task.

In work on thread classification, Baldwin et al. (2007) attempted to classify forum threads scraped from Linux-related newsgroups according to three attributes: (1) Task Oriented: *is the thread about a specific problem?*; (2) Complete: *is the problem described in adequate detail?*; and (3) Solved: *has a solution been provided?* They manually annotated a set of 250 threads for these attributes, and extracted a set of features to describe each thread based on the aggregation of features from posts in different sections of the thread. We provide a novel extension of this concept, whereby we aggregate posts from a given user.

Wanas et al. (2008) develop a set of post-level features for a classification task involving post and rating data from Slashdot. Their task involves classifying posts into one of three quality levels (High, Medium or Low), where the gold-standard is provided by user annotations from the forum. This is conceptually very similar to our task, and we build on this feature set.

Extracting community structure from networks can yield insights into the relationships between users in a forum (Newman and Girvan, 2004; Drineas et al., 2004; Chapanond et al., 2005), and could in turn aid in engineering descriptions of the users more suited to a particular task. Agrawal et al. (2003) describe a technique for partitioning the users in an online community based on their opinion on a given topic. They find that basic text classification techniques are unable to do better than the majority-class baseline for this particular task. They then describe a technique based on modeling the community as a *reply-to* network, with users as individual nodes, and edges indicating that a user has replied to a post by another user; using this representation, they are able to do much better than the baseline. Fortuna et al. (2007) build on this work, defining additional classes of networks that represent some of the relationships present in an online community. Part of our feature set is derived from modelling Internet forum users on the basis of the interactions that exist between them, such as a tendency to reply to each other or to co-participate in threads. We extend the social network analysis of Agrawal et al. (2003) and Fortuna et al. (2007) to generate user-level features.

Malouf and Mullen (2008) present the task of determining the political leaning of users on a U.S. political discussion site. They apply network analysis to the task, based on the observation that users tend to quote users of opposing political leaning more than they quote those of similar political leaning. They found that standard text categorisation methods performed poorly over their task, and that the results were improved significantly by incorporating network-derived features.

In a similar vein, Carvalho et al. (2007) used a combination of textual features (in the form of "email speech acts") and network-based features to learn which users were team leaders. They found that the network-based features enhanced classification accuracy.

Sentiment analysis (Pang and Lee, 2008) relates to this work as one of our user characteristics (POSITIVITY) is an expression of user sentiment. However, sentiment analysis has tended to focus on individual documents, and rarely takes into account the author. An exception to this is the work of Thomas et al. (2006), who attempted to predict

which way each speaker in a U.S. Congressional debate on a proposed bill voted, on the basis of both what was said and the indication of agreement between speakers. Their task is related to ours in that it involves a user-level classification, but it focused on extracting information identifying where the speakers agree and disagree.

Expert finding is the task of ranking experts relative to each of a series of queries, and has been part of the TREC Enterprise Track (Craswell et al., 2005; Soboroff et al., 2006; Balog et al., 2006; Fang and Zhai, 2007). The challenge is to estimate the likelihood of a given individual being an expert on a particular topic, on the basis of a document collection. There is certainly scope to evaluate the utility of the user characteristics proposed in this research in the context of the TREC expert finding task, although only a small fraction of the document collection (the mailing list archives) has the threaded structure requisite for our methods, and our focus is on the general characteristics of the user rather than their topic-specific expertise.

3 User Characteristics

We have designed a set of user-level attributes which we expect to be useful in improving information access over forum data. The attributes were selected based on our personal experiences in interacting with online communities. In this, we sought to capture the attributes of users who provide meaningful contributions, as follows:

CLARITY: How clear is what the user meant in each of their posts, in the broader context of the thread?

PROFICIENCY: What level of perceived technical competence does the user have in their posts?

POSITIVITY: How positive is the user in their posts?

EFFORT: How much effort does the user put into their posts?

Each user-level attribute is quantified by way of a 5 point ordinal scale, as detailed in Table 1.

While we have described the four attributes as if they were orthogonal to each other, in reality there are obvious overlaps. For example, high clarity often implies high effort, but the reverse is not necessarily true. For simplicity, we do not consider the interactions between the characteristics in this work, leaving it as a possibility for further research.

4 Dataset

We created a new dataset specifically for this work based on data crawled from LinuxQuestions,[1] a popular Internet forum for Linux troubleshooting. From this forum, we scraped a background collection of 34157 threads, spanning 126094 posts by 25361 users.

In order to evaluate how well we can automatically rate forum users in each of our four user characteristics (from Section 3), we randomly selected 50 users who had each participated in more than 15 different threads in the full dataset. We asked four independent annotators to annotate the 50 users over each of the 4 attributes. The annotators all had a computer science background, and had participated in Linux-related online communities. For each attribute, the annotators were asked to choose a rating on a five-point scale, based on the description of user attributes from Section 3.

For each of the 50 users, we randomly selected 15 threads that they had participated in, and partitioned these into 5 separate annotation instances as follows: for the first instance, we selected 1 thread; for the second instance we selected 2 threads; and so on, giving us 5 instances, each with 1 to 5 threads. This gave us a total of 250 annotation instances (with 5 instances per user). We chose to annotate each user multiple times in order to build a more complete picture of the user. Each instance presented a different number of threads to the annotator, in order to give the annotators maximal context in annotating a user while still minimizing the number of threads we required the user to have participated in.

Each annotator was asked to rate all 250 annotation instances, meaning that they actually saw each of the 50 users a total of five times each. Annotators were not alerted to the fact that they would annotate each user five times, and all usernames were removed from the threads before being displayed to the annotator. However, for a given annotation instance, the annotator was alerted to which posts the user being annotated had authored. The posts of other users in those threads where also presented to provide the full thread context, but the annotators were instructed to use those posts only to interpret the posts of the user in question.

Since each annotator annotated each user 5 times for each attribute, we compute a score for each user–annotator–attribute combination, which

[1] `http://www.linuxquestions.org`

Attribute	*Value*		*Description*
CLARITY	1	**Unintelligible**	It is impossible to make sense of the user's posts; clear as mud!
	2	**Somewhat confused**	The meaning of the user's posts is ambiguous or open to interpretation
	3	**Comprehensible**	With some effort, it is possible to understand the meaning of the post
	4	**Reasonably clear**	You occasionally question the meaning of the user's posts
	5	**Very clear**	Meaning is always immediately obvious relative to the thread; sparkling clarity!
PROFICIENCY	1	**Hack**	The posts of this user make it patently obvious that they have no technical knowledge relevant to the threads they participate in; get off the forum!
	2	**Newbie**	Has limited understanding of the very basics, but nothing more
	3	**Average**	Usually able to make a meaningful technical contribution, but struggles with more difficult/specialized problems
	4	**Veteran**	User gives the impression of knowing what they are talking about, with good insights into the topic of the thread but also some gaps in their knowledge
	5	**Guru**	The posts of this user inspire supreme confidence, and leave the reader with a warm, fuzzy feeling!
POSITIVITY	1	**Demon**	Deliberately and systematically negative with no positive contribution; the prince/princess of evil!
	2	**Snark**	The user is somewhat hurtful in their posts
	3	**Dull**	The user's posts express no strong sentiment
	4	**Jolly**	The user's posts are generally pleasant
	5	**Solar**	Goes out of his/her way in trying to make a positive contribution in all possible ways; positively radiant!
EFFORT	1	**Loser**	Zero effort on the part of the user
	2	**Slacker**	Obvious deficiency in effort
	3	**Plodder**	User's posts are unremarkable in terms of the effort put in
	4	**Strider**	Puts obvious effort into their post
	5	**Turbo**	Goes out of his/her way in trying to make a contribution; an eager beaver!

Table 1: A detailed description of the user-level attribute values

is simply the sum across the 5 annotations. Using this score, we then rank the users for each pairing of annotator–quality.

We formulated the user-level classification task as four separate classification tasks, across the four attributes. In order to account for subtle variance in annotators' interpretations of the ordinal scale, we took a non-parametric approach to the data: we pooled all of the annotator ratings and established a single ranking over all the annotated users for each attribute. We then discretized this ranking into 5 equal-sized bins, in order to provide a more coarse-grained view of the relative ordering between users. Therefore, our task can be interpreted as assigning each user to their corresponding uniformly-distributed quintile on each attribute.

4.1 Inter-annotator Agreement

We calculate inter-annotator agreement on each of the four attributes via leave-one-out cross-validation. For each user-annotator-attribute combination, we calculate two scores: the sum of ratings given by the annotator being considered, and the sum of ratings given by all the other annotators. For each of the four attributes, we rank the users based on each of these two scores, and compute Kendall's τ (Kendall, 1938) between the two ranklists (Table 2), as well as the p-value for the significance of the τ value.

Attribute	*Annotator*	τ	p
Clarity	Annotator 1	0.235	0.016
	Annotator 2	0.221	0.024
	Annotator 3	0.292	0.003
	Annotator 4	0.307	0.002
Effort	Annotator 1	0.517	0.000
	Annotator 2	0.707	0.000
	Annotator 3	0.682	0.000
	Annotator 4	0.610	0.000
Proficiency	Annotator 1	0.582	0.000
	Annotator 2	0.460	0.000
	Annotator 3	0.536	0.000
	Annotator 4	0.407	0.000
Positivity	Annotator 1	0.009	0.924
	Annotator 2	0.434	0.000
	Annotator 3	0.473	0.000
	Annotator 4	0.436	0.000

Table 2: Inter-annotator agreement, based on Kendall's τ and associated p-value

We see that for all attributes, there is a statistically significant correlation between the annotations. This correlation is strongest in the EFFORT and PROFICIENCY attributes, and weakest in the

CLARITY attribute. This is partly to be expected, since CLARITY is more subjective than EFFORT or PROFICIENCY. POSITIVITY shows an interesting quirk, where the ratings from one annotator appear completely uncorrelated with those of all the others. This suggests that POSITIVITY as an attribute is slightly more subjective than the others.

5 Feature Extraction

We extract features for each user based on aggregating post-level features and via social network analysis.

5.1 Post-Aggregate Features

The most basic feature set we consider is a simple bag-of-words (BOW), computed as the sum of the bag-of-words model over each of the user's individual posts.

We also make use of two post-level feature sets from the literature on web user forum classification. The first is that of Baldwin et al. (2007) (BALDWIN^{Post}), and outlined in Table 3. It was designed to represent key posts in a thread for a thread-level classification (see Section 2) task. We compute this feature set for each of a user's posts.

The second is that of Wanas et al. (2008), and is described in Table 4. In this case, it was developed for a post-level classification task rating post quality, and thus lends itself readily to our post-aggregate user representation.

From each of BALDWIN^{Post} and WANAS, we derive a user-level feature set by finding the mean of each feature value over all of the user's posts in the full dataset. For boolean features, this can be directly interpreted as the proportion of the user's posts in which the feature is present. These feature sets are referred to as BALDWIN$^{Post}_{\text{AGG}}$ and WANAS$_{\text{AGG}}$ respectively.

Whereas it is possible for us to engineer a novel post-level feature set, our aim in this research is not to analyze the feature sets themselves, but rather to show that our techniques utilizing user-level structure perform better than techniques which ignore this information. We leave post-level feature engineering as an open avenue of further work.

5.2 Network Features

Fortuna et al. (2007) present a method of describing forum data using Social Network Analysis. The network is a graph representation of relationships within the forum. Building on Fortuna et al. (2007), we consider *User Networks*, where each node represents a user, and *Thread Networks*, where each node represents a thread. In this work, we consider two User Networks and one Thread Network, namely: (1) POSTAFTER, (2) THREADPART, and (3) COMMONAUTHORS, respectively. The networks we define build directly on work done by Fortuna et al. (2007), but the application to user-level feature extraction is novel.

Feature name	*Description*	*Type*
distribution	Mention of Linux distribution name?	Boolean
beginner	Mention "newbie" terms?	Boolean
emoticons	Presence of "smiley faces"?	Boolean
version numbers	Presence of version numbers?	Boolean
URLs	Presence of hyperlinks?	Boolean
words	Number of words in post	Integer
sentence	Number of sentences in post	Integer
question sentence	Number of questions in post	Integer
exclaim sentence	Number of exclamations in post	Integer
declarative sentence	Number of declarative sentences	Integer
other sentence	Number of other sentences	Integer

Table 3: The BALDWIN^{Post} feature set

Feature name	*Description*	*Type*
onTopic	Post's relevance to the topic of a thread	Real
overlapPrev	Post's largest overlap with a previous post	Real
overlapDist	Distance to previous overlapping post	Integer
timeliness	Ratio of time from prev post to average inter-post interval	Real
lengthiness	Ratio of post length to average post length in thread	Real
emoticons	Ratio of emoticons to sentences	Real
capitals	Ratio of capitals to sentences	Real
weblinks	Ratio of links to number of sentences	Real

Table 4: The WANAS feature set

POSTAFTER is modeled on the *reply-to* network described in Fortuna et al. (2007). Our data does not contain explicit annotation about the reply structure in a thread, so we approximate this information by the temporal relationship between posts. There exist more sophisticated approaches to the discovery of reply structure in a thread (Kim et al., 2010), and we consider integrating such methods to be an important avenue of further work.

POSTAFTER is parametrized with two values: $dist$ and $count$. Being a User Network, the nodes represent users. Two users $A1$ and $A2$ have a directed edge from $A1$ to $A2$ if and only if $A1$ submits a post to a thread that is within $dist$ posts after a post in the same thread by $A2$ on at least $count$ occasions. Note that this can occur more than once in a single thread. For our experiments, we used $dist = 1$ and $count = 3$.

THREADPART is implemented as described in Fortuna et al. (2007): nodes are again users, and each undirected edge indicates that two users have posted in the same thread on at least k occasions. Fortuna et al. (2007) set $k = 5$, but we only report on results for $k = 2$ and $k = 3$, as we found that for our dataset, the network is too sparse for higher values.

COMMONAUTHORS is also implemented as described in Fortuna et al. (2007): nodes are threads, and each undirected edge indicates that two threads have at least m users in common. We follow Fortuna et al. (2007) in setting $m = 3$.

In User Networks, the edges represent some relationship between users. From a User Network, we generate a feature vector v for each user. v is of length N, where N is the total number of nodes, or equivalently, the total number of users in the network. v has at least one feature set to 1, which corresponds to the user described by this feature vector, which we will hereafter refer to as the originator. Features representing users directly connected to the originator in the network receive a feature value of 1, and users that are second-level neighbours of the originator are set to a feature value of 0.5. All other values in v are set to 0.

For Thread Networks, edges represent relationships between threads. The method for computing a feature vector is similar to that for User Networks. The key difference is that in this instance, nodes represent threads and not users. Therefore, to describe a particular user, we consider threads that the user has posted in. We define a vector v of length T, where T is the total number of threads in the forum. Given the set S_0 of threads that the user has posted in, for each thread in S_0, we assign the value 1 to the feature in v corresponding to that thread. We then consider S_1, the set of immediate neighbours of S_0, and assign the value 1 to their corresponding features in v. Finally, we consider S_2, the immediate neighbours of S_1, and assign the value of 0.5 to their corresponding features. All other features are assigned the value 0.

6 Experimental Methodology

In all experiments, we build our classifiers using a support vector machine (`SVM`: Joachims (1998)), using `bsvm` (Hsu and Lin, 2006) with a linear kernel. For each combination of features, we evaluate it by carrying out 10-fold cross-validation. The partitioning is performed once and re-used for each pairing of learner and feature set.

Our experiments were performed using `hydrat` (Lui and Baldwin, 2009a), an open-source framework for comparing classification systems. `hydrat` provides facilities for managing and combining feature sets, setting up cross-validation tasks and automatically computing corresponding results. Features were extracted from the forum data using `forum_features`,[2] a Python module implementing a data model for forum data.

We evaluate our classifiers using microaveraged F-score ($\mathcal{F}_\mu$), reflecting the average performance *per-document*. As our classes are ordinal (representing quintiles of users), we additionally present results based on mean absolute error (MAE). MAE is the average absolute distance of the predicted ($Pred$) ordinal value from the gold-standard (G) value. It is a reflection of how far off the mark the average prediction is, with an MAE of 0 indicating perfect classifier performance.

As a baseline, we use a simple majority-class (`ZeroR`) classifier. A benchmark classifier is constructed based on a BOW feature set, as is the standard in text categorization. To derive an upper bound for the task, we perform leave-one-out cross-validation over our annotations, and calculate the mean F-score and MAE between each annotator and the combination of the remaining annotators.

When comparing a result to a baseline or a benchmark value, we also compute the p-value for a two-tailed paired t-test. In line with standard practice, we interpret $p < 0.05$ as statistically significant.

7 Results

First, we present results for each of the feature sets in isolation over the four user characteristics (Table 5). In each case, we present the results for the majority class (`ZeroR`) baseline and the bag-of-words (BOW) benchmark in the first two rows. Statistically-significant improvements over `ZeroR` (including BOW) are suffixed with "*", and statistically-significant improvements over BOW are suffixed with "$^+$". The best overall result for a given task achieved across all combinations of feature sets is presented in **boldface**, and is achieved for a single feature set

[2] `http://github.com/saffsd/forum_features`

Attribute	*Feature Set*	$\mathcal{F}_\mu$	*MAE*
CLARITY	ZeroR	0.020	2.040
	BOW	0.120	1.620*
	WANAS$_{AGG}$	0.100	1.760
	BALDWIN$^{Post}_{AGG}$	0.120	1.860
	THREADPART$_2$	0.240*	1.540*
	THREADPART$_3$	**0.260**$^{*+}$	**1.360***
	POSTAFTER	0.220*	1.440*
PROFICIENCY	ZeroR	0.000	1.980
	BOW	**0.240***	**1.380***
	WANAS$_{AGG}$	0.000	2.080
	BALDWIN$^{Post}_{AGG}$	0.060	1.740
	THREADPART$_2$	0.180*	1.820
	THREADPART$_3$	0.120*	1.700
	POSTAFTER	0.200*	1.900
POSITIVITY	ZeroR	0.040	1.880
	BOW	0.140	1.660
	WANAS$_{AGG}$	0.120	1.680
	BALDWIN$^{Post}_{AGG}$	0.120	1.580
	THREADPART$_2$	0.180*	1.720
	THREADPART$_3$	0.120	1.760
	POSTAFTER	**0.220** *	**1.340***
EFFORT	ZeroR	0.000	1.760
	BOW	**0.320***	**1.240***
	WANAS$_{AGG}$	0.100	1.600
	BALDWIN$^{Post}_{AGG}$	0.300*	1.420
	THREADPART$_2$	0.180*	1.700
	THREADPART$_3$	0.140*	1.700*
	POSTAFTER	0.100*	1.900

Table 5: Results for individual feature sets.

in the case of CLARITY and POSITIVITY, both using User Network feature sets.

The benchmark results (BOW) are considerably more impressive than the ZeroR baseline. For CLARITY, THREADPART$_3$ achieves the best result for the task, beating the BOW at a level of statistical significance for $\mathcal{F}_\mu$. Recall that THREADPART$_2$ was based on a graph of co-participation in threads, suggesting that knowledge of which users co-post to threads is informative in predicting how clear their posts are on average. In other words, there are clusters of users who co-predict their respective post clarity.

For POSITIVITY, POSTAFTER beats the BOW benchmark, but not at a level of statistical significance in this case. POSTAFTER may work in capturing POSITIVITY due to sets of antagonistic users who respond to each other's posts negatively (e.g. commonly engage in flame wars), or to cooperative users who engage in a mutually-supportive dialogue, each building positively on the previous poster's comments.

For both CLARITY and POSITIVITY, the aforementioned individual feature sets achieve the best overall results in our experiments, i.e. combining these feature sets with BOW or other feature sets did not improve the results. In both cases, the MAE is around 1.3.

For PROFICIENCY and EFFORT, the BOW $\mathcal{F}_\mu$ results were notably higher, to the degree that none of the feature sets in isolation were able to better it. As a result, we looked to the combination of up to three feature sets, and present in Table 6 the best-achieved results with two or three feature sets for PROFICIENCY and EFFORT. In both cases, it is the combination of the BOW feature set with one of the User Network feature sets and one of the post-level feature sets that produces the best result, illustrating the complementary nature of the three basic feature set types. Results for the BOW feature set in isolation, along with results for BOW with each of the two feature sets in the best-performing method, are presented to illustrate the relative effect of each. In the case of PROFICIENCY, THREADPART$_2$ and BALDWIN$^{Post}_{AGG}$ both lead to increased $\mathcal{F}_\mu$ when combined with BOW, as compared to the simple feature set (but only the combination of all three is significantly better than simple BOW). That is, PROFICIENCY appears to be the most multi-faceted of the four user classification attributes, in being best captured through the combination of lexical choice, macro post-level features, and network-based analysis of thread co-participation. With the network-based features, we suggest this is largely a negative effect, in that "hacks" and "newbies" are characterised by a *lack* of thread co-participation.

With EFFORT, BOW achieves by far its highest $\mathcal{F}_\mu$ across all four classification tasks, and the combination with THREADPART$_3$ and WANAS$_{AGG}$ barely surpasses it, at a level which is not statistically significant.

That the best results are achieved in all four classification tasks with network-based features (possibly in combination with other feature sets) is telling, and underlines the potential of network analysis for user classification. The aggregate post-level feature sets BALDWIN$^{Post}_{AGG}$ and WANAS$_{AGG}$ are less effective, but bear in mind that they were not tailored specifically for the user classification task, so it is a positive result that they have an impact when aggregated over user-level structure, and suggests that further work in customizing the per-post feature set will yield further improvements on this task.

Finally, we turn to analysis of inter-annotator agreement for the four user classification subtasks,

Attribute	*Feature Sets Present*	$\mathcal{F}_\mu$	*MAE*
PROFICIENCY	BOW	0.240	1.380
	BOW $\oplus$ THREADPART$_2$	0.260	1.280
	BOW $\oplus$ BALDWIN$^{Post}_{\text{AGG}}$	0.320	1.200
	BOW $\oplus$ THREADPART$_2$ $\oplus$ BALDWIN$^{Post}_{\text{AGG}}$	**0.360**$^+$	**1.080**
EFFORT	BOW	0.320	1.240
	BOW $\oplus$ THREADPART$_3$	0.320	1.240
	BOW $\oplus$ WANAS$_{\text{AGG}}$	0.300	1.280
	BOW $\oplus$ THREADPART$_3$ $\oplus$ WANAS$_{\text{AGG}}$	**0.340**	**1.220**

Table 6: Results for augmented feature sets

Attribute	BOW	*Best*	*MIA*	p_{BoW}	p_{Best}
CLARITY	0.120	0.260	0.240	**0.049**	0.723
PROF	0.240	0.360	0.395	**0.009**	0.427
POS	0.140	0.220	0.335	**0.011**	0.126
EFFORT	0.320	0.340	0.410	0.108	0.193

Table 7: BOW benchmark, best result and mean inter-annotator (MIA) $\mathcal{F}_\mu$ over each user attribute

to gauge the quality of the results achieved by our best classifiers in each case. In Table 7, we reproduce the BOW and best $\mathcal{F}_\mu$ results from Tables 5 and 6, and additionally present the mean inter-annotator (MIA) $\mathcal{F}_\mu$ based on leave-one-out cross-validation. We additionally present the p-value for the two-tailed paired t-test for each of BOW–MIA and best–MIA. In addition to being able to compare the $\mathcal{F}_\mu$ values directly, we can observe that for CLARITY, PROF(ICIENCY) and POS(ITIVITY), the best-performing classifier is both significantly better than the BOW benchmark (and `ZeroR` baseline), and statistically indistinguishable from the upper bound figure. In the case of EFFORT, there is no significant difference between BOW and the upper bound, so it would highly unlikely that we could achieve a significant improvement over BOW for any of our classifiers.

In summary, we were able to consistently exceed the majority class baseline on this task using user-level features, attaining results that were competitive with those utilising a state-of-the-art bag-of-words benchmark. We found that in most cases our results exceeded the benchmark to a high degree of statistical significance, with network-based features featuring prominently for all classification subtasks.

8 Further Work

Given that the intention of this work is to enhance information access over web forum data, the next step we intend to take is to apply our trained classifiers to a larger corpus of web forum data, and assess the impact of the predictions in a task-based evaluation. Examples of such tasks include predicting perceived post quality (Weimer and Gurevych, 2007) and identifying troubleshooting-oriented threads (Baldwin et al., 2007). We also note that there is limited room for progress given our current interpretation of the inter-annotator agreement. We intend to further analyze the annotations. In particular, since each annotator annotated each user five times, we intend to study the interaction between the number of context posts and the ratings given by the annotator.

9 Conclusion

In this work, we introduced a novel user classification task over web user forums. We prepared an annotated dataset relevant to the task, which we will release to the research community.

We extracted user-level features over aggregations of user posts, as well as via anaylsis of social networks in a web forum. We investigated each feature set we defined in isolation as well as in combination with the benchmark feature sets. We have shown that these user-level features can consistently outperform a majority-class baseline over a user classification task.

We succeeded in showing that user-level features have empirical utility in user classification, and we expect that the use of these features will generalize well to tasks over other aspects of threaded discourse, for example in profiling users or in ranking threads for information retrieval.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

Rakesh Agrawal, Sridhar Rajagopalan, Ramakrishnan Srikant, and Yirong Xu. 2003. Mining newsgroups using networks arising from social behavior. In *Proceedings of the Twelfth International World Wide Web Conference (WWW'03)*, pages 529–535, Budapest, Hungary.

Timothy Baldwin, David Martinez, and Richard Baron Penman. 2007. Automatic Thread Classification for Linux User Forum Information Access. In *Proceedings of the Twelfth Australasian Document Computing Symposium (ADCS 2007)*, pages 72–79, Melbourne, Australia.

Timothy Baldwin, David Martinez, Richard B Penman, Su Nam Kim, Marco Lui, Li Wang, and Andrew Mackinlay. 2010. Intelligent Linux Information Access by Data Mining : the ILIAD Project. In *Proceedings of the NAACL 2010 Workshop on Computational Linguistics in a World of Social Media: #SocialMedia*, pages 15–16, Los Angeles, USA.

Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. 2006. Formal models for expert finding in enterprise corpora. In *Proceedings of 29th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*, pages 43–50.

Sergei Brin and Larry Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117.

Vitor Carvalho, Wen Wu, and William Cohen. 2007. Discovering leadership roles in email workgroups. In *Proceedings of the 4th Conference on Email and Anti-Spam (CEAS 2007)*, Mountain View, USA.

Anurat Chapanond, Mukkai S. Krishnamoorthy, and Bülent Yener. 2005. Graph theoretic and spectral analysis of Enron email data. *Computational and Mathematical Organization Theory*, 11(3):265–281.

Gao Cong, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. 2008. Finding question-answer pairs from online forums. In *Proceedings of 31st International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*, pages 467–474, Singapore.

Nick Craswell, Arjen P. de Vries, and Ian Soboroff. 2005. Overview of the TREC-2005 Enterprise track. In *Proceedings of the 14th Text REtrieval Conference (TREC 2005)*, Gaithersburg, USA.

Petros Drineas, Mukkai S. Krishnamoorthy, Michael D. Sofka, and Bülent Yener. 2004. Studying e-mail graphs for intelligence monitoring and analysis in the absence of semantic information. In *of the IEEE International Conference on Intelligence and Security Informatics*, pages 297–306, Tucson, USA.

Jonathan L. Elsas and Jaime G. Carbonell. 2009. It pays to be picky: An evaluation of thread retrieval in online forums. In *Proceedings of 32nd International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'09)*, pages 714–715, Boston, USA.

Hui Fang and ChengXiang Zhai. 2007. Probabilistic models for expert finding. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR'07)*, pages 418–430, Rome, Italy.

Blaz Fortuna, Eduarda Mendes Rodrigues, and Natasa Milic-Frayling. 2007. Improving the classification of newsgroup messages through social network analysis. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management (CIKM '07)*, pages 877–880, Lisboa, Portugal.

Chih-Wei Hsu and Chih-Jen Lin. 2006. BSVM-2.06. `http://www.csie.ntu.edu.tw/cjlin/bsvm/`. Retrieved on 15/09/2009.

Thorsten Joachims. 1998. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, Germany.

M. G. Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93.

Su Nam Kim, Li Wang, and Timothy Baldwin. 2010. Tagging and Linking Web Forum Posts. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 192–202, Uppsala, Sweden.

Marco Lui and Timothy Baldwin. 2009a. hydrat. `http://hydrat.googlecode.com`. Retrieved on 15/09/2009.

Marco Lui and Timothy Baldwin. 2009b. You Are What You Post: User-level Features in Threaded Discourse. In *Proceedings of the 14th Australasian Document Computing Symposium*, pages 98–105, Sydney, Australia.

Marco Lui. 2009. *Impact of user characteristics on online forum classification tasks*. Honours thesis, The University of Melbourne, Australia.

Robert Malouf and Tony Mullen. 2008. Taking sides: Graph-based user classification for informal online political discourse. *Internet Research*, 18(2):177–190.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK.

Mark E.J. Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical Review E*, 69. Article Number 26113.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135.

Jangwon Seo, W. Bruce Croft, and David A. Smith. 2009. Online community search using thread structure. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 1907–1910, Hong Kong, China.

Ian Soboroff, Arjen P. de Vries, and Nick Craswell. 2006. Overview of the TREC-2006 Enterprise track. In *Proceedings of the 15th Text REtrieval Conference (TREC 2006)*, Gaithersburg, USA.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 327–335, Sydney, Australia.

Nayer Wanas, Motaz El-Saban, Heba Ashour, and Waleed Ammar. 2008. Automatic scoring of online discussion posts. In *Proceedings of the 2nd ACM Workshop on Information Credibility on the web (WICOW '08)*, Napa Valley, USA.

Markus Weimer and Iryna Gurevych. 2007. Predicting the perceived quality of web forum posts. In *Proceedings of the 2007 Conference on Recent Advances in Natural Language Processing (RANLP-07)*, Borovets, Bulgaria.

Markus Weimer, Iryna Gurevych, and Max Mühlhäuser. 2007. Automatically assessing the post quality in online discussions on software. In *Proceedings of the 45th Annual Meeting of the ACL: Interactive Poster and Demonstration Sessions*, pages 125–128, Prague, Czech Republic.

Fun with Filtering French

Alexandra L. Uitdenbogerd
RMIT University
Melbourne, Victoria, Australia

Abstract

Early use of corpora for language learning has included analysis of word usage via concordancing. In addition, some attempts have been made to use readability criteria for recommending reading to learners. In this paper we discuss various tools and approaches for enhanced language learning support, including different methods of filtering text based on vocabulary and grammatical criteria. We demonstrate the effects of various criteria on the retrieval of text, assuming the user is English-speaking and learning French. Filtering text based on a small vocabulary of frequently occurring words, a set of English-French cognates and named entities, and high coverage criteria, results in the retrieval of short readable extracts from French literature. We expect that text available from the web may yield many more documents of appropriate readability.

1 Introduction

There is a considerable need for people to learn and become proficient in foreign languages: the majority of scientific discourse is published in English; students travel to different countries to study; people migrate for career opportunities.

Language skills are often divided into four communication tasks: listening, speaking, reading, and writing. Each of these skills can be developed and practised separately to a certain extent. Improving reading skill in a language would clearly involve devoting a substantial amount of time to reading.

It has been demonstrated that *extensive* reading at a comfortable level in a foreign language is more effective for improving language acquisition than *intensive* reading at more difficult levels (Bell, 2001). Therefore there is a need for reading material at multiple language skill levels to allow learners to practise. Some publishers provide graded reading books targeted at the foreign language learner, with levels indicated either by an assumed base vocabulary size, a standard level such as that defined by the Common European Framework of Reference for Languages (COE, 2003), year of study, or an unexplained level structure. The simplest graded readers based on vocabulary size that we have seen use a base vocabulary of 150 words. Beginner readers tend to be much shorter in length than those for advanced learners. For example, the level 1 readers in the Bibliobus Collection A are approximately 150 words in length, in a comic book format, consisting of two short stories (Cowling, 1982).

Several researchers independently proposed the idea of retrieving reading material from the Web based on its readability for the purpose of reading practice or study (Collins-Thompson and Callan, 2004a; Ghadirian, 2002; Katz and Bauer, 2001; Nilsson and Borin, 2002; Uitdenbogerd, 2003). This motivated some new studies of measuring readability (Collins-Thompson and Callan, 2004b; Schwarm and Ostendorf, 2005; Si and Callan, 2001) that are more sophisticated than was possible in the initial period of readability research (Bormuth, 1966; Chall and Dale, 1995; Cornaire, 1988; Granowskey and Botel, 1974; Klare, 1974). Our earlier work demonstrated that simple techniques are still very powerful for foreign language readability measurement, but could be improved by the inclusion of automated cognate detection for a specific first and second language pair (Uitdenbogerd, 2005). Recently, the methodology of producing readability measures has been questioned, with alternative approaches defined (van Oosten et al., 2010). In a related idea, filtering according to lexical constraints was ap-

Alexandra L. Uitdenbogerd. 2010. Fun with Filtering French. In *Proceedings of Australasian Language Technology Association Workshop*, pages 58–66

plied to the results of queries to a concordancer to make the query results easier for learners to read (Wible et al., 2000).

In this paper, we explore a corpus consisting of several classic French texts, with the goal of determining the feasibility of finding reading material using strict criteria for lexical content, and with some exploration of grammatical complexity. Exploiting the considerable overlap in language pairs, such as French and English, due to their cognate content, provides a substantially larger pool of reading resources than if cognates are ignored.

2 Related Research

Early studies in readability measurement largely led to formulae that consisted of two main factors of readability: lexical and grammatical. The lexical difficulty is often approximated by word length in terms of the number of syllables or characters (Klare, 1974). Alternatively, the presence or absence of a word in a list determined its difficulty (Chall and Dale, 1995). Grammatical complexity was usually modelled by a measure of sentence length Klare (1974).

Recent years have seen an increase in output specifically on automated readability measurement for text retrieval. We discuss some contributions below.

Researchers involved with the REAP project have developed a system for delivering reading material of an appropriate level to users, where the material is retrieved from the Web (Collins-Thompson and Callan, 2004a). They have used a range of readability measurement techniques based on statistical models on lexical and grammatical features that predict the grade level of the text (Heilman et al., 2008).

Miyazaki and Norizuki (2008) developed a reading retrieval system more suited to Japanese learners of English, allowing the readability measurement to be learnt from a user's ratings of text.

Tanaka-Ishii et al. (2010) had a novel approach to training a classifier to measure readability. They used precisely two classes, being for easy and hard texts, trained on text for children and adults respectively. Texts are classified as pairs to determine which is more difficult.

Little work has been published specifically for French readability as a foreign language. One recent development on readability of French as a foreign language uses a machine learning approach applied to a range of features, including the verb tenses occurring in the text (François, 2009).

The only work on assessing the suitability of on-line text for learners that we know of is ours (Uitdenbogerd, 2006), in which we concluded that the percentage of web-based text (in the English language) that is in a useful range for learners is between 8 and 19%. We are unaware of any that look at extracts of larger texts.

3 Experiments

In this current piece of work we are exploring the potential of filtering text based on strict lexical and grammatical criteria within the context of two languages that have a large set of exact cognates.

Our research questions were:

- *What is the frequency distribution of distinct sentence structures in text?* If there are frequent patterns, then these could be the basis for grammatical study for beginners in the language. They could also form part of the criteria for selecting text on readability. On the usual observation that shorter sentences are more readable, we were interested in discovering whether there were useful portions of natural texts to be found that could be used for reading practice at the early stages.

- *What proportion of a French text consists of French-English cognates and vice versa?* In this work, we restrict ourselves to words that have identical spelling in both languages. Accented words were not included. As the presence of cognates allow people to understand more of a text than when there are no cognates, we wanted to estimate the cognate content of the text. When combined with a small vocabulary of frequent words, the coverage of the text should be substantially greater. This idea was again to be applied to the process of extracting potential reading material.

The tools that we used for our experiments include Tree Tagger (Schmid, 1994), and a first approximation to a cognate list using the intersection of the English and French lexicons provided as Tree Tagger parameter files for these languages.

3.1 Sentences in French Literature

Initial work was attempted with on-line French literature. One example of a long written work that is available is *Les Trois Mousquetaires* (The Three

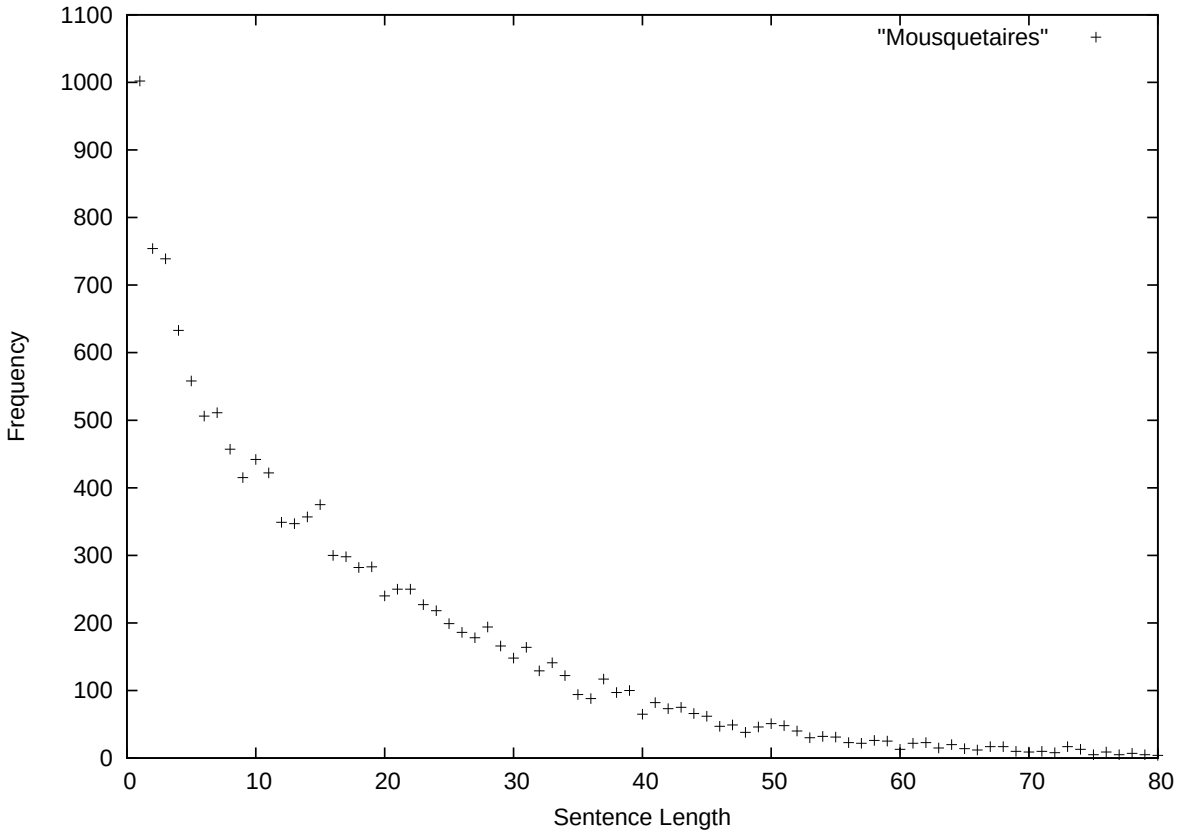


Figure 1: The frequency of each sentence length up to 80 in the text *Les Trois Mousquetaires*.

Musketeers) by Alexandre Dumas. The frequency of sentences of each length are shown in Figure 1.

We first determined the frequency of different sentence structures in the text, as described by the part-of-speech tags reported by Tree Tagger. Naturally, the majority of sentences are unique when compared in this way, however, some structures were frequent, particularly for very short utterances.

There were 11,539 different sentence structures found, of which 11,166 occurred precisely once. This figure is an overestimate, as the text file was not cleaned up, and there were a significant number of errors in the tagged data. The first four sentence types consisted of a single word, being either an interjection (263), a name (484), or a noun (65). (A sample extract from *Les Trois Mousquetaires* of single word sentences is shown in Figure 2. Figure 3 shows a sequence of single word sentences that occurs in the story.) Many frequent structures are such expressions as “said Aramis”, which are usually tagged as separate sentences to the utterance of the character. The most frequently occurring sentence of length 4 is this type of phrase: “s’écria d’Artagnan”, which occurs 14 times in the text. Similarly for *Le Petit Prince* (The Little Prince), the most frequent sentence structure beyond single-word interjections is the phrase “dit le petit prince” (“said the little prince”), which occurs 10 times.

Ah ! , – Non . , – Porthos ? , – Non . , – Aramis ? , – Non . , – Oh ! , – Oh ! , – Oh ! , – Comment ! , – Oh ! , – Quoi ? , – Pardieu ! , – Oh ! , – Silence ! , – Quoi ? , – Ah ! , – Allez . , – Ah ! , – Silence ! , – Silence ! , – Silence ! , ” Oh ! , – Ah ! , Ah ! , ” Assez ? , Allez . , – Qui ? , – Peut-être . , – Ah ! , ah ! , Pourquoi ? , – Hol ! , – Parle . ,

Figure 2: Some single-word sentences in *Les Trois Mousquetaires*.

The first interesting repeated sentence structure in Le Petit Prince is “PRO:PER ADV VER:pres ADV SENT”, which occurs 5 times[1]. One example is “Elle ne change pas.” (She doesn’t change.) By contrast, the positive version of this sentence structure doesn’t occur at all.

When grouping several French texts together

[1]PRO:PER represents personal pronouns, ADV adverbs, VER:pres present tense verbs, and SENT end of sentence

435 – Aramis !
436 – Porthos !
437 – Eh !
438 Messieurs !
439 Messieurs !

2369 – Arrêté !
2370 Athos !
2371 arrêté !
2372 pourquoi ?

Figure 3: Some single-word sentence sequences in *les Trois Mousquetaires*. Each sentence is preceded by its sentence number.

(2.8 million words, including some noise) before analysis the trend is similar, in that the most common sentence structure is the single-word utterance. Then there are two word sentences occurring frequently, such as VER:pres PRO:PER SENT, which represented "Pardonnez-moi!" (pardon me), despite being an imperative rather than simple present tense. There are several recurring sentence structures of about four words, such as "PRO:DEM VER:pres DET:ART NOM SENT", which includes "c'est l'amour" ("It's love"), so a larger collection can provide some useful simple examples for study[2]. Table 2 shows the frequency of particular sentence structures of different lengths in a corpus consisting of approximately 2.8 million words from French literature.

Our earlier work in French readability for readers with an English-speaking background revealed that average sentence length was a better indicator of readability than other standard measures. On this premise, we attempted to find extracts with a very low average sentence length. Figure 4 shows an extract from *Les Trois Mousquetaires* with a maximum sentence length of 4, as well as the first extract of at least 100 words, which is retrieved when the maximum sentence length is increased to 10, and has an average sentence length of about 5.

[2]PRO:DEM for demonstrative pronouns, DET:ART for articles, and NOM for nouns

3734 – Connaissez -vous Athos ?
3735 – Non .
3736 – Porthos ?
3737 – Non .
3738 – Aramis ?
3739 – Non .
3740 Quels sont ces Messieurs ?
3741 – Des mousquetaires du roi .

– Ah ! fit Rochefort avec un sourire , voilà un hasard bien heureux ! et qui satisfera Son Eminence ! L' avez-vous prévenue ?

– Je lui ai écrit de Boulogne . Mais comment êtes-vous ici ?

– Son Eminence , inquiète , m'a envoyé à votre recherche .

– Je suis arrivée d'hier seulement .

– Et qu'avez-vous fait depuis hier ?

– Je n'ai pas perdu mon temps .

– Oh ! je m'en doute bien !

– Savez-vous qui j'ai rencontré ici ?

– Non .

– Devinez .

– Comment voulez-vous ? ...

– Cette jeune femme que la reine a tirée de prison .

– La maîtresse du petit d'Artagnan ?

– Oui , Mme Bonacieux , dont le cardinal ignorait la retraite .

Figure 4: Extracts from *Les Trois Mousquetaires* with a maximum length of 4 and 10 respectively.

Table 1: The 20 most frequently occurring words in French newspapers and their main meanings.

1 to 10	Def.	11 to 20	Def.
de	of	que	that
le	the	dans	in
la	the	il	he/it
et	and	à	at/to
les	the	en	in
des	of the	ne	not
est	is	on	one (pronoun)
un	a/an/one	qui	who
une	a/an/one	au	at/to the
du	of the	se	him-/her-/it-self

3.2 Cognates, Named Entities, and Common Words

Our cognate (and named entity) list consisted of the intersection of the English and French lexicon provided in the parameter files with Tree Tagger. These lexicons contain all the tokens recognised by the tagger in the two languages. Clearly there are some *false friends* in this list, that is, words that look the same in both languages, but have different meanings. In addition, many cognates that are spelt differently were excluded. The initial list for English contained 358,097 words once duplicates were removed, and the French list had 475,209. Taking the intersection of the two lists gave a list of 17,908 words. Some false friends are very frequent in French. We determined the frequency of each cognate in *Les Trois Mousquetaires* and found that the majority of the highest ranked terms were either false friends, or were included due to French phrases that occur in English (eg. "fait accompli" and "laissez faire"). Highest ranked false friends in our list included: ment, pour, dans, tout, comme, plus, nous, quel, amis, fait, tend, main, voir, faire, jour, deux, ours, part, dire, sent, rend, and fort. The interjections "Ah" and "Oh" were not in the cognate list, so these were added manually.

In addition to the cognate list, a list of the 20 most frequent words in French newspapers (according to Crystal (1987), and listed in Table 1) was included in the "known" words to test the extreme case of a complete beginner. Named entities from Tree Tagger were also used in the list of permitted words. To this list we added the names of the characters from *Les Trois Mousquetaires* (Aramis, Porthos, d'Artagnan), as they were missing. Note that this procedure is to test the feasibility of the concept of retrieving useful extracts for learners, not a recommended technique for cognate generation. However, our observations discussed later provide ideas for future automation of cognate detection.

Un serpent !

Les provisions !

il est impossible !

est il possible ?

Un voyage sans fatigue et sans danger !

Impossible de continuer le commerce .

Figure 5: Some cognate-filtered sentences in the collection of several French texts, using the most frequent 125 words that occurred in the most frequent 200 words of *Les Trois Mousquetaires* and *Notre Dame de Paris*.

Using the above list provides many sentences (1409 for the larger collection, including duplicates), and some sentence sequences. On our larger collection we found 101 short sequences, including the following short fragment from *Les Misérables* that would be very easy for a beginner with English background to read:

141780 Une barricade !

141781 Ah !

141782 le bandit !

Expanding the list of frequent words to 125 (based on the most frequent 200 tokens occurring in *Les Trois Mousquetaires* and *Notre Dame de Paris*), which is approximately the size of the smallest vocabulary of published readers, a larger set of sentences is retrieved. Examples from the full collection of texts are shown in Figure 5.

Using the same level of filtering while including sentences that have at least 90% of the words in the lists, the sentence filter produces more substantial extracts. Examples are shown in Figure 6.

We calculated some general statistics to estimate the proportion of cognates in French text, as well as that of highly frequent words. Table 3 shows that based on our rough method of identifying cognates, French texts tend to consist of approximately 10% cognate content. The 20 most frequent words make up approximately 26% of the text.

270 – Ah !

271 fit d’ Artagnan .

272 – Non ; elle vous a été prise .

273 – Prise !

274 et par qui ?

887 – C’ est avec Monsieur que je me bats , dit Athos en montrant de la main d’ Artagnan , et en le saluant du même geste .

888 – C’ est avec lui que je me bats aussi , dit Porthos .

13007 Tout à coup elle jeta un grand cri de joie et s’élança vers la porte , elle avait reconnu la voix de d’ Artagnan .

13008 ” D’ Artagnan !

13009 d’ Artagnan !

Figure 6: Some cognate-filtered sentence sequences in *Les Trois Mousquetaires*.

4 Discussion

It is clear from the relative lack of repetition of sentence structures beyond those of fewer than 5 words, that either more sophisticated summaries of sentences would be required for use as sentence examples, or very large corpora. Using a chunking phase before grouping sentences may provide larger sets of related examples. The use of n-grams of different lengths would also allow learners to observe patterns of interest. For example, to better understand how adjectives are placed in French, users can look at occurrences of “ADJ NOM” as well as “NOM ADJ”.

Given that for English-speaking readers of French a good estimate of French readability is the average sentence length of the text, there is considerable scope for finding suitable extracts for reading. Filtering text based on sentence length provided extracts for reading practice that have an average sentence length of 5.

The frequency of some false friends in the cognate list suggests a simple automated technique would be to compare the relative frequency of the words in each language. Where there is a large discrepancy (for example, “aura”, which means“will have” in French), the word is highly likely to be a false friend rather than a cognate.

Applying our filter based on exact cognates, very frequent words and named entities allowed numerous sentences to be found in the corpus. Relaxing these requirements slightly by allowing some unknown words can produce extracts consisting of several sentences for reading — enough to get a sense of the moment in the story, but not as long as the shortest published stories for beginners in a language (about 75 words). Our cognate list was very restrictive in that it required words to have the exact same spelling in both languages (or as a related word, such as “arriver”, meaning “to arrive” in French). It is expected that allowing accents, typical variants such as the presence or absence of the letter “e” as a suffix, and common verb endings, will increase the size and quantity of extracts. Applying the filter to much larger bodies of text, such as found on the Web should also result in considerably more material being retrieved. Our previous work on measuring the readability of web text showed that a significant portion (8–19%) of web documents had the same readability range as stories published for those learning English Uitdenbogerd (2006). While cultural differences may mean that the range of readability of French differs from English on the web[3], we remain optimistic that many extracts can be retrieved that conform to these very strict criteria.

It should also be noted, that the texts used in the present study are relatively difficult to read. Texts written specifically for children would have smaller vocabularies, and translations into French (from English) would be likely to have larger cognate content.

Our earlier work on readability in French (Uitdenbogerd, 2005) demonstrated that sentence length was as good, if not better than the commonly used readability measures for predicting text difficulty where the person reading has an English-speaking background, and the language being read is French. Incorporating a measure of the cognate content was an even more reliable predictor of readability. The texts studied varied widely in their cognate rate, with some texts written specifically for people with an English-speaking background exploiting cognates. It might be expected that the more technical the text,

[3] An example of a relevant cultural difference is that there are many classic novels written in English for children, but none in French until relatively recently

the more common words there will be between French and English, however, our results in Table 3 showed a fairly consistent level of cognates in text. In contrast, the manual count of cognates in the samples of approximately 100 words used in Uitdenbogerd (2005) revealed a range from 5 to 42%, however, the two texts with the highest cognate count were specifically written for French learners with an English-speaking background, in the early stages of learning. When only texts written by and for native speakers of French are considered, the range was 7 to 12%, which doesn't differ too much from our estimate in the present study.

Studies in people's ability to predict the meanings of words from their context indicate that a knowledge of 95% of the words in the text are needed for comprehension (Ghadirian, 2002). Using this figure as a basis, it has been concluded that a vocabulary of 5000 (relatively frequent words) is required to be able to comfortably read any text in a given language (Groot, 2000), a figure we confirmed with our study of French texts (Uitdenbogerd, 2005). The cognate content in French texts probably reduces this figure somewhat. We can expect that about 10% of the infrequent words would be known as cognates. So, using *Les Trois Mousquetaires* as an example, the 95% threshold assuming no knowledge of cognates requires a vocabulary of about 3400 frequent words. Assuming 10% of the remaining vocabulary is known, this figure drops to about 3120. However, at the early stages of learning, when a person's vocabulary is small, the gains from cognates are greater. For example, a knowledge of 20 words gives a coverage of about 31% (when combining the total of all words regardless of their part of speech), whereas the additional cognate knowledge increases that coverage to 38%.

5 Conclusion and Future Work

We demonstrated potential techniques for identifying short extracts from French literature based on lexical or grammatical criteria to allow reading practice at the very early stages without the intensive work of translation. Experiments are currently underway that attempt to apply the same technique in reverse for English web documents, that is, applying strict lexical filters based on a small frequent words list and a large list of cognates.

Future work will include incorporating inexact cognate detection (Kondrak, 2001; Inkpen et al., 2005) to allow words with slightly different spelling to be found, and more sophisticated grammatical matching. Also, the idea of determining whether a word with the same spelling in both languages is a cognate or not based on its relative frequency and other available data, such as POS tags, will be explored.

References

Bell, T. (2001). Extensive reading: speed and comprehension. *The Reading Matrix*, 1(1).

Bormuth, J. R. (1966). Readability: a new approach. *Reading Research Quarterly*, 1:79–132.

Chall, J. S. and Dale, E. (1995). *Readability revisited: the new Dale-Chall readability formula*. Brookline Books, Massachusetts, USA.

COE (2003). Common European framework of reference for languages: Learning, teaching, assessment. `http://www.coe.int/T/DG4/Linguistic/CADRE_EN.asp#TopOfPage`. Accessed 8 September, 2006.

Collins-Thompson, K. and Callan, J. (2004a). Information retrieval for language tutoring: An overview of the REAP project. In *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, Sheffield, UK. Poster.

Collins-Thompson, K. and Callan, J. (2004b). A language modeling approach to predicting reading difficulty. In *Proceedings of the HLT/NAACL 2004 Conference*, pages 193–200, Boston.

Cornaire, C. M. (1988). La lisibilité: Essai d'application de la formule courte d'henry, au français langue étrangère. *Canadian Modern Language Review*, 44(2):261–273.

Cowling, S., editor (1982). *Bibliobus*. Mary Glasgow Publications Ltd, London, UK.

Crystal, D. (1987). *The Cambridge encyclopedia of language*. Cambridge University Press, New York, NY, USA.

François, T. L. (2009). Combining a statistical language model with logistic regression to predict the lexical and syntactic difficulty of texts for FFL. In *Proceedings of the EACL Student Research Workshop*, pages 19–27, Athens, Greece.

Association for Computational Linguistics, Association for Computational Linguistics.

Ghadirian, S. (2002). Providing controlled exposure to target vocabulary through the screening and arranging of texts. *Language Learning and Technology*, 6(1):147–164.

Granowskey, A. and Botel, M. (1974). Background for a new syntactic complexity formula. *The Reading Teacher*, 28:21–35.

Groot, P. J. M. (2000). Computer-assisted second language vocabulary acquisition. *Language Learning and Technology*, 4(1):60–81.

Heilman, M., Collins-Thompson, K., and Eskenazi, M. (2008). An analysis of statistical models and features for reading difficulty prediction. In *The 3rd Workshop on Innovative Use of NLP for Building Educational Applications*.

Inkpen, D., Frunza, O., and Kondrak, G. (2005). Automatic identification of cognates and false friends in French and English. In Mitkov, R., editor, *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 251–257, Borovets, Bulgaria.

Katz, I. R. and Bauer, M. I. (2001). Sourcefinder: Course preparation via linguistically targeted web search. *Educational Technology and Society*, 4(3):45–49.

Klare, G. R. (1974). Assessing readability. *Reading Research Quarterly*, X:62–102.

Kondrak, G. (2001). Identifying cognates by phonetic and semantic similarity. In Knight, K., editor, *Proceedings of NAACL 2001: 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, volume 2, pages 103–110, Pittsburgh, PA, USA. NAACL.

Miyazaki, Y. and Norizuki, K. (2008). Developing a computerized readability estimation program with a web-searching function to match text difficulty with individual learner's reading ability. In *WorldCALL*.

Nilsson, K. and Borin, L. (2002). Living off the land: The web as a source of practice texts for learners of less prevalent languages. In *Proceedings of LREC 2002, Third International Conference on Language Resources and Evaluation*, pages 411–418, Las Palmas. ELRA.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In Somers, H. and Jones, D., editors, *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.

Schwarm, S. E. and Ostendorf, M. (2005). Reading level assessment using support vector machines and statistical language models. In *Proceedings of the Association for Computational Linguistics*.

Si, L. and Callan, J. (2001). A statistical model for scientific readability. In Liu, L. and Grossman, D., editors, *Proc. International Conference on Information and Knowledge Management*, volume 10, pages 574–576, Atlanta, Georgia, USA. ACM, ACM.

Tanaka-Ishii, K., Tezuka, S., and Terada, H. (2010). Sorting texts by readability. *Computational Linguistics*, 36(2):203–227.

Uitdenbogerd, A. L. (2003). Using the web as a source of graded reading material for language acquisition. In Zhou, W., Nicholson, P., Corbitt, B., and Fong, J., editors, *International Conference on Web-based Learning*, volume 2783 of *Lecture Notes in Computer Science*, pages 423–432, Melbourne, Australia. Springer.

Uitdenbogerd, A. L. (2005). Readability of French as a foreign language and its uses. In Turpin, A. and Wilkinson, R., editors, *Australasian Document Computing Symposium*, volume 10.

Uitdenbogerd, A. L. (2006). Web readability and computer-assisted language learning. In Cavedon, L. and Zukerman, I., editors, *Australasian Language Technology Workshop*, pages 99–106.

van Oosten, P., Tanghe, D., and Hoste, V. (2010). Towards an improved methodology for automated readability prediction. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Wible, D., Kuo, C.-H., Chien, F.-Y., and Wang, C. (2000). Toward automating a personalized concordancer for datadriven learning: a lexical difficulty filter for language learners. In Ketteman, B. and Marko, G., editors, *Conference on Teaching and Language Corpora*, volume 4, pages 147–154, Graz. Rodopi.

Table 2: Example sentence structures of each length in the corpus of French literature. Note that frequencies for items are approximate due to the inaccuracy of the tagger and the noise in the data. Note also that some tags are incorrect, such as the sentence of length 2, which should really have been labelled as “imperative” instead of “present” tense.

Len	Structure Example Translation	Freq
1	INT SENT Ah! Ah!	943
2	VER:pres PRO:PER SENT Sauvez-moi! Save me!	204
3	VER:pres DET:ART NOM SENT Videz le vase! Empty the vase	123
4	PRO:PER VER:simp DET:ART NOM SENT Il leva les yeux. He raised his eyes. 73	
5	PRO:PER ADV PRO:PER VER:pres ADV SENT Elle ne la sait pas. She doesn’t know her/it.	51
6	PRO:PER VER:simp DET:ART NOM PRP NOM SENT Elle resta un moment sans parler. She remained speechless for a moment.	15
7	PRO:PER ADV PRO:PER VER:pres ADV PRP NOM SENT Il n’ y a pas de jardin. There is no garden	11
8	DET:ART NOM VER:pres DET:ART NOM PRP DET:ART NOM SENT La philosophie est le microscope de la pensée. Philosophy is the microscope of thought	9
9	PRO:PER VER:impf DET:ART NOM KON PRO:PER VER:impf DET:ART NOM SENT Elle était la lumière et il était l’ ombre She was the light and he was the shade	4
10	DET:ART NOM VER:impf DET:ART NOM PUN DET:ART NOM VER:impf DET:ART NOM SENT Les assaillants avaient le nombre ; les insurgés avaient la position. The assailants had the numbers; the insurgents had the position.	3

Table 3: Statistics of occurrence of cognates (including named entities and false friends), and highly frequent words for French texts

Text	Types	Tokens	Cognates	Top 20 News Words
Le Petit Prince	2,614	16,484	1,773 (11%)	4,214 (26%)
Les Méditations	3,040	29,976	3,030 (10%)	9,111 (30%)
Les Trois Mousquetaires	16,029	235,056	23,137 (9.8%)	61,439 (26%)
Notre Dame de Paris	18,100	176,245	18,451 (10%)	51,880 (29%)

Parser Features for Sentence Grammaticality Classification

Sze-Meng Jojo Wong
Centre for Language Technology
Macquarie University
Sydney, NSW, Australia
sze.wong@mq.edu.au

Mark Dras
Centre for Language Technology
Macquarie University
Sydney, NSW, Australia
mark.dras@mq.edu.au

Abstract

Automatically judging sentences for their grammaticality is potentially useful for several purposes — evaluating language technology systems, assessing language competence of second or foreign language learners, and so on. Previous work has examined parser 'byproducts', in particular parse probabilities, to distinguish grammatical sentences from ungrammatical ones. The aim of the present paper is to examine whether the primary output of a parser, which we characterise via CFG production rules embodied in a parse, contains useful information for sentence grammaticality classification; and also to examine which feature selection metrics are most useful in this task. Our results show that using gold standard production rules alone can improve over using parse probabilities alone. Combining parser-produced production rules with parse probabilities further produces an improvement of 1.6% on average in the overall classification accuracy.

1 Introduction

Automatically judging sentences for their grammaticality has been a long-standing research problem within the natural language processing community. The ability of distinguishing grammatical sentences from ungrammatical ones has many potential applications, which include evaluating language technology systems such as natural language generation (Mutton et al., 2007) and machine translation (Gamon et al., 2005), as well as assessing language competence of second language or foreign language learners (Brockett et al., 2006; Gamon et al., 2008; Han et al., 2010).

Various approaches have been proposed in the past to address this typical classification problem. A number of these existing studies attempt to exploit some form of 'parser byproduct' as classification features for machine learning: for instance, (log) probability of a parse tree, number of partial (incomplete) parse trees, parsing duration, and such (Mutton et al., 2007; Sun et al., 2007; Foster et al., 2008; Wagner et al., 2009). The aim of this paper is to examine whether the primary output of a parser contains useful information for this classification problem; we characterise this information by the CFG production rules embodied in a parse. The intuition is that particular production rules might be strongly characteristic of ungrammatical sentences, and that looking at individual rules might provide clues that are aggregated out in measures such as the parse tree probability.

We carry out experiments to test this intuition, using the Penn treebank and an artificially created ungrammatical version created by Foster et al. (2008). This allows a large amount of data to be used for classification, embodying a controlled ungrammaticality that is suitable for preliminary work in this direction; it is for similar reasons that construction of erroneous corpora has become a more prominent line of computational linguistic research lately (Foster and Andersen, 2009; Han et al., 2010; Dickinson, 2010).

The present study is carried out in two stages. In the first stage, following Foster et al. (2008), we induce three models from a probabilistic parser by re-training it with a (presumably) grammatically well-formed corpus, a grammatically ill-formed corpus, and a mixed corpus consisting of both grammatical and ungrammatical sentences. The model which outperforms the others is then used for all the subsequent parsing tasks. In the next stage, we utilise the outputs of the parser from the first stage and a parser trained on only grammatical text for sentence grammaticality classification, in which two classes of feature are to be examined — parse probabilities based on the parser outputs

Sze-Meng Jojo Wong and Mark Dras. 2010. Parser Features for Sentence Grammaticality Classification. In *Proceedings of Australasian Language Technology Association Workshop*, pages 67−75

and production rules based on both the gold standard and the parser outputs. A number of feature selection metrics are explored to obtain a set of discriminative parse rules for classifying English sentences based on their grammaticality.

The remainder of this paper is structured as follows. We review some related work in Section 2. In Section 3, we detail the experimental settings and the feature selection metrics. Sections 4.1 and 4.2 then present the parsing results and the classification results, respectively; followed by discussion in Section 5.

2 Related Work

In this section, we briefly review some of the related studies on judging sentence grammaticality. We also discuss some of the recent work concerning the construction of erroneous corpora for the purpose of grammatical error detection.

2.1 Sentence Grammaticality Judgement

In some ways, the present study is an extension of the work presented by Foster et al. (2008). Their intention is to improve the robustness of a probabilistic parser which might not be initially designed to handle ungrammatical sentences. Retraining on both grammatical and ungrammatical sentences enabled a parser to parse ungrammatical sentences at a relatively satisfactory level without compromising its initial performance on grammatical sentences. To attain an optimal parsing accuracy, the parser output of a sentence is chosen according to the highest parse probability of the most likely parse tree returned by the three induced models of the Charniak and Johnson reranking parser (Charniak and Johnson, 2005) trained across three different corpora — grammatical, ungrammatical, and a combination of both. Their experiments show that their parse probability-based classifier which can be considered as an integration of two parsers (one trained on grammatical data and the other trained on some ungrammatical data) is able to parse ungrammatical sentences better than the original parser trained exclusively on a grammatical corpus. The grammatical corpus used by Foster et al. (2008) is the Wall Street Journal (WSJ) treebank, and the ungrammatical version is one that they generated (see Section 2.2).

In a related work (Wagner et al., 2009), a number of parser outputs are utilised for classifying a sentence as to whether it is grammatical or ungrammatical. In addition to the widely used part-of-speech n-grams, they made use of two types of parsers, each based on a different grammar — the precision grammar parser (XLE parser) and the probabilistic parser (Charniak and Johnson parser). Features extracted from the probabilistic parser, which include the differences in log probabilities of parse trees and the structural differences between parse trees, are better discriminants as compared to both the n-gram features and the parser statistics outputs obtained from the precision-grammar-based parser. The overall accuracy achieved is within the range of 65-75% by using the combination of all the feature sets.

A similar idea had been used by Mutton et al. (2007), who discovered that parser outputs can be used as metrics for assessing generated sentence fluency. The underlying idea is that a poorer performance of the parser on one sentence relative to another might indicate that there is some degree of ungrammaticality or disfluency in the former. Outputs from multiple parsers, such as log probability of the most likely parse, number of partial parse trees, and number of invalid parses were investigated. The combination of multiple parser outputs outperforms individual parser metrics.

Parse probability was also used by Sun et al. (2007) for machine learning based classification. There, the type of feature they term 'labelled sequential patterns' like non-contiguous n-grams, proves more important for sentence grammaticality classification with an accuracy rate of over 80%. To provide useful feedback to learners of English as a Second Language (ESL), two English learner corpora are used — Japanese and Chinese.

The techniques of phrase-based SMT have been adapted for grammaticality judgement on ESL sentences as well. Brockett et al. (2006) treat error correction as a translation task, and solve it by using the noisy channel model. They made use of the Chinese Learner Error Corpus as a template for training data creation; but also needed large sets of parallel corpora.

2.2 Erroneous Corpora Construction

Large-scale ungrammatical corpora are crucial for research concerning grammaticality judgement, in particular for classification training. Recently, a number of pieces of corpus-based research have been undertaken to collect authentic errors as well as to generate synthetic errors for this purpose.

The thesis work of Foster (2005) involved an extensive analysis of grammar error types across a 20K word corpus consisting of newspaper articles, emails, Internet forum postings, and academic papers. This led to the development of an ungrammatical version of the WSJ treebank according to a model derived from this analysis (Foster, 2007); this also included a procedure for constructing trees for the ungrammatical sentences. Ungrammatical sentences are constructed by introducing errors into the original (grammatical) WSJ sentences through the operations of word insertion, substitution, and deletion. Each ungrammatical sentence is then tagged with the gold standard parse tree, a transformation of the original parse tree of its grammatical counterpart with the *intended* meaning remained intact. The types of errors introduced include missing word, extra word, real-word spelling, agreement, and verb form: according to Foster, these comprise 72% of the analysed errors. Subsequently, Foster and Anderson (2009) developed an automated error generation tool that can be applied to any text.

Okanohara and Tsujii (2007) attempt to produce grammatically ill-formed sentences termed as *pseudo-negative* examples which are not representative of authentic errors but more like machine translation outputs. Han et al. (2010), construct an error-annotated English corpus comprised of texts written by Korean learners of English and demonstrate that classifiers trained on error-annotated data outperform those that trained exclusively on well-formed data produced by native English speakers. Dickinson (2010), in other recent corpus-based research aiming to address morphological errors found in highly inflecting languages, creates learner-like morphological errors from a segmented lexicon.

3 Experimental Setup

We first describe the data used, and then the consequent re-training of the parser in the first stage of the experiments. We follow that with a description of the feature selection metrics for the classification experiments in the second stage.

3.1 Grammatical and Ungrammatical Corpora

Given that the goal of the present study is to distinguish between grammatical and ungrammatical sentences, two corpora are needed. For the grammatical sentences, we take the WSJ treebank by making the assumption that they are grammatically well-formed. On the other hand, the ungrammatical sentences are obtained from noisy (distorted) versions of WSJ created by Foster (2007) and used in Foster et al. (2008). As mentioned earlier, the grammatically ill-formed WSJ sentences were generated by introducing errors to the initially well-form WSJ sentences through the operations of insertion, deletion, and substitution.

It should be noted that there are two noisy versions of WSJ. The first is a complete parallel of the original WSJ which consists of 24 sections (from Section 0 to Section 23) and the second set is a much smaller one covering only 6 sections (including Section 0, Section 2-5, and Section 23). The latter is considered noisier data since the sentences were generated by applying the error generation procedures to the first set of ungrammatical WSJ sentences. Hencefore, we denote the three sets of WSJ treebank as follows: *PureWSJ* — the original WSJ; *NoisyWSJ* — the first set of less noisy WSJ; and *NoisierWSJ* — the second set of more noisy WSJ.

In Figure 1 we give examples of sentences with trees generated by insertion and deletion, and their grammatical counterparts.

3.2 Re-training of Parsers

In order to enable a parser to be able to parse ungrammatical sentences, we re-train a probabilistic parser on both grammatical and ungrammatical corpora. This idea is adopted from Foster et al. (2008). By and large, we replicate the experiments conducted in Foster et al. (2008) with the exception that the parser used in our study is the Stanford Parser (Klein and Manning, 2003), chosen for ease of re-training.

In this first stage, we conduct five experiments to re-train the Stanford Parser to induce a more robust parser capable of parsing both grammatical and ungrammatical sentences. In the first three experiments, three models of parser are induced by training on three different sets of corpora — first on the original WSJ (*PureWSJ*); second on the noisy WSJ (*NoisyWSJ*); and third on both the original and noisy WSJ (*PureWSJ* plus *NoisyWSJ*). We denote these three parser models as *PureParser*, *NoisyParser*, and *MixedParser*. In order to gauge its ability of parsing both grammatical and ungrammatical sentences, each of these models is

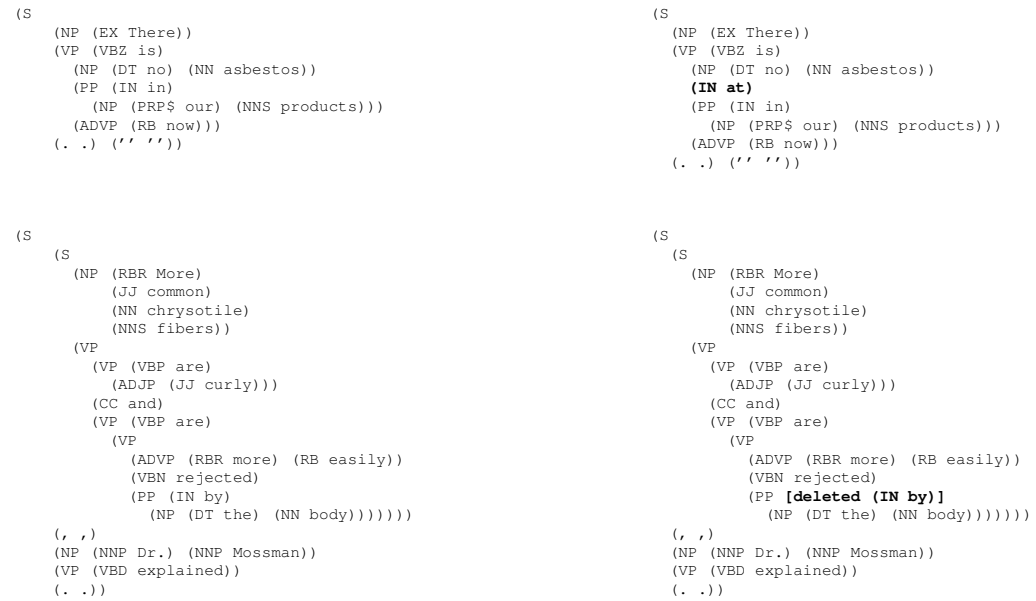


Figure 1: Grammatical (left) and ungrammatical (right) versions of sentences, illustrating insertion errors (top) and deletion errors (bottom)

then evaluated against the three sets of WSJ (i.e. *PureWSJ*, *NoisyWSJ*, and *NoisierWSJ*) using the labelled f-score measure.

The last two experiments can be viewed as the use of an integrated parser, in which each test sentence is parsed by two types of parser — one trained exclusively on grammatical data (i.e. *PureParser*) and the other trained on some ungrammatical data (i.e. either *NoisyParser* or *MixedParser*). The best parse is selected by choosing the one with the higher parse probability. Hence, *PureParser* is integrated with *NoisyParser* for the fourth experiment and with *MixedParser* for the last experiment. (It should be noted that all trainings are performed on Section 2 to Section 21 while all testings are on Section 0.)

3.3 Sentence Classification

This second stage is the core of the present study where we experiment with production rules as features for sentence grammaticality classification. Apart from the parse probabilities returned together with the parse trees, we extract the individual production rules (from either the gold standard or the parse trees) and their corresponding rule probabilities (from parse trees) as classification features. The use of the gold standard is a kind of oracle, to assess the impact of parser inaccuracies. An example with a grammatical-ungrammatical pair is given in Figure 2. We explore various feature selection metrics to obtain a set of production rules for classifying grammatical and ungrammatical sentences.

Parse probability features For the feature class of parse probabilities, we perform similar procedures as in the last two experiments in the first stage. As before, each sentence (be it for training or testing) is parsed with two types of parser — *PureParser* and either *NoisyParser* or *MixedParser*. The parse probability returned by each parser type is used as a classification feature. Therefore, there are only two feature values for this feature class — the parse probability from *PureParser* and the parse probability from either *NoisyParser* or *MixedParser*. A classifier consisting only of these two features is our baseline.

Production rule features We first parse the sentences (for both training and testing) by using the best performing parser induced from the five experiments in the first stage. Production rules are then extracted automatically from both the gold standard and the parser outputs. Various feature selection metrics are used to select a set of discriminative parse rules as classification features.[1] The metrics we use are as follows (with r representing a production rule and c a class, i.e. gram-

[1]There were approximately 26K unique production rules drawn from the training data that could possibly be used as classification features. However, our machine learner described below could not handle this large set of features; but in any case, further experiments showed that a larger feature set resulted in a monotonically lower accuracy.

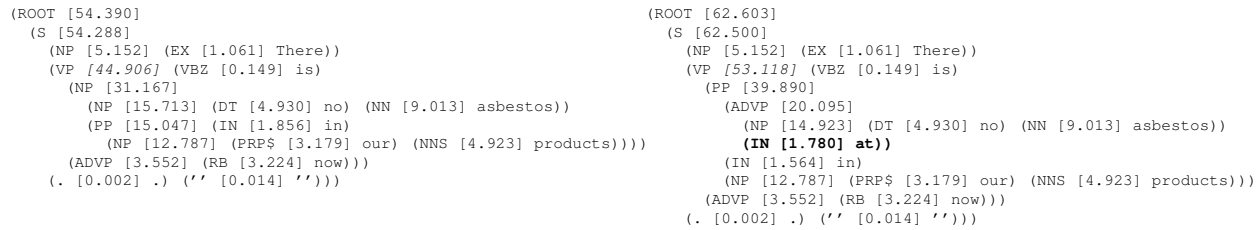


Figure 2: Parser outputs for original (left) and insertion-error (right) variants, annotated with log probabilities for each production rule

matical or ungrammatical):

- *Frequency* (FREQ): We take the n most frequently occurring parse rules within the grammatical corpus and the ungrammatical corpus, where $n \in \{50, 100, 2500\}$. Feature values are the relative frequency of each parse rule within a sentence and also the binary value of their presence or absence.

- *Ratio* (RATIO): We take the ratio of the number of occurrences of a parse rule in the grammatical corpus to the number of occurrences of that rule in the ungrammatical corpus. We pick the 50 parse rules with the highest ratio and another 50 parse rules with the lowest ratio as features. Feature values are of binary type.

- *Mutual information* (MI): We calculate the mutual information between a parse rule and each class (i.e grammatical and ungrammatical). The 100 parse rules with the highest mutual information are selected as features with binary-typed values. We adopt the formula from Yang and Pedersen (1997):

$$MI(r, c) = \log \frac{\Pr(r \wedge c)}{\Pr(r)\Pr(c)} \quad (1)$$

- *Information gain, version 1* (IG-FREQ): We pick the 100 and 500 rules with the highest information gain as features. The formula is again adopted from Yang and Pederson (1997), with $m = 2$.

$$\begin{aligned} IG(r) = & -\sum_{i=1}^{m} \Pr(c_i) \log \Pr(c_i) \\ & + \Pr(r) \sum_{i=1}^{m} \Pr(c_i|r) \log \Pr(c_i|r) \\ & + \Pr(\bar{r}) \sum_{i=1}^{m} \Pr(c_i|\bar{r}) \log \Pr(c_i|\bar{r}) \end{aligned} \quad (2)$$

- *Information gain, version 2* (IG-PROB): In addition, we attempt a different way to calculate the information gain of a parse rule, where the probability of each parse rule $\Pr(r)$ is estimated based on its rule probabilities extracted from the parse trees instead of its occurrence in the corpora. Hence, $\Pr(r)$ is the sum of all the parse probabilities of a parse rule divided by the sum of the parse probability of all the parse trees. All feature values are of binary type. The intuition is that it might not be particular production rules that are characteristic of grammaticality, but their probability: for example, ungrammatical parses might have more unlikely rules. As an illustration, the grammatical tree in Figure 2 (left) has log prob 44.906 at the highest VP node, while the ungrammatical tree (right) has log prob 53.118; notwithstanding the contribution of 1.780 from the insertion of the lexical item *at*, there are some unlikely production rules in this subtree of the ungrammatical tree.

- *Bi-normal separation* (BNS): Forman (2003) suggested that this feature selection metric can be competitive with information gain. The metric is defined as below, where *F(x)* = cumulative probability function of a normal distribution:

$$BNS(r, c) = \left| F^{-1}(\Pr(r|c)) - F^{-1}(\Pr(r|\bar{c})) \right| \quad (3)$$

Similarly, the 100 and 500 rules with the highest BNS scores are selected as classification features with binary-typed values.

Besides investigating these five feature selection methods individually, we also explore the effects of their combinations as well as the combination with parse probabilities.

Training set The training set is a balanced set of grammatical and ungrammatical sentences. As mentioned in Section 3.1, the grammatical sentences are adopted from the *PureWSJ*, while the ungrammatical sentences are from the *NoisyWSJ*; both are based on Section 2 to Section 21. There are 79664 sentences in total for training.

Testing set The testing set is also a balanced set of grammatical and ungrammatical sentences. However, we have two sets of testing data. The first set is formed from *PureWSJ* and *NoisyWSJ*,

Exp	Parser	PureWSJ	NoisyWSJ	NoisierWSJ
1	PureParser	85.61	78.42	72.64
2	NoisyParser	84.31	80.32	76.19
3	MixedParser	82.63	78.69	74.25
4	Pure-NoisyParser	85.39	80.43	76.40
5	Pure-MixedParser	85.49	80.04	75.53

Table 1: Parsing results (labelled f-score %) of five experiments on three versions of WSJ Section 0

and the second set is from *PureWSJ* and *NoisierWSJ*; all are based on Section 0. The latter set is used to testify whether the degree of noisiness in the data would have any effects on the classification performance. There are 3840 sentences in total for testing.

Classifiers A support vector machine (SVM) is used for all the classification tasks. We use the online SVM tool *LIBSVM* (Version 2.89), implemented by Chang and Lin (2001). All the classifications are first conducted under the default settings where the radial basic function (RBF) kernel is used. The kernel is further tuned to find the best pair of parameters (C, γ) for an optimal classification model. [2] In addition to SVM, another machine learner — logistic regression — is also examined to study its effects on classification. Here, we use the logistic regression classifier with ridge regularization from WEKA (Version 3.6.1) (Witten and Frank, 2005).

4 Results

4.1 Parsers

In Table 1, we present the parsing results of the five experiments conducted in the first stage where the intention is to induce a more robust parser that can handle ungrammatical sentences without compromising its performance on grammatical ones.

The integrated parser in Experiment 4 — *Pure Parser* integrated with *Noisy Parser* — is able to attain a relatively good parsing performance for ungrammatical data while at the same time maintaining its performance for grammatical data. This parser is therefore the one that was used for all the parsing tasks in the second stage.

[2] As there is no significant difference between the classification results prior to and after tuning, we only report the prior ones. In addition, no other kernels demonstrated better results than the RBF, so we omit these.

Feature	PureWSJ-NoisyWSJ	PureWSJ-NoisierWSJ
Parse Prob	65.42	74.19

Table 2: SVM results (accuracy %) with parse probabilities as features on both NoisyWSJ and NoisierWSJ

Feature (Metrics)	Gold Standard	Parser Output
FREQ	64.35	53.28
RATIO	50.08	50.0
MI	50.0	n/a
IG-FREQ	67.65**	60.67
IG-PROB	n/a	54.22
BNS	63.75	57.58

Table 3: SVM results (%) with parse rules as features on NoisyWSJ — based on top 100 rules from both gold standard and parser outputs

4.2 Classification

4.2.1 Parse Probabilities

For classification, by using just parse probabilities alone as features, we can see that a reasonably good accuracy is achievable (see Table 2). As expected, for more noisy data, their ability to distingush grammatical sentences from ungrammatical sentences is even more prominent — comparing the classification accuracy of 65.42% (*NoisyWSJ*) with 74.19% (*NoisierWSJ*). This classifier is our baseline for the rest of the sentence grammaticality classifications utilising production rules.

4.2.2 Production Rules

As mentioned in Section 3.3, we first examined the production rules extracted from both the gold standard and the parser outputs with five different feature selection metrics. The classification accuracies achieved by using the top 100 rules for the testing of the less noisy ungrammatical data — *NoisyWSJ* — are shown in Table 3.

It appears that standard information gain (IG-FREQ) outperforms the rest of the selection metrics and it is the only one that performs better than parse probabilities if the gold standard parse trees were available (with this result being statistically significant with 95% confidence). [3] It is, however, worth noting that information gain which utilises rule probabilities (IG-PROB) does not turn out to be a better discriminant as compared to information gain (IG-FREQ). Bi-normal separation and frequency are the next potential candidates; but the former is a better choice in the absence of the gold standard. Ratio and mutual information perform no better than chance.

[3] All significance tests are based on the McNemar's test.

Feature (Metrics)	n = 200	n = 500	n = 2500
FREQ	54.84	n/a	54.04
MI	n/a	50.0	n/a
IG-FREQ	n/a	58.72	n/a
BNS	n/a	61.93	n/a

Table 4: SVM results (%) with parse rules as features on NoisyWSJ — based on larger numbers of rules from gold standard

Feature (Metrics)	Gold Standard	Parser Output
IG-FREQ	75.65*	63.44
BNS	71.2	61.51

Table 5: SVM results (%) with parse rules as features on NoisierWSJ — based on top 100 rules from both gold standard and parser outputs

Table 4 presents results showing the impact of using more production rules as selected by the various metrics; and the results were all poorer than using just 100 rules. In view of this poorer result, we subsequently made use of only the top 100 rules for all the subsequent classifications.

Next, we performed testing on more noisy data — *NoisierWSJ* — to see whether the degree of noisiness in data would have any effects on the classification. Not surprisingly, the more noisy data appears to be easier to be distingushed from the grammatically well-formed data (see Table 5). Similarly, standard information gain (IG-FREQ) may perform better than parse probabilities if the gold standard were available (although this is only marginally statistically significant at 90% confidence.) We only examined two metrics here — IG-FREQ and BNS — as these are the two most competitive ones.

4.2.3 Combinations of Features

From the tables above, it is observed that using production rules by itself for sentence grammaticality classification is generally not better than using parse probabilities alone. We therefore attempted to combine the various metrics for parse rules as well as with the parse probabilities. Again, we use only IG-FREQ and BNS.

Table 6 shows that combining various metrics for production rules does not lead to any significant improvement in classification accuracy

Features	IG-FREQ+BNS	IG-FREQ+FREQ	BNS+FREQ
NoisyWSJ	64.76	66.38	63.98

Table 6: SVM results (%) with the combinations of metrics as features on NoisyWSJ — based on top 100 rules from gold standard

Features	NoisyWSJ	NoisierWSJ
IG-FREQ (gold standard) + Parse probabilities	66.59***	77.58***
IG-FREQ (parser output) + Parse probabilities	65.6	75.31***
BNS (gold standard) + Parse probabilities	66.85***	77.66***
BNS (parser output) + Parse probabilities	66.02*	75.6***

Table 7: SVM results (%) with the combinations of parse rules (IG-FREQ and BNS) and parse probabilities as features

Feature (Metrics)	NoisyWSJ	NoisierWSJ
IG-FREQ (gold standard)	67.65	75.65
IG-FREQ (parser output)	60.83	63.41
BNS (gold standard)	63.83	71.28
BNS (parser output)	57.97	62.79

Table 8: Logistic regression results (%) with parse rules as features — based on top 100 rules from both gold standard and parser outputs

(i.e. their combinations still do not perform better than using parse probabilities alone). However, combining parse rules with parse probabilities as shown in Table 7 does demonstrate some modest improvement of 1.6% on average in the overall classification accuracy. With either gold standard or parser-derived production rules, combinations on more noisy data (*NoisierWSJ*) are statistically better than just using parse probabilities alone (all marked with *** are significant at 99% confidence level). This is also true on the less noisy data (*NoisyWSJ*), but only for gold standard production rules.

4.2.4 Effects of Classifiers

As mentioned in Section 3.3, we also examined the effects of using a different classifier — logistic regression. It appears that logistic regression performs on par with SVM as seen in some of the results for logistic regression presented in Table 8.

5 Discussion

Classification accuracy The overall classification accuracies are broadly in line with the published literature (approximately 65% to 80%), although direct comparisons are not possible because of the use of different data sets. Our classification accuracy may have been affected by the choice of parser. Our parser (Stanford) turns out to perform at a somewhat lower level compared to the one used in Foster et al. (2008) (Charniak and Johnson): on the original (grammatical) WSJ, the f-scores are around 85% vs 90%, while there is

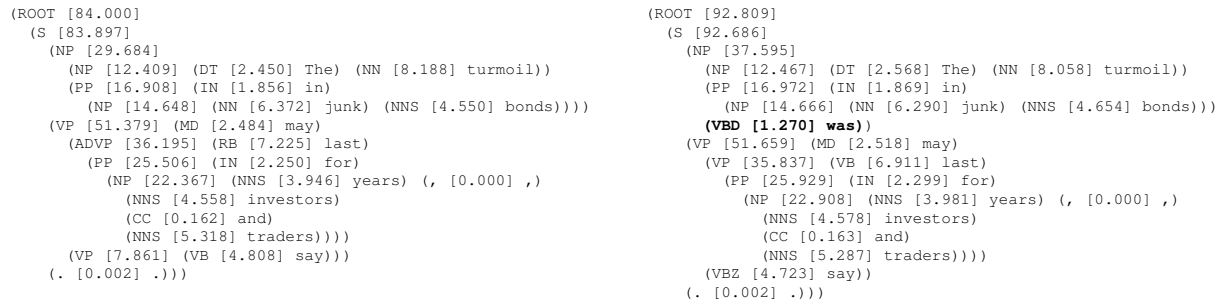


Figure 3: Example of rule selected by IG-FREQ

Feature (Metrics)	Prod Rule	Gram	Ungram
Information Gain (IG-FREQ)	NP → DT DT JJ NN	2	225
	VP → TO TO VP	0	89
	PP → IN IN S	0	73
	PP → NN IN NP	0	70
	NP → NP PP VBD	0	54
Bi-normal Separation (BNS)	PP → IN IN NP	105	1858
	NP → NP IN PP	6	275
	VP → VBZ VBZ NP	0	157
	NP → DT DT NN	2	531
	S → NP VBD VP .	0	242
Ratio	NP → NP , NP , VBD	0	48
	NP → VBP DT JJ CD	0	15
	PP → CD IN NP	0	9
	VP → VB VP PRP	0	9
	S → CC CC NP VP	0	48

Table 9: Examples of parse rules chosen by various metrics (IG-FREQ, BNS, and RATIO)

a slightly bigger difference on the noisy data set, with f-scores of 78–80% vs 85–90%.

Analysis of features We admit to some surprise that looking in detail at production rules did not perform better in general. We examined some of the chosen features under each metric, and these do appear to be strongly characteristic of ungrammatical parses; in particular, there are several instances where probabilities used in IG-PROB appear in our inspection to differ quite noticeably between grammatical and ungrammatical alternatives. We present the top 5 for each of IG-FREQ, BNS and RATIO in Table 9, along with the number of counts in the grammatical versus ungrammatical training corpora. Figure 3 shows an example of one of these rules in a corpus instance.

The problem may be due to feature vector sparsity; looking at other types of cross-sections of parse trees, not only horizontal production rules, (as is done in the parse reranking approach of Charniak and Johnson (2005)), may help with this.

Substitution rules Inspecting the features above, it appears to be the case that substitution cases are hard to detect because the parser is too robust. The way that the Stanford parser handles cases of substitution, even where there is a significant change of part of speech (e.g. *if* for *is*, an example generated in the ungrammatical corpus), results in a parse that is identical to the original grammatical one: the parser is not troubled at all by the ungrammaticality. Supplementing production rules and parser probabilities by n-grams is likely to improve this.

Feature selection metrics It was not entirely surprising that mutual information performed poorly: it tends to select rare instances (Manning and Schutze, 1999) and often does poorly in classification tasks (Forman, 2003). Also as per Forman (2003), IG and BNS performed well. Interestingly, IG perform better in every case with rules alone, while BNS performed better in every combination of rules with parse probabilities, which was overall better than rules alone.

6 Conclusion

The present study has confirmed that parse probabilities are good discriminators for judging the grammaticality of sentences. The idea of exploiting details of the parses in the form of production rules, combined with the parse probabilities, leads to some modest improvement to the overall classification performance.

There are a number of ways in which we might develop further. One would be to use a wider range of features, as in the parser reranking approach noted in Section 5, to avoid sparsity problems. An alternative would be to adopt the noisy channel model: in an alternative to Brockett et al. (2006), ungrammatical trees would be considered noisy versions of their grammatical counterparts. Applying the approach to real ESL data might have different results, with the kinds of errors being less constrained and hence perhaps leading to more significant, and detectable, parse tree changes.

Acknowledgments

The authors would like to acknowledge the support of ARC Linkage Grant LP0776267. Much gratitude is due to Jennifer Foster for the erroneous version of WSJ.

References

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 249–256, Sydney, Australia.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan.

Markus Dickinson. 2010. Generating learner-like morphological errors in Russian. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10)*, pages 259–267, Beijing, China.

George Forman. 2003. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305.

Jennifer Foster and Oistein Andersen. 2009. GenERRate: Generating errors for use in grammatical error detection. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 82–90, Boulder, Colorado.

Jennifer Foster, Joachim Wagner, and Josef van Genabith. 2008. Adapting a WSJ-trained parser to grammatically noisy text. In *Proceedings of ACL-08: HLT, Short Papers*, pages 221–224, Columbus, Ohio.

Jennifer Foster. 2005. *Good Reasons for Noting Bad Grammar: Empirical Investigations into the Parsing of Ungrammatical Written English*. Ph.D. thesis, Department of Computer Science, Trinity College, University of Dublin.

Jennifer Foster. 2007. Treebanks gone bad: Parser evaluation and retraining using a treebank of ungrammatical sentences. *International Journal of Document Analysis and Recognition*, 10(3–4):129–145.

Michael Gamon, Anthony Aue, and Martine Smets. 2005. Sentence-level MT evaluation without reference translations: Beyond language modeling. In *European Association for Machine Translation (EAMT'05)*, pages 103–111, Budapest, Hungary.

Michael Gamon, Jianfeng Gao, Chris Brockett, and Re Klementiev. 2008. Using contextual speller techniques and language modeling for ESL error correction. In *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP'08)*, pages 449–456, Hyderabad, India.

Na-Rae Han, Joel Tetreault, Soo-Hwa Lee, and Jin-Young Ha. 2010. Using an error-annotated learner corpus to develop an ESL/EFL error correction system. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*, pages 763–770, Valletta, Malta.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Christopher Manning and Hinrich Schutze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press.

Andrew Mutton, Mark Dras, Stephen Wan, and Robert Dale. 2007. GLEU: Automatic evaluation of sentence-level fluency. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351, Prague, Czech Republic.

Daisuke Okanohara and Jun'ichi Tsujii. 2007. A discriminative language model with pseudo-negative samples. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 73–80, Prague, Czech Republic.

Guihua Sun, Xiaohua Liu, Gao Cong, Ming Zhou, Zhongyang Xiong, John Lee, and Chin-Yew Lin. 2007. Detecting erroneous sentences using automatically mined sequential patterns. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 81–88, Prague, Czech Republic.

Joachim Wagner, Jennifer Foster, and Josef van Genabith. 2009. Judging grammaticality: Experiments in sentence classification. *CALICO Journal*, 26(3):474–490.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2 edition.

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*, pages 412–420.

A Corpus for Evidence Based Medicine Summarisation

Diego Molla
Macquarie University
Sydney, Australia
diego.molla-aliod@mq.edu.au

Abstract

In this paper we motivate the need for a corpus for the development and testing of summarisation systems for evidence-based medicine. We describe the corpus which we are currently creating, and show its applicability by evaluating several simple query-based summarisation techniques using a small fragment of the corpus.

1 Introduction

Current clinical guidelines urge medical practitioners to practise Evidence Based Medicine (EBM) when providing care for their patients (Sackett et al., 2000). EBM has been defined as "the conscientious, explicit, and judicious use of current best evidence in making decisions about the care of individual patients" (Sackett et al., 1996). To find and appraise the evidence the medical practitioner has access to systematic reviews available through search tools such as the Cochrane Library[1] and UpToDate[2]. However, there is not always a systematic review that addresses the specific topic at hand (Sackett et al., 2000) and then a search on the primary literature becomes necessary.

The amount of documents that exist in the primary literature is overwhelming. The US National Library of Medicine, for example, offers PubMed,[3] a database of medical publications that comprises more than 20 million citations. A search in PubMed often returns thousands of documents. With such amount of text, summarising the information becomes crucial. The tools available to the medical practitioner — see e.g. the Survey by Berkowitz (2002) — typically focus on finding and ranking the relevant papers, often with easy access to the abstracts and type of study, and sometimes with highlight of matching terms. But surprisingly little effort has been placed on summarising the information for easy perusal and appraisal by the user.

In this paper we stress the lack of corpora to help research in evidence-based summarisation of clinical articles (Section 2). We present the characteristics of the corpus we are developing (Section 3), and we show the use of a small fragment of the corpus for the evaluation of simple summarisation techniques (Section 4).

2 Where is the Corpus for Summarisation?

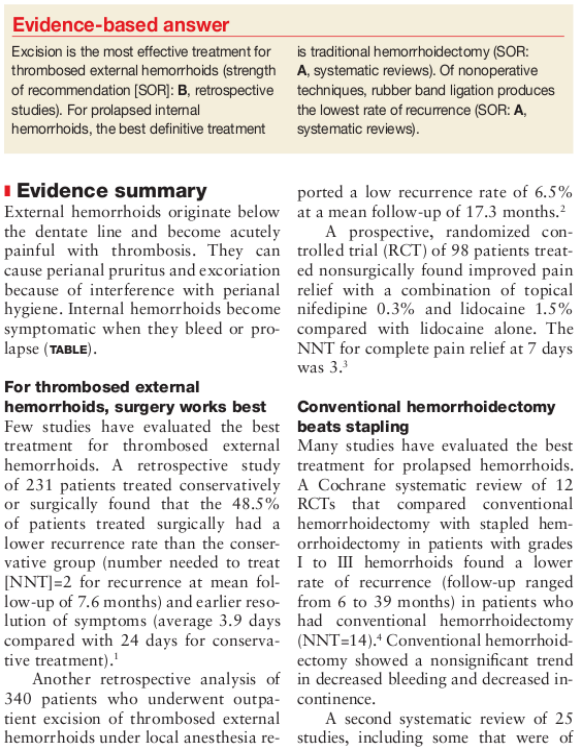
Current summarisation systems have been developed and tested by using corpora built ad-hoc and there is no common corpus readily available specifically for the task. Afantenos et al. (2005) surveys research in summarisation from medical documents. One such summariser is CENTRIFUSER/PERSIVAL (Elhadad et al., 2005), which builds structured query-based representations of the documents as source for the summaries. The system was built using an iterative design that accommodates the feedback of a cohort of users. However, their developers acknowledge the lack of appropriate corpora, and to our knowledge neither CENTRIFUSER nor PERSIVAL were tested on a specific corpus for comparison with other systems.

SemRep (Fiszman et al., 2004) provides abstractive summarisation by producing a semantic representation based on the UMLS concepts and their relations (Bodenreider, 2004) as found in the text. The evaluation was based on human judgement and therefore its results are not readily comparable.

The system by Demner-Fushman and Lin (2006) produces multi-document summaries based on clusters of the main intervention found.

[1] http://www.thecochranelibrary.com/
[2] http://www.uptodateonline.com
[3] http://www.ncbi.nlm.nih.gov/pubmed

Diego Molla. 2010. A Corpus for Evidence Based Medicine Summarisation. In *Proceedings of Australasian Language Technology Association Workshop*, pages 76−80



Evidence-based answer

Excision is the most effective treatment for thrombosed external hemorrhoids (strength of recommendation [SOR]: **B**, retrospective studies). For prolapsed internal hemorrhoids, the best definitive treatment is traditional hemorrhoidectomy (SOR: **A**, systematic reviews). Of nonoperative techniques, rubber band ligation produces the lowest rate of recurrence (SOR: **A**, systematic reviews).

Evidence summary

External hemorrhoids originate below the dentate line and become acutely painful with thrombosis. They can cause perianal pruritus and excoriation because of interference with perianal hygiene. Internal hemorrhoids become symptomatic when they bleed or prolapse (TABLE).

For thrombosed external hemorrhoids, surgery works best

Few studies have evaluated the best treatment for thrombosed external hemorrhoids. A retrospective study of 231 patients treated conservatively or surgically found that the 48.5% of patients treated surgically had a lower recurrence rate than the conservative group (number needed to treat [NNT]=2 for recurrence at mean follow-up of 7.6 months) and earlier resolution of symptoms (average 3.9 days compared with 24 days for conservative treatment).[1]

Another retrospective analysis of 340 patients who underwent outpatient excision of thrombosed external hemorrhoids under local anesthesia reported a low recurrence rate of 6.5% at a mean follow-up of 17.3 months.[2]

A prospective, randomized controlled trial (RCT) of 98 patients treated nonsurgically found improved pain relief with a combination of topical nifedipine 0.3% and lidocaine 1.5% compared with lidocaine alone. The NNT for complete pain relief at 7 days was 3.[3]

Conventional hemorrhoidectomy beats stapling

Many studies have evaluated the best treatment for prolapsed hemorrhoids. A Cochrane systematic review of 12 RCTs that compared conventional hemorrhoidectomy with stapled hemorrhoidectomy in patients with grades I to III hemorrhoids found a lower rate of recurrence (follow-up ranged from 6 to 39 months) in patients who had conventional hemorrhoidectomy (NNT=14).[4] Conventional hemorrhoidectomy showed a nonsignificant trend in decreased bleeding and decreased incontinence.

A second systematic review of 25 studies, including some that were of

Figure 1: Extract of a clinical inquiry from the Journal of Family Practice for the question "Which treatments work best for hemorrhoids?".

The authors present a fine review of possible evaluation methods and they finally settled for a combination of a factoid-based evaluation method, together with the automatic tool for summary evaluation ROUGE (Lin, 2004). The model summaries used for the automatic evaluation were the original paper abstracts. However, by evaluating on a set of abstracts the evaluation was not able to measure the system's ability to perform query-based summarisation, since the abstracts were written prior to any query.

The system by Fiszman et al. (2009) uses factoid-based evaluation that tests the summary ability to find good interventions. This kind of evaluation is not suitable for assessing the summary's ability to indicate the quality of the clinical evidence or other aspects of the summaries that could be important to the medical doctor.

There are collections of clinical questions with their answers that could be used as development and evaluation corpora, such as the Parkhurst Exchange collection,[4] but to our knowledge none of the answers in these collections contain explicit pointers to primary literature. Therefore, as they stand these collections could be used for question-answering tasks but not for query-based summarisation.

3 A Corpus for Summarisation

We are currently developing a corpus of questions and evidence-based information sourced from the Journal of Family Practice (JFP)[5]. We are using all the 496 publicly available documents of the "Clinical Inquiries" section (JFPCI henceforth).[6] Each clinical inquiry from JFPCI contains a clinical question, a short evidence-based answer that includes the strength of recommendation as specified by the Strength of Recommendation Taxonomy (SORT) (Ebell et al., 2004), and a justification of the answer that includes specific references. An extract of a clinical inquiry is shown in Figure 1.

There are two main advantages of using JFPCI rather than direct systematic reviews such as the Cochrane Reviews[7] as a source for our corpus.

1. The format of each inquiry is relatively uniform across all inquiries and therefore it enables a semi-automatic method to convert the data to a corpus that can be used by a machine.

2. The text in each inquiry is much more compact than in a Cochrane review. This results on target text that is closer to what a busy medical practitioner would want to read.

There are other sources of evidence-based text that could be used, such as the project ATTRACT by Public Health Wales (Brassey, 2001).[8] We prefer JFPCI because their procedure to find the answers is more methodical than ATTRACT's and JFPCI includes a short evidence-based answers followed by longer explanations, thus allowing for the use of the corpus for multiple-document and single-document summarisation.

The corpus we are developing is being encoded in XML and each item has the following information (see Figure 2 for a fragment of the encoding of the information from Figure 1):

[4] http://www.parkhurstexchange.com/searchQA

[5] http://jfponline.com/

[6] As of 6 September 2010.

[7] http://www.cochrane.org/cochrane-reviews

[8] http://www.attract.wales.nhs.uk/

<question>**Which treatments work best for hemorrhoids?**</question>
<answer> <snip ID="1">*Excision is the most effective treatment for thrombosed external hemorrhoids.* *<SOR type="B">retrospective studies</SOR>*

<long>A retrospective study of 231 patients treated conservatively or surgically found that the 48.5% of patients treated surgically had a lower recurrence rate than the conservative group (number needed to treat [NNT]=2 for recurrence at mean follow-up of 7.6 months) and earlier resolution of symptoms (average 3.9 days compared with 24 days for conservative treatment). <ref ID="15486746"/ ></long>

<long>Another retrospective analysis of 340 patients ... <ref ID="12972967"/ ></long></snip>

</answer>

Figure 2: Information extracted from a clinical inquiry (formatted to enhance readability)

- A question, which corresponds to the title of the clinical inquiry.
- The answer, which is split into "snips" each one delimited by its evidence level in the original clinical inquiry (e.g. there are 3 answer snips in Figure 2).
- The evidence level of each answer snip (A, B, C) as marked by the source.
- Additional "long" text that justifies the answer by providing a summary of the explicit evidence. This text is manually extracted from the main text body.
- References used in the additional text. Manual lookup in PubMed is being done to locate the PubMed ID.

Not all of the text from the original source is mapped to the XML data (e.g. the sentence "Few studies ..." has been removed in Figure 2), and sometimes minor rephrasing is required to avoid incoherent text.

4 Summarisation Experiments

At the time of writing we had 12 clinical inquiries available for a pilot study. With this fragment we have evaluated several simple query-based single-document summarisation methods. Given a question and an abstract, the summarisers attempt to find those sentences that best satisfy the question information needs. The evaluation system uses ROUGE[9] taking the corresponding <long> element as the model text. For example, in Figure 2, given the abstract with PubMed ID 15486746, the model text is the first <long> element. The 12 clinical inquiries produce a total of 73 text-reference pairs that were used for our evaluation.

We used two baselines:

[9]We used the default settings of ROUGE.

System	*n*	*Avg F*	*Confidence*
Last	3	0.183	[0.159–0.206]
Outcomes	3	0.181	[0.158–0.205]

Table 1: Baseline results

1. (*Last*): Return the last n sentences of the abstract for $n = 1, 3, 7$. We obtained the best values for $n = 3$ with no statistically significant difference between $n = 3$ and $n = 7$.[10]
2. (*Outcomes*): Return the sentences extracted by the US National Library of Medicine (NLM)'s outcome extractor (Demner-Fushman et al., 2006). We chose this system because it reports very good results in the task of finding the outcome information and it is the closest that we have found to the aims of our summarisers. The system returns 3 sentences ($n = 3$).

The results of the evaluation of the baselines are summarised in Table 1.

4.1 Finding the most similar sentences

The two baselines introduced in the previous section return summaries that do not incorporate information from the question. We tested the following summarisers that reward sentences with higher similarity with the question:

1. (*Simple*): Return the n sentences that share any words (except stop words)[11] with the question, for $n = 1, 3, 7$. We found the best results for $n = 3$.

[10]All tests of statistical significance in this paper are based on the 95% confidence intervals returned by ROUGE.

[11]The stop words used are: '[', ']', 'of', 'a', 'the', 'in', 'to', 'and', 'or', 'should', 'than', 'both', 'for', 'with',' through', 'is', 'as', 'that', '.', ',', ';', ':', '(', ')', 'who', 'are', 'this', 'those', 'at', 'has', 'have', 'had', 'been', 'be', 'it', 'were', 'was'.

System	*n*	*Avg F*	*Confidence*
Simple	3	0.180	[0.157–0.203]
UMLS Concepts	3	0.185	[0.161–0.209]
UMLS Graph	3	0.172	[0.149–0.194]

Table 2: Results with query similarity methods

System	*n*	*Avg F*	*Confidence*
No Overlap	3	0.184	[0.161–0.206]
Word	3	0.178	[0.154–0.199]
UMLS	3	0.185	[0.160–0.209]

Table 3: Results with abstract structure

2. (*UMLS Concepts*): Attempt to account for the existence of synonyms by incorporating the information from UMLS. In particular, return the last n sentences that share any UMLS concepts with the question. UMLS concepts are extracted via NLM's MetaMap (Aronson, 2001).

3. (*UMLS Graph*): Incorporate word relations other than synonymy. We do this by incorporating a word similarity measure that is based on random walks through the graph formed by UMLS relations (Agirre et al., 2009). The summarisers of this group return the n sentences that have the greatest similarity score with the question.

We found the best results for $n = 3$ as reported in Table 2. None of the approaches have statistically significant differences on the value of the average F against each other nor against the baselines.

4.2 Using the structure of the abstracts

Many of the source abstracts contain labelled sections. In the next group of summarisers we have attempted to use such structured abstracts to help the summarisers focus on specific sections of the abstracts. We have mapped each abstract section labels into one of "background", "setting", "design", "results", "conclusion", "evidence" and "appendix".[12] Then we have used this information to build summarisers that extract n sentences using this sequence of steps:

1. Extract the first n sentences of the "conclusion" sections.[12]

2. If we have less than n sentences, fill from the first sentences of the last "results" section. If there are still less than n sentences, fill from the first sentences of the second last "results" section, and so on until we have n sentences or we have exhausted all "results" sections.

3. If we still have less than n sentences, fill form the "design" sections using the same method as with the "results" sections described in step 2.

If the abstract did not have structure, the summariser returns the last n sentences as in Section 4.1. We are also studying methods to automatically structure the unstructured abstracts.

We tried a variation that did not use information from the question (*No Overlap*), another one that selected only sentences with word overlap with the question (*Word*), and another one that selected sentences with UMLS overlap with the question (*UMLS*). The results are shown in Table 3. The results are not statistically different among each other or against the results of the previous section.

5 Summary and Conclusions

We have argued for the creation of a corpus for evidence-based medical summarisation. The corpus is currently under construction, and here we have presented a pilot study of the use of a fragment of the corpus to test simple evidence-based summarisers.

We have seen no statistically different results between the approaches presented. We expect to complete the corpus by end 2010. Then we will repeat the experiments and confirm whether there is no real difference in the results. More importantly, we will release the corpus and test its use with more data-intensive approaches including machine learning methods.

The corpus is designed to facilitate the development of multi-document summarisation techniques and this will be one of the of the main research paths that we plan to follow.

Acknowledgments

Parts of this work have been funded by a travel grant by ORISE and by a Macquarie University Research Development Grant (MQRDG).

[12] Note that an abstract may have several sections that result mapped to the same target label.

References

Stergos Afantenos, Vangelis Karkaletsis, and Panagiotis Stamatopoulos. 2005. Summarization from medical documents: a survey. *Artificial Intelligence in Medicine*, 33(2):157–177, February. PMID: 15811783.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27, Morristown, NJ, USA. Association for Computational Linguistics.

A. R. Aronson. 2001. Effective mapping of biomedical text to the umls metathesaurus: the metamap program. *Proc AMIA Symp*, pages 17–21.

Lyle Berkowitz. 2002. Review and evaluation of internet-based clinical reference tools for physicians. Technical report, UpToDate.

Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(Database issue):D267–270. PMID: 14681409.

J. Brassey. 2001. Just in time information for clinicians: a questionnaire evaluation of the ATTRACT project. *BMJ*, 322(7285):529–530.

Dina Demner-Fushman and Jimmy Lin. 2006. Answer extraction, semantic clustering, and extractive summarization for clinical question answering. In *Proceedings ACL*. The Association for Computer Linguistics.

Dina Demner-Fushman, Barbara Few, Susan E Hauser, and George Thoma. 2006. Automatically identifying health outcome information in medline records. *J Am Med Inform Assoc*, 13(1):52–60.

Mark H Ebell, Jay Siwek, Barry D Weiss, Steven H Woolf, Jeffrey Susman, Bernard Ewigman, and Marjorie Bowman. 2004. Strength of recommendation taxonomy (sort): a patient-centered approach to grading evidence in the medical literature. *Am Fam Physician*, 69(3):548–556, Feb.

N Elhadad, M-Y Kan, J L Klavans, and K R McKeown. 2005. Customization in a unified framework for summarizing medical literature. *Artificial Intelligence in Medicine*, 33(2):179–198, February. PMID: 15811784.

Marcelo Fiszman, Thomas C. Rindflesch, and Halil Kilicoglu. 2004. Abstraction summarization for managing the biomedical research literature. In *Procs. HLT-NAACL Workshop on Computational Lexical Semantics*, pages 76–83.

Marcelo Fiszman, Dina Demner-Fushman, Halil Kilicoglu, and Thomas C. Rindflesch. 2009. Automatic summarization of MEDLINE citations for evidence-based medical treatment: A topic-oriented evaluation. *Journal of Biomedical Informatics*, 42:801–813.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, page 10.

David L. Sackett, William M. Rosenberg, Jamuir Gray, R. Brian Haynes, and W. Scott Richardson. 1996. Evidence based medicine: What it is and what it isn't. *BMJ*, 312(7023):71–72.

David L. Sackett, Sharon E. Straus, W. Scott Richardson, William Rosenberg, and R. Brian Haynes. 2000. *Evidence-Based Medicine: How to Practice and Teach EBM*. Churchill Livingstone, 2 edition.

Speaker-Dependent Variation in Content Selection for Referring Expression Generation

Jette Viethen
Centre for Language Technology
Macquarie University
Sydney, Australia
jette.viethen@mq.edu.au

Robert Dale
Centre for Language Technology
Macquarie University
Sydney, Australia
robert.dale@mq.du.au

Abstract

In this paper we describe machine learning experiments that aim to characterise the content selection process for distinguishing descriptions. Our experiments are based on two large corpora of human-produced descriptions of objects in relatively small visual scenes; the referring expressions are annotated with their semantic content. The visual context of reference is widely considered to be a primary determinant of content in referring expression generation, so we explore whether a model can be trained to predict the collection of descriptive attributes that should be used in a given situation. Our experiments demonstrate that speaker-specific preferences play a much more important role than existing approaches to referring expression generation acknowledge.

1 Introduction

Since at least the late 1980s, referring expression generation (REG) has been a key topic of interest in the natural language generation community (see, for example, (Dale, 1989; Dale and Haddock, 1991; Dale and Reiter, 1995; van der Sluis, 2001; Krahmer and Theune, 2002; Krahmer et al., 2003; Jordan and Walker, 2005; van Deemter, 2006; Gatt and van Deemter, 2006; Kelleher and Kruijff, 2006)); and it has recently served as the focus for the first major evaluation efforts in natural language generation (see, for example, (Belz et al., 2009; Gatt et al., 2009)). This level of attention is due in large part to the consensus view that has arisen as to what is involved in referring expression generation: the task is widely accepted as involving a process of selecting those attributes of an intended referent that distinguish it from other potential distractors in a given context, resulting in what is often referred to as a *distinguishing description*.

Most existing REG algorithms rely on handcrafted decision procedures whose behaviour is either entirely deterministic (Dale, 1989; Dale and Haddock, 1991; Gardent, 2002) or can be influenced to some degree using parameters such as preference orderings or cost functions over the available properties in order to choose those that should appear in a referring expression (Dale and Reiter, 1995; van der Sluis, 2001; Krahmer and Theune, 2002; Krahmer et al., 2003; van Deemter, 2006; Gatt and van Deemter, 2006; Kelleher and Kruijff, 2006). However, only very limited attempts have been made to determine how these parameters should best be instantiated in order to allow an algorithm to mimic human-produced referring expressions. Furthermore, the results of recent evaluation exercises (Gupta and Stent, 2005; Viethen and Dale, 2006; Belz and Gatt, 2007; Gatt et al., 2007; Gatt et al., 2008) show that none of these algorithms can be considered an accurate model of human production of referring expressions in any of their instantiations.

In this paper, we take a speaker-oriented perspective on REG that is aimed in part at exploring the factors that impact on the choices that humans make when they refer, and ultimately at finding models for REG which can claim at least a certain level of cognitive plausibility by being able to replicate human referring behaviour. To this end we use two large corpora of referring expressions to train machine learning models on the task of content determination. The larger of these corpora is being introduced for the first time here. We first attempt to build models that are able to predict the content of a referring expression based only on the visual characteristics of the surrounding scene. We then contrast the results of this experiment to those of a second set of experiments in which the machine learner was told which participant had

Jette Viethen and Robert Dale. 2010. Speaker-Dependent Variation in Content Selection for Referring Expression Generation. In *Proceedings of Australasian Language Technology Association Workshop*, pages 81–89

produced each description. Our results show that, while there is too much variation in the data to reliably predict the content of a referring expression based on the visual features of a scene alone, much of this variation can be accounted for by additionally taking into account participant-specific preferences. Even models based on the identity of the participants alone, while not as successful as the models based solely on scene characteristics, performed surprisingly well, underlining the importance of speaker preferences in the choice of semantic content for referring expressions.

In Section 2, we provide an overview of previous work relevant to the approach we take in this paper. Section 3 describes the two corpora that we use for training and testing our models. Section 4 details the experimental setup we used, and in Section 5 we discuss the results of our experiments. Finally, in Section 6, we summarise the key conclusions of this work and point to some future research directions we aim to pursue.

2 Related Work

There exist a number of approaches to the use of machine learning in referring expression generation, although they are typically focussed on aspects of the problem that are distinct from those considered here.

Poesio et al. (1999) addressed the decision of what *type* of NP to use to refer to a given discourse entity in the contexts of museum item descriptions and pharmaceutical information leaflets. They used a statistical model to choose between a large set of NP types, including proper names, definite descriptions, or pronouns. More recently, Stoia et al. (2006) attempted a similar task, but in an interactive navigational domain; as well as deciding what type of referring expression to use, they trained decision trees to determine whether a modifier should be included. Cheng et al. (2001) tried to learn rules for the incorporation of non-referring modifiers into noun phrases. In a domain of spoken negotiations over apartment furniture, Jordan and Walker (2005) used features based on different models of discourse theory to learn rules about which attributes to include in a referring expression. The functions performed by the referring expressions in their corpus went far beyond the simple identification task at hand in our corpora, and they had to take account of a variety of discourse-related factors impacting on their data.

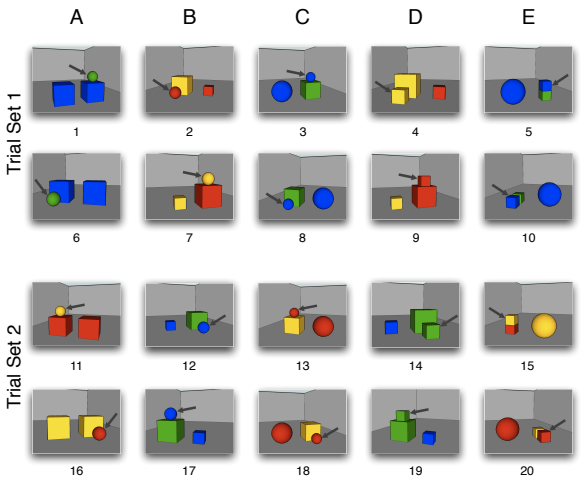


Figure 1: The 20 stimulus scenes for GRE3D3.

A number of the contributions to the 2008 and 2009 GREC and TUNA evaluation tasks have made use of machine learning techniques. The GREC task is primarily concerned with the choice of form of reference (i.e. whether a proper name, a descriptive NP or a pronoun should be used), and so is less relevant to the focus of the present paper. Much of the work on the TUNA task (Gatt et al., 2008) is relevant, however, since this also is concerned with determining the content of referring expressions in terms of the attributes used to build a distinguishing description. In particular, Fabbrizio et al. (2008) explored the impact of individual style and priming on attribute selection for referring expression generation, and Bohnet (2008; 2009) used a nearest-neighbour learning technique to acquire an individual referring expression generation model for each person. Other related approaches to attribute selection in the context of the TUNA task are explored in (Gervás et al., 2008; de Lucena and Paraboni, 2008; Kelleher and Namee, 2008; King, 2008; Hervás and Gervás, 2009; de Lucena and Paraboni, 2009).

3 Two Corpora of Referring Expressions

3.1 Stimulus Design

The experiments in this paper are based on two corpora of human-produced referring expressions. The referring expressions were elicited by showing participants small visual scenes containing a number of simple abstract objects. One of the objects was marked by an arrow to indicate its status as the target referent to be described.

One of the initial intentions underlying both corpus collections was to investigate the condi-

tions under which participants use spatial relations to describe the target referent. Therefore, the design of all scenes is carefully controlled so that the use of relations is encouraged, but not strictly necessary in order to identify the referent. In particular, the target referent is always placed on top of or directly adjacent to a second object, which we call the *landmark* object. Each object is either a ball or a cube, large or small, and of one of two colours. The landmark object is always a cube in order to avoid unnatural looking situations where the target object would be balanced on top of a ball.

The main difference between the stimuli used to collect the two corpora lies in the number of objects contained in the scenes. The first corpus, GRE3D3,[1] has been described in detail elsewhere (Viethen and Dale, 2008); here we only summarise the key points. The stimulus scenes used to collect GRE3D3 are shown in Figure 1; they contain three objects each.

The stimuli used for the second corpus, GRE3D7,[2] contained seven objects. One of these objects was always placed on its own to one side of the scene; the remaining six appeared as three pairs of directly adjacent objects. The target was always a member of the most central pair, and one of the other pairs had the same spatial relation as that holding between the target and the landmark object. The 32 stimuli scenes for GRE3D7 are shown in Figure 2. Their design was balanced for four within-participant factors and one between-participant factor, which were chosen based on the assumption that they might impact on the use of spatial relations. These factors were the size of the landmark object, the commonness of the landmark's size (based on the number of objects sharing the landmark's size), the type of relation holding between the target and the landmark object (vertical or lateral), and two Boolean factors capturing whether the landmark and the target shared size and colour.

3.2 Procedure and Participants

The corpora were collected in two separate self-paced on-line language production experiments. Participants were asked to describe the target referent in each scene in a way that would enable another party looking at the same scene to pick it out

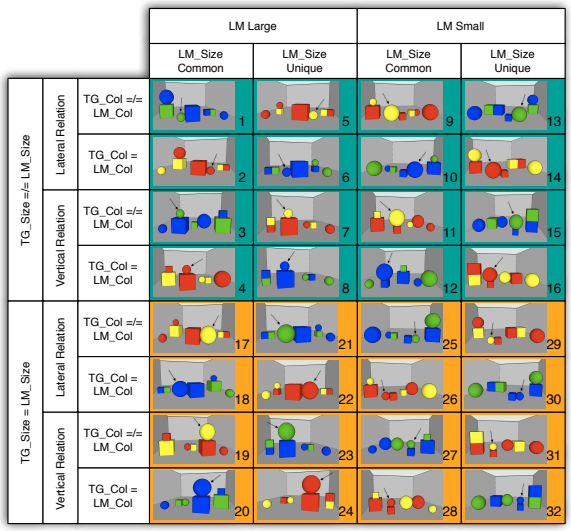


Figure 2: The 32 stimulus scenes for GRE3D7. The top half constitutes Trial Set 1 and the bottom half is Trial Set 2.

from the other objects. The scenes were presented consecutively above a text box into which the participants were required to type a description before clicking 'DONE' to move on to the next scene. In the GRE3D3 collection experiment, the scenes were presented in a preset order directly following each other. For the GRE3D7 experiment, each stimulus scene was preceded by a filler scene. The filler scenes were designed to distract the participants from noticing the similarities between the stimulus scenes. Additionally, the order in which the stimuli and the filler scenes were presented was randomised before each trial.

To encourage the use of fully distinguishing referring expressions, participants were told that they had only one chance at describing the object. After being presented with all the scenes in the trial, participants were asked to complete an exit questionnaire, which asked for their opinion on whether the task became easier over time, and any other comments they might wish to make.

The data from 63 participants in the GRE3D3 collection exercise and from 280 participants in the GRE3D7 collection exercise were used to form the final corpora. A small amount of data from both collections were discarded because the participants did not complete the whole experiment or clearly had not understood the instructions correctly. All participants were self-reported native English speakers.

Both sets of stimuli were subdivided into two trial sets and each participant saw only one of

[1] GRE3D3 stands for **G**enerating **R**eferring **E**xpressions in **3D** scenes with **3** objects.

[2] GRE3D7 stands for **G**enerating **R**eferring **E**xpressions in **3D** scenes with **7** objects.

	Content Pattern	Example Description	% Relative Frequency GRE3D3	GRE3D7
R	⟨tg_size, tg_col, tg_type⟩	the small blue ball	22.70	47.88
D	⟨tg_col, tg_type⟩	the blue ball	27.30	36.70
W	⟨tg_size, tg_col, tg_type, rel, lm_size, lm_col, lm_type⟩	the small blue ball on top of the large green cube	4.76	5.31
F	⟨tg_col, tg_type, rel, lm_col, lm_type⟩	the blue ball on top of the green cube	7.78	2.70
T	⟨tg_size, tg_col, tg_type, rel, lm_col, lm_type⟩	the small blue ball on top of the green cube	4.92	2.08
I	⟨tg_col, tg_type, rel, lm_size, lm_col, lm_type⟩	the blue ball on top of the large green cube	1.90	1.03
ZF	⟨tg_type⟩	the ball	8.25	0.07
Z	⟨tg_size, tg_type⟩	the small ball	4.44	0.38
N	⟨tg_size, tg_col, tg_loc, tg_type⟩	the small blue ball in the left	0.32	0.87
ZK	⟨tg_type, rel, lm_type⟩	the ball on top of the cube	3.49	0.40

Table 1: The ten most common content patterns that occur in both GRE3D3 and GRE3D7.

these trial sets. So, each participant in the GRE3D3 collection provided ten descriptions, while each GRE3D7 participant described 16 stimulus scenes. This resulted in 630 GRE3D3 descriptions (30 for each scene in Trial Set 1, and 33 for each scene in Trial Set 2) and 4480 GRE3D7 descriptions (140 for each stimulus scene).

3.3 Annotation of Semantic Content

In order to be able to analyse the semantic content of the referring expressions, we annotated the attributes and relations contained in each of them. The attributes that participants used in the referring expressions contained in the two corpora, and their possible values, are as follows:

- type [ball, cube]
- colour [blue, green, red, yellow]
- size [large, small]
- location [right, left, front, top]
- relation [on-top-of, in-front-of, left-of, right-of]

In our annotations, each attribute is prefixed by either tg or lm to mark whether it pertains to the target or the landmark object. For example, tg_size indicates that the size of the target was mentioned. This results in nine component properties.[3]

Each description contained in the GRE3D3 and GRE3D7 corpora can be characterised in terms of a *content pattern* defined by the presence or absence of each of these nine component properties. Table 1 lists the ten most common of these content patterns along with example descriptions and the relative frequency with which these patterns occurred in each corpus. 37 different content patterns can be found across the two corpora; the GRE3D3 corpus contains 31 of these 37 content patterns, four more than the much larger GRE3D7 corpus. 21 of the patterns occur in both corpora.

4 Experimental Setup

Most work on referring expression generation attempts to determine what attributes should be used in a description by taking account of aspects of the context of reference. An obvious question is then whether we can learn the content patterns in this data from the contexts in which they were produced. To explore this, we define a number of features that capture the relevant aspects of the visual context in our stimulus scenes. Importantly, these features are general enough to be able to capture both GRE3D3 and GRE3D7 scenes. We use two types of features: *direct property features*, which simply record the attribute value of a certain object in the scene, and *comparative features*, which compare the attribute values of one object to those of the other objects. In a second step, we additionally include Participant_ID as a scene-independent feature. The complete list of 12 features used is shown in Table 2.

The features pay particular attention to the properties of the target and the landmark objects for two reasons: firstly, the nature of the task is such that these two objects can be expected to be closest to the participant's focus of attention; and secondly, these are the only two objects that can be identified as corresponding to each other across all scenes, in particular in the GRE3D7 stimuli.

As direct property features we use the type of spatial relation holding between target and landmark, as people generally show a preference for

[3] As noted by one reviewer, the ethno-cultural background of speakers can have a large impact especially on the use of spatial information. The data would look very different if it had been collected from speakers of languages that mostly make absolute reference to points of the compass rather than using relative information such as 'left' and 'right'.

	Attribute	Explanation	Values
direct property features	TG_Size	size of the target object	small, large
	LM_Size	size of the landmark object	small, large
	Relation_Type	type of relation between target and landmark	horizontal, vertical
comparative features	Num_TG_Size	number of objects of same size as the target	numeric
	Num_LM_Size	number of objects of same size as landmark	numeric
	TG_LM_Same_Size	target and landmark share size	Boolean
	Num_TG_Col	number of objects of same colour as target	numeric
	Num_LM_Col	number of objects of same colour as landmark	numeric
	TG_LM_Same_Col	target and landmark share colour	Boolean
	Num_TG_Type	number of objects of same type as target	numeric
	Num_LM_Type	number of objects of same type as landmark	numeric
	TG_LM_Same_Type	target and landmark share type	Boolean
	Participant_ID	ID number of the description giver	alphanumeric

Table 2: The features and their value formats.

vertical relations over horizontal ones (Lyons, 1977; Gapp, 1995; Bryant et al., 2000; Landau, 2003; Arts, 2004; Tenbrink, 2005), and the sizes of these two objects. We do not include colour or type as features because the actual values of these attributes are unlikely to have an impact on their use. Rather, we expect the proportion of objects sharing these properties, captured in the comparative features, to be of importance. This is different for size, as a large object stands out more from its surroundings than a small one, even independently of the sizes of the other objects. location is not included as it was almost constant across all scenes and can therefore not be used to distinguish between them.

We used the C4.5 decision tree learning algorithm (Quinlan, 1993) implemented in the Weka workbench (Witten and Frank, 2005). We tested both pruned and unpruned trees, but in what follows we comment on the results of the unpruned trees only where they are different from those of the pruned trees. Decision tree pruning is a post-training step that simplifies the trees to reduce over-fitting to the training data. This is especially relevant if the trained models are used on unseen data. However, if the ability of a feature set to characterise a set of natural data is at question, unpruned trees can also be of interest.

5 Results and Discussion

In the following, the fit of the trained models is measured by the Accuracy with which they predict held-out test data or characterise the training data. It is defined as the number of instances predicted correctly divided by the total number of instances in the test or training set.

5.1 Content Selection Based on Scene Characteristics

The Accuracy results achieved by the models trained on the scene-based feature set, without taking into account Participant_ID, are shown in Table 3. As a baseline we report the success rate of a model that simply chooses the majority class in each case. We used three different test methods: (1) testing on the complete training set shows how well the learned model characterises the data and thereby gives an indication of the extent to which the chosen features can explain the variation in the data; (2) ten-fold cross-validation is used to assess the ability of the learned model to generalise to unseen data; and finally, (3) cross-corpus testing gives insights into the difference in variation between the two data sets.

Both models significantly outperform the majority class baseline in all three test methods.[4] No difference can be found between the results for testing on the training sets and cross-corpus testing. However, three interesting observations can be made from these results:

1. Training and testing on the GRE3D7 corpus achieves better results than training and testing on the GRE3D3 corpus.

2. Both the baseline and the decision trees trained on GRE3D3 perform better on GRE3D7 than on GRE3D3 itself, while the GRE3D7-trained models achieve the lowest results when tested on GRE3D3.

3. Overall, none of the decision trees achieve very high Accuracy levels.

[4]We used χ_2 with a maximum $p<.05$ for all significance tests in this paper.

training corpus	test method	maj. class baseline	pruned tree
GRE3D3	training set	27.30%	**46.51%**
	10 fold X	27.30%	**46.51%**
	cross-corpus	36.70%	**47.88%**
GRE3D7	training set	47.88%	**64.93%**
	10 fold X	47.88%	**64.71%**
	cross-corpus	22.70%	**36.98%**

Table 3: Accuracy for the models purely based on scene characteristics. (Bold values are statistically significantly different from the baseline.)

The first two of these points indicate that the content of the referring expressions found in the GRE3D7 corpus is easier to predict than that in the GRE3D3 corpus, a fact that was already foreshadowed by the lower number of different content patterns contained in GRE3D7. The second point in particular shows that the predicted usage patterns for the different content patterns for GRE3D7 are subsumed by the GRE3D3 usage patterns.

One might consider these results to be slightly surprising, as the GRE3D7 corpus with its much larger participant base and size could have been expected to contain more variation than GRE3D3. It is possible that the filler items used in GRE3D7 prevented the participants from noticing how similar their responses to the stimulus scenes were, and thereby also prevented them from intentionally varying the content in their descriptions. A second, related, factor could be that the slightly more complex GRE3D7 scenes forced participants to concentrate on the task more, which also would lead to a reduced number of intentionally-varied descriptions.

From the third observation above we conclude that neither of the learned decision trees are able to accurately model the referring behaviour displayed by the participants in our corpora. In fact, both models predict the use of only two content patterns, patterns R and D, the two most common ones in both data sets, as shown in Table 1. The tree trained on GRE3D3 is shown in Figure 3: it only has three nodes. The GRE3D7-trained tree is at 15 nodes more complex, but nonetheless only predicts the same two most common patterns.

The overall low performance of the models might either be due to some of the variation in the data being in fact unpredictable (due to factors that we did not capture in the collection experiments) or random, or it may indicate that the features we made available to the machine learning algorithm were not sufficient to model the variation.

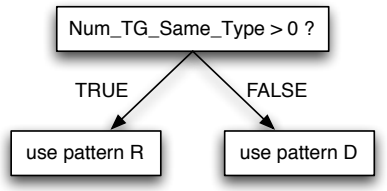


Figure 3: The participant-insensitive decision tree trained on GRE3D3.

5.2 Participant-Dependent Modelling

Based on observations we made in (Viethen and Dale, 2006) for a different data set, we hypothesise that one main factor that might be at play in producing the variation in the two corpora used here are the differing preferences of the individual participants. We therefore introduced the feature Participant_ID and carried out two further experiments: first, we tested the predictive ability of this feature on its own by removing all other features from the set provided to the machine learner; and second, we combined the scene-based features with the Participant_ID feature, in order to assess the extent to which the individual participants were taking the features of the scene into account when referring to the target referents.

Table 4 compares the results of the second two experiments to those of the participant-insensitive decisions trees from the previous section.[5]

We firstly observe that the size of the learned decision trees, measured in terms of the number of nodes they contain, increases dramatically when Participant_ID is taken into account, even when the other, scene-based, features are also available. This indicates that, when given the option to use this feature, the machine learner chooses to do so in every case, demonstrating the usefulness of Participant_ID in characterising our data.

The trees based on Participant_ID alone also achieved surprisingly good performance, although these trees are forced to choose one content pattern for all descriptions produced by a given participant. Only the Accuracy of the tree trained on GRE3D3 was significantly lower than that of the corresponding participant-insensitive tree; the other scores are surprisingly close to those based

[5]Because the participants in the two data collection exercises were not the same, cross-corpus testing of the participant-sensitive models is not possible.

training corpus	test method	+[scene features] −Participant_ID pruned Acc	pruned nodes	−[scene features] +Participant_ID n/a Acc	n/a nodes	+[scene features] +Participant_ID pruned Acc	pruned nodes	+Participant_ID unpruned Acc	unpruned nodes
GRE3D3	training set	46.51%	3	41.91%	64	**91.27%**	415	**98.10%**	573
	10 fold X	46.51%		**31.11%**		54.44%		**57.61%**	
GRE3D7	training set	64.93%	15	62.28%	281	**82.59%**	1023	**93.77%**	2798
	10 fold X	64.71%		57.12%		67.01%		63.71%	

Table 4: Accuracy and tree size for the models based on scene and participant information. (Bold values are statistically significantly different to the participant-insensitive trees.)

on scene features only.[6]

Combining the scene-based features with Participant_ID gives better results than either of the two exclusive models achieve. To the best of our knowledge, their cross-validation scores are also higher than any Accuracy scores reported in the literature for any existing algorithm instantiated with a set parameter setting.[7] However, in 10-fold cross-validation, only the unpruned GRE3D3 model achieves a statistically significant improvement over the participant-insensitive model. When testing on the training set, the pruned and unpruned trees for both corpora vastly outperform the models that do not take participant preferences into account. In particular, the Accuracy scores achieved by the unpruned models are very high.

These results confirm the hypothesis that speaker preferences play a very important role in shaping the semantic content of referring expressions in identification tasks. Trees using Participant_ID as the only feature perform surprisingly well, and the trees that take account of both the features of the scene and the preferences displayed by individual speakers are able to characterise our two data sets with very high accuracy. Our particular choice of scene-based features is also supported by these results, as they do seem to capture the factors that individual speakers rely on when they build referring expressions.

The fact that they only achieve high scores if tested directly on the training set shows that these models are very specific to the data they were trained on, and would not necessarily generalise well to unseen data. A likely explanation for the large differences between the cross-validation results and results on the training set is the low number of instances per participant in both corpora. We have ten descriptions from each participant in the GRE3D3 corpus and 16 in GRE3D7, and neither of the corpora contains multiple descriptions from the same participant for a given stimulus.

[6]Note that pruning has no effect on trees using only one feature, in this case Participant_ID.

[7]This comparison must be viewed with caution, as the other evaluations were carried out on different test corpora.

6 Conclusions and Future Work

This paper is based on the view that a primary consideration in the study of REG should be the development of systems that are able to explain and replicate the semantic content found in human data. We hold this view for two reasons: firstly, such systems can aid the exploration of factors that impact on the semantic choices that people make when they refer and ultimately might be able to claim some level of psychological reality; and secondly, generating the same referring expressions as humans can also serve a utilitarian purpose, as only human-like reference is likely to be accepted as fully natural by listeners.

We have chosen a straightforward approach to building REG models that take into account what people do by training decision trees on two human-produced corpora of distinguishing descriptions in visual scenes. We defined a set of features to capture the relevant visual aspects of the stimuli used in the data collection exercises for the two corpora. In our first experiment we established that decision trees trained using these features are able to outperform a majority class baseline, but are not able to replicate a large enough proportion of the data to be considered accurate models of human reference behaviour. In a second experiment we added the Participant_ID feature, which allowed the machine learner to establish participant-specific behaviour patterns. Trees based on this feature alone achieved surprisingly good results, and the participant-sensitive trees which also took into account the features of the scene achieved much higher Accuracy scores than

the participant-insensitive trees.

The main conclusion we draw from these experiments is that speaker-dependent variation is one of the most important factors shaping content selection processes in the referring behaviour of humans. This is an observation that has been overlooked in the development of most existing algorithms for REG. However, if our aim is to build algorithms that are able to accurately model corpora of human referring expressions, as was the case in the recent public evaluation campaigns in REG (Belz and Gatt, 2007; Gatt et al., 2008; Gatt et al., 2009), then we cannot ignore this fact.

Our next step is to take this work further by training individual models for each speaker. Such speaker-specific trees will allow us to explore the different strategies that people follow when they refer, and to compare the strategies of different speakers to each other. We think it unlikely that every individual speaker is idiosyncratic in this regard; our hypothesis is that it will be possible to use automatic clustering techniques to identify groups of people who follow the same strategies. Such clusters can then be used to make predictions that are sensitive to between-participant differences while benefitting from the commonalities in people's behaviour. It might also be interesting to see if non-linguistic characteristics of speakers, such as age, gender, and social or cultural background, can account for some of the between-participant variation in reference behaviour.

In a second strand of work we are exploring an alternative approach to learning human reference behaviour from this data. We are training attribute-specific trees that make binary decisions about the use of each individual attribute in a given reference situation, instead of predicting whole content patterns. The attribute-specific trees for a given participant can then be combined into a *speaker profile* predicting complete referring expressions produced by this speaker.

References

Anja Arts. 2004. *Overspecification in Intructive Texts*. Ph.D. thesis, University of Tilburg, The Netherlands.

Anja Belz and Albert Gatt. 2007. The attribute selection for GRE challenge: Overview and evaluation results. In *Proceedings of UCNLG+MT: Language Generation and Machine Translation*, 75–83, Copenhagen, Denmark.

Anja Belz, Eric Kow, Jette Viethen, and Albert Gatt. 2009. The GREC Main Subject Reference Generation Challenge 2009: Overview and evaluation results. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation (UCNLG+Sum 2009)*, 79–87, Singapore.

Bernd Bohnet. 2008. The fingerprint of human referring expressions and their surface realization with graph transducers. In *Proceedings of the 5th International Conference on Natural Language Generation*, 207–210, Salt Fork OH, USA.

Bernd Bohnet. 2009. Generation of referring expression with an individual imprint. In *Proceedings of the 12th European Workshop on Natural Language Generation*, 185–186, Athens, Greece.

David J. Bryant, Barbara Tversky, and M. Lanca. 2000. Retrieving spatial relations from observation and memory. In E. van der Zee and U. Nikanne (Eds.), *Cognitive interfaces: Constraints on linking cognitive information*, 94–115. Oxford University Press, Oxford, UK.

Hua Cheng, Massimo Poesio, Renate Henschel, and Chris Mellish. 2001. Corpus-based NP modifier generation. In *Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh PA, USA.

Robert Dale and Nicolas Haddock. 1991. Content determination in the generation of referring expressions. *Computational Intelligence*, 7(4):252–265.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.

Robert Dale. 1989. Cooking up referring expressions. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, 68–75, Vancouver BC, Canada.

Diego Jesus de Lucena and Ivandré Paraboni. 2008. USP-EACH: Frequency-based greedy attribute selection for referring expressions generation. In *Proceedings of the 5th International Natural Language Generation Conference*, 219–220, Salt Fork OH, USA.

Diego Jesus de Lucena and Ivandré Paraboni. 2009. USP-EACH: Improved frequency-based greedy attribute selection. In *Proceedings of the 12th European Workshop on Natural Language Generation*, 189–190, Athens, Greece.

Giuseppe Di Fabbrizio, Amanda J. Stent, and Srinivas Bangalore. 2008. Referring expression generation using speaker-based attribute selection and trainable realization (ATTR). In *Proceedings of the 5th International Natural Language Generation Conference*, 211–214, Salt Fork OH, USA.

Klaus-Peter Gapp. 1995. Angle, distance, shape, and their relationship to projective relations. In *Proceedings of the 17th Conference of the Cognitive Science Society*, 112–117, Pittsburgh PA, USA.

Claire Gardent. 2002. Generating minimal definite descriptions. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 96–103, Philadelphia PA, USA.

Albert Gatt and Kees van Deemter. 2006. Conceptual coherence in the generation of referring expressions. In *Proceedings of the 21st COLING and the 44th ACL Conference*, 255–262, Sydney, Australia.

Albert Gatt, Ielka van der Sluis, and Kees van Deemter. 2007. Evaluating algorithms for the generation of referring expressions using a balanced corpus. In *Proceedings of the 11th European Workshop on Natural Language Generation*, 49–56, Schloß Dagstuhl, Germany.

Albert Gatt, Anja Belz, and Eric Kow. 2008. The TUNA challenge 2008: Overview and evaluation results. In *Proceedings of the 5th International Natural Language Generation Conference*, 198–206, Salt Fork OH, USA.

Albert Gatt, Anja Belz, and Eric Kow. 2009. The TUNA-REG Challenge 2009: Overview and evaluation results. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, 174–182, Athens, Greece.

Pablo Gervás, Raquel Hervás, and Carlos León. 2008. NIL-UCM: Most-frequent-value-first attribute selection and best-scoring-choice realization. In *Proceedings of the 5th International Natural Language Generation Conference*, 215–218, Salt Fork OH, USA.

Surabhi Gupta and Amanda Stent. 2005. Automatic evaluation of referring expression generation using corpora. In *Proceedings of the Workshop on Using Corpora for Natural Language Generation*, 1–6, Brighton, UK.

Raquel Hervás and Pablo Gervás. 2009. Evolutionary and case-based approaches to REG: NIL-UCM-EvoTAP, NIL-UCM-ValuesCBR and NIL-UCM-EvoCBR. In *Proceedings of the 12th European Workshop on Natural Language Generation*, 187–188, Athens, Greece.

Pamela W. Jordan and Marilyn A. Walker. 2005. Learning content selection rules for generating object descriptions in dialogue. *Journal of Artificial Intelligence Research*, 24:157–194.

John Kelleher and Geert-Jan M. Kruijff. 2006. Incremental generation of spatial referring expressions in situated dialog. In *Proceedings of the 21st COLING and the 44th ACL Conference*, 1041–1048, Sydney, Australia.

John D. Kelleher and Brian Mac Namee. 2008. Referring expression generation challenge 2008: DIT system descriptions. In *Proceedings of the 5th International Natural Language Generation Conference*, 221–224, Salt Fork OH, USA.

Josh King. 2008. OSU-GP: Attribute selection using genetic programming. In *Proceedings of the th International Natural Language Generation Conference*, 225–226, Salt Fork OH, USA.

Emiel Krahmer and Mariët Theune. 2002. Efficient context-sensitive generation of referring expressions. In Kees van Deemter and Rodger Kibble (Eds.), *Information Sharing: Reference and Presupposition in Language Generation and Interpretation*, 223–264. CSLI Publications, Stanford CA, USA.

Emiel Krahmer, Sebastiaan van Erk, and André Verleg. 2003. Graph-based generation of referring expressions. *Computational Lingustics*, 29(1):53–72.

Barbara Landau. 2003. Axes and direction in spatial language and spatial cognition. In Emilie van der Zee and Jon M. Slack (Eds.), *Representing Direction in Language and Space*, 18–38. Oxford University Press, Oxford, UK.

John Lyons. 1977. *Semantics*, volume 2. Cambridge University Press.

Massimo Poesio, Renate Henschel, Janet Hitzeman, and Rodger Kibble. 1999. Statistical NP generation: A first report. In *Proceedings of the ESSLLI Workshop on NP Generation*, Utrecht, The Netherlands.

J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco CA, USA.

Laura Stoia, Darla Magdalene Shockley, Donna K. Byron, and Eric Fosler-Lussier. 2006. Noun phrase generation for situated dialogs. In *Proceedings of the 4th International Conference on Natural Language Generation*, 81–88, Sydney, Australia.

Thora Tenbrink. 2005. Semantics and application of spatial dimensional terms in English and German. Technical Report Series of the Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition, No. 004-03/2005, Universities of Bremen and Freiburg, Germany.

Kees van Deemter. 2006. Generating referring expressions that involve gradable properties. *Computational Linguistics*, 32(2):195–222.

Ielka van der Sluis. 2001. An empirically motivated algorithm for the generation of multimodal referring expressions. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, Student Session*, 67–72, Toulouse, France.

Jette Viethen and Robert Dale. 2006. Algorithms for generating referring expressions: Do they do what people do? In *Proceedings of the 4th International Conference on Natural Language Generation*, 63–70, Sydney, Australia.

Jette Viethen and Robert Dale. 2008. Generating referring expressions: What makes a difference? In *Australasian Language Technology Association Workshop 2008*, 160—168, Hobart, Australia, Dec.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco CA, USA.

Reranking a wide-coverage CCG parser

Dominick Ng and **Matthew Honnibal** and **James R. Curran**
School of Information Technologies
University of Sydney
NSW 2006, Australia
{dong7223,mhonn,james}@it.usyd.edu.au

Abstract

n-best parse reranking is an important technique for improving the accuracy of statistical parsers. Reranking is not constrained by the dynamic programming required for tractable parsing, so arbitrary features of each parse may be considered.

We adapt the reranking features and methodology used by Charniak and Johnson (2005) for the C&C Combinatory Categorial Grammar parser, and develop new features based on the richer formalism. The reranker achieves a labeled dependency F-score of 87.59%, which is a significant improvement over prior results.

1 Introduction

Accurate syntactic parsing has proven to be critical for many tasks in natural language processing (NLP), including semantic role labeling (Gildea and Jurafsky, 2002), question answering (Echihabi and Marcu, 2003), and machine translation (DeNeefe and Knight, 2009). Improved parser accuracy benefits many downstream tasks in the field.

One method of improving parsing accuracy is *reranking* – the process of reordering the top n analyses as determined by a base parser (Collins, 2000). The statistical models used in phrase-structure and dependency parsers rely on dynamic programming algorithms that restrict possible features to a local context. This is necessary for efficient decoding of the potential parse forest, ensuring tractability at the cost of excluding any non-local features from consideration. Reranking operates over complete trees that are the most probable derivations under the dynamic programming model, allowing arbitrary complex features of the parse to be incorporated without sacrificing efficiency. Poor local decisions made by parsers are easier to model and capture in the reranking phase.

Collins (2000) reports a 1.55% accuracy improvement with reranking for the Collins parser, and Charniak and Johnson (2005) reports a 1.3% improvement for a reranked Charniak parser. An open question is how well reranking applies to parsers of different design to the Charniak and Collins parsers. An attempt to port the Charniak and Johnson reranker for the Berkeley parser (Petrov et al., 2006) produced only minimal accuracy improvements (Johnson and Ural, 2010), suggesting that careful feature engineering is necessary for good performance.

In this paper we describe the implementation of a discriminative maximum entropy reranker for the C&C parser (Clark and Curran, 2007), a state-of-the-art system based on Combinatory Categorial Grammar (CCG). We reimplement the features described in Charniak and Johnson (2005) to suit the CCG parser and replicate the Charniak reranker setup. Our experiments show that the PCFG-style features are less effective at reranking CCG than Penn Treebank-style trees. We hypothesise that the binary-branching structure of CCG is the cause, as CCG trees are deeper and create different constituent structures compared to Penn Treebank trees. To address this, we develop a number of new features to take advantage of the more detailed formalism and the evaluation over recovered dependencies. We also experiment with regression and classification approaches, variations in feature pruning, and differing numbers of n-best parses for the reranker to consider.

The reranker achieves a best labeled dependency F-score of 87.13% on Section 00 of CCGbank and 87.59% on Section 23. The performance gains are statistically significant, but small in real terms, indicating that crafting reranking features is not a trivial process. However, the continued improvements in parsing accuracy will benefit downstream applications utilising the parser through more accurate syntactic analysis.

Dominick Ng, Matthew Honnibal and James R. Curran. 2010. Reranking a wide-coverage CCG parser. In *Proceedings of Australasian Language Technology Association Workshop*, pages 90−98

2 Parser Reranking

Reranking has been successfully applied to dependency parsing (Sangati et al., 2009), machine translation (Shen et al., 2004), and natural language generation with CCG (White and Rajkumar, 2009). Collins (2000) describes reranking for the Collins (Model 2) parser (Collins, 1999). 36,000 sentences from Sections 02-21 of the Penn Treebank WSJ data are parsed with a modified version of the base parser, producing an average of 27 parses per sentence. Features are extracted from the parses to create reranker training data, including lexical heads and the distances between them, context-free rules in the tree, n-grams and their ancestors, and parent-grandparent relationships. Collins reports a final PARSEVAL F-score of 89.75% using a boosting-based reranker, a 1.55% improvement compared to the baseline parser.

The potential benefits from reranking are dependent on the quality of the candidate n-best parses. Huang and Chiang (2005) describe efficient and accurate algorithms for this task based on a directed hypergraph analysis framework (Klein and Manning, 2001). By improving the quality of the candidate parses, Huang and Chiang demonstrate how oracle reranking scores (using a perfect reranker that always choses the best parse from an n-best list) can be dramatically improved compared to the parses used in Collins (2000).

Charniak and Johnson (2005) describe discriminative reranking for the Charniak parser. A coarse-to-fine parsing approach allows high-quality n-best parses to be tractably computed while retaining dynamic programming in the parser. When run in 50-best mode the Charniak n-best parser has an oracle F-score of 96.8% in the standard PARSEVAL metric – much higher than the 89.7% parser baseline. The reranker produces a final F-score of 91.0% in 50-best mode. This is an improvement of 1.3% over the baseline model. Self-training over the reranked parses further improves performance to 92.1% F-score, which remains the state-of-the-art (McClosky et al., 2006). Self-training provides the additional benefit of improving the Charniak parser's performance on out-of-domain data – a known weakness of supervised parsing.

More recently, the Charniak reranking system has been adapted for the Berkeley parser (Petrov et al., 2006). Unlike the Collins and Charniak parsers, which are broadly similar and heavily based on lexicalised models, the Berkeley parser uses a split-merge technique to acquire a much smaller, unlexicalised grammar from its training data. Johnson and Ural (2010) report that reranking leads to negligible performance improvements for the Berkeley parser, and acknowledge that the reranker's feature set, adapted from Charniak and Johnson (2005), may be implicitly tailored to the Charniak parser over the Berkeley parser. In particular, the feature pruning process for reranking was conducted over output from the Charniak parser, which may have prevented useful features for the Berkeley parser from being chosen.

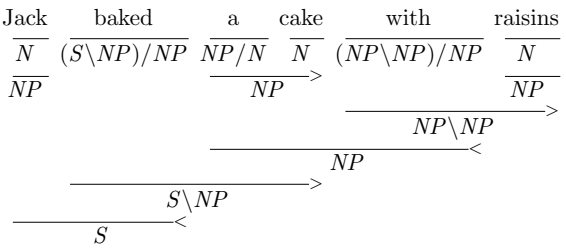


Figure 1: A simple CCG derivation.

3 Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG, Steedman (2000)) is a lexicalised grammar formalism based on combinatory logic. The grammar is directly encoded in the lexicon in the form of categories that govern the syntactic behaviour of each word. A small number of generic rules combine categories together to form a spanning analysis.

Categories may be atomic or complex. Atomic categories represent words and constituents that are syntactically complete, such as nouns (N), noun phrases (NP), prepositional phrases (PP), and sentences (S). Complex categories are binary structures of the form X/Y or $X\backslash Y$, and represent structures which combine with an argument of category Y to produce a result of category X. The forward and backward slashes indicate that Y is expected to the right and left respectively.

Complex categories can be thought of as functors that require particular arguments to produce a grammatical construction. Subcategorization information is encoded using nested categories. For example, transitive verbs have the category $(S\backslash NP)/NP$, which indicates that one object NP is expected to the right to form a verb phrase $S\backslash NP$, which in turn expects one subject NP to the left to form a sentence.

In addition to forward and backward application, CCG has a number of other binary combinators based on function composition. There are also unary type-changing combinators that take a single category and transform it into another category. Figure 1 gives a simple CCG derivation, showing how categories are successively combined together to yield an analysis.

4 The C&C parser

The C&C parser (Clark and Curran, 2007) is a fast, highly accurate parser based on the CCG formalism. The parser is used in question answering systems (Bos et al., 2007), computational semantics tools (Bos et al., 2004), and has been shown to perform well in recovering unbounded dependencies (Rimell et al., 2009).

The parser divides the parsing process into two main phases: *supertagging* and *parsing*. First, the supertagger assigns a small set of initial categories to each word in the sentence. Then, the parser attempts to find a spanning analysis using the proposed categories using the modified CKY algorithm described in Steedman (2000). If the parser cannot find an analysis (i.e. there is no sequence of combinators that can combine the proposed categories) the supertagger is run again at a higher ambiguity level, giving each word a larger set of possible categories, and the process is repeated. The supertagging phase dramatically reduces the number of derivations for the parser to consider, making the system highly efficient.

An n-best version of the C&C parser has recently been developed (Brennan, 2008), incorporating the algorithms described in Huang and Chiang (2005). The n-best parser is almost as efficient as the baseline 1-best version, and we use it as the basis for all experiments presented in this paper.

CCGbank is the standard corpus for English parsing with CCG. It is a transformation of the Penn Treebank WSJ data into CCG derivations and dependencies (Hockenmaier and Steedman, 2007). Sections 02-21 are the standard training data for the C&C parser, with Section 00 used for development and Section 23 for evaluation. The supertagger requires part-of-speech information for each word as part of its feature set, so a POS tagger is also included with the C&C parser. Both the supertagger and the POS tagger are trained over tags extracted from Sections 02-21 of CCGbank.

5 Methodology

We frame the reranking task for CCG parsing as follows: given an n-best list of parses, ranked by the parser, choose the parse that is as close as possible to the gold standard. We use the standard CCG labeled dependency metric as described in Hockenmaier (2003) to define closeness to the gold standard, allowing us to explore both classification and regression as frameworks for the task. In classification, the closest sentence(s) to the gold standard with respect to F-score are labeled as positive, while all other sentences are labeled as negative. If there are multiple parses with the highest F-score, they are all labeled as positive. In regression, the F-score of each parse is used as the target value. Both classification and regression approaches were implemented using MEGAM[1].

n-best lists of parses were generated using the n-best C&C parser using Algorithm 3 of Huang and Chiang (2005). We used the normal-form model for the C&C parser as described in Clark and Curran (2007) for all experiments in this paper. Reranker training data was created using n-best parses of each sentence in Sections 02-21 of CCGbank. As this is also the parser's training data, care must be taken to avoid generating training data where the parser's confidence level is different to that at run-time (caused by parsing the training data). We constructed ten folds of Sections 02-21, training the POS tagger, supertagger, and parser on nine of the folds and producing n-best parses over the remaining fold.

Features were generated over the n-best parses of the folded training data and the appropriate label assigned based on the F-score. This data was used to train the reranker. Similarly, Section 24 of CCGbank was parsed using a model trained over Sections 02-21 for use as tuning data. At run-time, features were generated over the n-best parses of the test data, and the most probable parse (classification) or the parse with the highest predicted F-score (regression) was returned.

We experimented with values of 10 and 50 for n to balance between the potential accuracy improvement and the efficiency of the reranker. n was kept constant between the training data and the final test data (i.e. a reranker trained on 50-best parses was then tested over 50-best parses).

Following Charniak and Johnson (2005) we implemented feature pruning for the reranker train-

[1] http://www.umiacs.umd.edu/~hal/megam

ing data as follows. For each sentence, define a feature as being *pseudo-constant* if it does not differ in value over all the parses for that sentence. We keep all features that are non pseudo-constant in at least t sentences in the training data. We experimented with values of 0, 2, and 5 for t to investigate the effect of feature pruning.

6 Reranking Features

The features described in this section are calculated over CCG derivation trees produced by the C&C parser. We began by implementing the features described by Charniak and Johnson (2005), before developing features specifically for CCG derivations. We also implemented the CCG parsing features described by Clark and Curran (2007), so that our reranking model would have access to the information used by the parser. These features include various combinations of word-category, word-POS, CCG rule, distance, and dependency information. Finally, the log score and rank assigned to each derivation by the parser were encoded as core features for the reranker.

CCG derivation trees have some important structural differences from the trees that the Charniak and Johnson features were designed for. The most important difference is that CCG trees are at most binary branching[2]. As the longest non-terminal in the Penn Treebank has 51 children, features designed to generalise long production rules are useful in the Charniak and Johnson reranker but are less relevant to CCG trees.

Another important difference is that CCG production rules are constrained by the combinatory rules, whereas Penn Treebank productions combine unrelated atomic symbols. For instance, a Penn Treebank production NP→ NP PP would be translated into CCG as $NP\rightarrow\ NP\ NP\backslash NP$. Much of the information in the production is already present in the structure of the $NP\backslash NP$ category. We speculate that this will make the features that capture the vertical context of a production rule less useful for CCG.

Finally, each ccg tree corresponds to exactly one dependency analysis, and this is produced as output by the C&C parser. This gives the reranker access to the full dependency analysis of each sentence, making the dependency-approximation heuristics used by Charniak and Johnson (2005) unnecessary for our purposes.

The features adapted from Charniak and Johnson (2005) are described in Sections 6.1 and 6.2 below. The novel CCG features we develop are described in Section 6.3. Most features were implemented as simple boolean indicator functions. Maximum entropy modelling exponentiates feature values, so real-valued features are more influential than boolean features. We mitigated this effect by taking the log of real-valued features.

6.1 Tree Topology Features

These features attempt to describe the overall shape of the parse tree, to capture the fact that English generally favours right-branching parse trees, with phonologically heavy constituents generally occurring in sentence-final position. Tree topology can also be useful in capturing the balance found in coordination attachment. These guidelines distinguish the correct parse tree in Figure 2a from the incorrect parse tree in Figure 2b – the incorrect tree is more left-branching than the correct tree, with a shallower depth of balance in the coordination.

CoPar: records coordination parallelism at various depths. Indicates whether both sides of a coordination are identical in structure and category labels at depths of 1 to 4 from the coordinator.

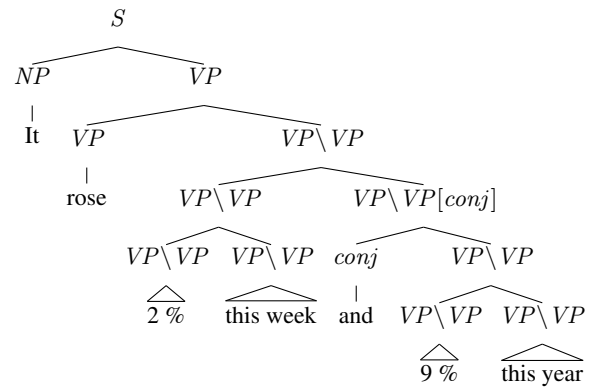
CoLenPar: indicates the difference in size between two halves of a conjunction (where size is the number of nodes in the yield) as well as whether the latter half is the final element.

Heavy: encodes the category and size of the subtree rooted at each non-terminal, whether the non-terminal is at the end of the sentence, and whether it is followed by punctuation. This crudely captures the tendency for larger constituents to lie further to the right in a tree.

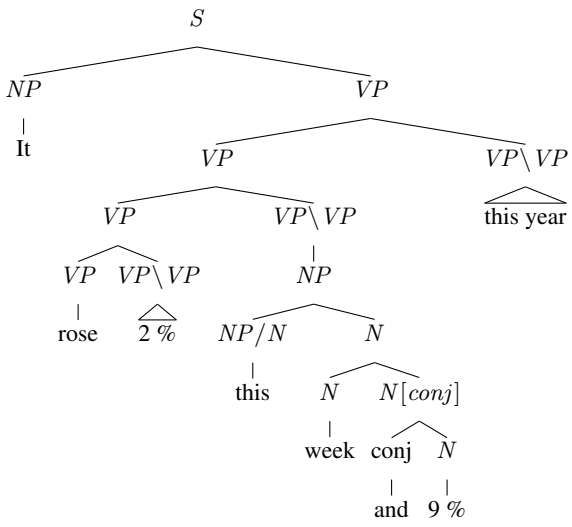
RightBranch: encodes the number of non-terminals on the longest path from the root of the tree to the right-most non-punctuation node in the tree, and the number of non-terminals in the tree that are not on this path.

SubjVerbAgr: captures the conjoined POS tags of the subject noun and verb in a sentence to distinguish cases where the pluralisation does not agree. The subject is assumed to be the final NP before the verb phrase ($S\backslash NP$) in a sentence.

[2] Steedman (2000) describes a ternary conjunction rule, but this is broken into two binary productions in CCGbank, using the marker $[conj]$.



(a) The correct parse.



(b) Parse featuring a conjunction error.

Figure 2: Two CCG derivations for the sentence, *It rose 2% this week and 9% this year.*

6.2 Local Context Features

These features, adapted from Charniak and Johnson (2005), attempt to represent various fragments of the tree as well as incorporate layers of vertical and horizontal context that are difficult to encode in the parser model.

Edge: captures the words and POS tags immediately preceding and following the subtree rooted at each non-terminal in the tree. This crudely captures poor attachment decisions in local trees.

Heads: represents pairs of constituent heads as indicated by the parser at various levels in the tree. Heads are encoded as lexical items and POS tags.

HeadTree: records the entire tree fragment (in a bracketed string format) projected upwards from the head word of the sentence.

Neighbours: encodes the category of each non-terminal, its binned size, and the POS tags of the ℓ_1 preceding words and the ℓ_2 following words, where $\ell_1 = 1$ or 2 and $\ell_2 = 1$. Binned size is the number of words in the yield of the non-terminal, bucketed into 0, 1, 2, 4, or 5+.

NGramTree: records tree fragments rooted at the lowest common ancestor node of $\ell = 2$ or 3 contiguous terminals in the tree. This represents the subtree encompassing each sequence of ℓ words in the sentence.

Rule: captures the equivalent CCG rule application represented at each non-terminal node; equivalent to a context-free production rule.

SynSemHeads: yield pairs of semantic heads (e.g. the rightmost noun in a noun phrase) and functional heads (e.g. the determiner in a noun phrase) at each non-terminal in the tree. Heads are encoded as lexical items and POS tags.

Word: yields each word in a sentence along with the categories of $\ell = 2$ or 3 of its immediate ancestor nodes in the tree.

WProj: for each terminal in the tree, encode the word combined with the category of its maximal projection parent, which is the first node found by climbing the tree until the child node is no longer the head of its parent.

6.3 CCG Features

We devised a number of new features for CCG aimed at uncovering various combinator sequences or combinations that may indicate an overly complicated or undesirable derivation. Additionally, these features attempt to encode more information about the dependencies licensed by the derivation as it is these dependencies which will be evaluated.

Balance: encodes the overall balance of the tree in terms of the ratio of leaves and the ratio of nodes in the left and right subtrees from the root. This feature reflects the decision to make all nominal compounds in CCGbank right branching (Hock-

enmaier and Steedman, 2007).

CoHeads: records the heads of both halves of a coordination as indicated by the parser, along with the depth at which the head is found. This attempts to encode the conjunction dependencies in the tree as incorrect conjunction dependencies propagate through to other dependencies in the tree. Heads are encoded as lexical items and POS tags.

LexDep: CCG dependencies can be partially captured via the children of non-terminals in the tree. This feature is active for non-terminals with two children and encodes the heads of the children in terms of lexical items, POS tags, categories, and depth from the non-terminal. Dependencies involving punctuation are ignored as they are not assessed in the evaluation.

NumDeps: distinguishes between parses based on the log number of dependencies that they yield ignoring punctuation. Dependencies are located using the same heuristic as the LexDep feature.

TypeRaising: indicates the presence of unary type-raising in the tree. While type-raising is necessary to analyse some constructions in CCG, it has tightly restricted in the parser due to its power, and is expected to appear only rarely.

UnaryRule, BiUnaryRule: indicates the unary rules present in the tree and the bigram combinations of these rules. The unary rules do not include type-raising and are non-standard in CCG; they were added by Hockenmaier and Steedman (2007) to CCGbank for constructions such as clausal adjuncts, which are poorly handled by the formalism.

C&C Features: Finally, we also incorporate the dependency and normal-form features used by the C&C parser as described in Clark and Curran (2007). These features encode various combinations of word-category, word-POS, root-word, CCG rule, distance, and dependency information.

7 Evaluation Measures

We follow the CCG dependency evaluation methodology established by Hockenmaier (2003), using the EVALUATE scorer distributed with the C&C parser. It evaluates a CCG parse as a set of labeled dependencies consisting of the head, its lexical category, the child, and the argument slot that it fills. A dependency is considered correct only if all four elements match the gold standard.

	LP	LR	LF	AF
Baseline	87.19	86.32	86.75	84.80
Oracle 10	91.98	90.89	91.43	89.47
Oracle 50	93.43	92.26	92.84	90.96

Table 1: Baseline and oracle n-best parser performance over Section 00 of CCGbank.

Statistical significance was calculated using the test described in Chinchor (1992), which measures the probability that the two sets of responses are drawn from the same distribution. A score below 0.05 is considered significant.

We report labeled precision (LP), labeled recall (LR), and labeled F-score (LF) results over gold standard POS tags and labeled F-score over automatically assigned POS tags (AF).

8 Results

8.1 Oracle Performance

Reranking is dependent on high-quality parses from the n-best parser. As seen in Table 1, the oracle labeled dependency F-score of the n-best C&C parser is 92.48% given a perfect reranker over 50-best parses. This is a significant improvement over the baseline result of 86.75% and provides a solid basis for a reranker.

Our oracle score falls notably short of the 50-best oracle of 96.8% reported by Charniak and Johnson (2005), over a baseline of 89.7%. However, these numbers refer to the PARSEVAL score for constituency parses, so they are not directly comparable to our dependency recovery metric.

We present results in Tables 2 and 3 comparing the 1-best C&C parser using the normal-form model (Clark and Curran, 2007), randomized baselines (choosing a parse at random from the n-best list), and the reranking C&C parser in labeled dependency recovery over Section 00 of CCGbank. Our best result for 10-best reranking is an F-score of 87.13% with gold POS tags and 85.22% with automatically assigned POS tags. This is achieved using the regression setup and all features without pruning. The best result for 50-best reranking is F-scores of 87.08% and 85.23% respectively, using the classification setup with all features and a pruning value of 2. These two results are both statistically significant improvements over the baseline parser.

Randomly choosing a parse from the n-best list

	t	LP	LR	LF	AF
Baseline	-	87.19	86.32	86.75	84.80
Random	-	85.40	84.46	84.93	83.00
Class+CJ	0	**87.21**	**86.06**	**86.63**	84.81
	2	87.16	85.98	86.57	**84.82**
	5	87.12	85.95	86.53	84.74
Class+CCG	0	87.17	86.18	86.67	**84.75**
	2	**87.19**	**86.19**	**86.69**	84.74
	5	87.11	86.09	86.59	84.68
Class+ALL	0	**87.32**	**86.32**	**86.82**	**84.85**
	2	87.29	86.29	86.78	84.82
	5	87.23	86.25	86.74	84.79
Regress+CJ	0	**86.96**	**85.99**	**86.47**	**84.58**
	2	86.75	85.76	86.26	84.34
	5	86.69	85.72	86.20	84.30
Regress+CCG	0	**87.27**	**86.41**	**86.83**	**85.08**
	2	87.05	86.12	86.58	84.70
	5	86.96	86.08	86.52	84.73
Regress+ALL	0	**87.60**	**86.67**	**87.13**	**85.22**
	2	87.42	86.47	86.94	85.00
	5	87.41	86.50	86.95	84.96

.

Table 2: 10-best reranking performance on Section 00 of CCGbank for various combinations of features, pruning values t, and classification and regression experiments. Bolded scores are the highest for the feature set and approach.

	t	LP	LR	LF	AF
Baseline	-	87.19	86.32	86.75	84.80
Random	-	83.90	82.58	83.24	81.50
Class+CJ	0	**86.93**	**85.87**	**86.40**	**84.69**
	2	86.72	85.54	86.12	84.48
	5	86.77	85.61	86.19	84.56
Class+CCG	0	87.17	86.10	86.63	84.62
	2	87.14	86.07	86.60	**84.66**
	5	**87.29**	**86.17**	**86.72**	**84.66**
Class+ALL	0	87.38	86.29	86.83	84.91
	2	**87.61**	**86.56**	**87.08**	**85.23**
	5	87.30	86.22	86.76	84.74
Regress+CJ	0	**86.49**	**85.64**	**86.07**	84.22
	2	86.44	85.46	85.95	**84.32**
	5	86.32	85.28	85.80	84.12
Regress+CCG	0	**87.08**	**86.15**	**86.61**	84.65
	2	87.00	86.06	86.53	84.66
	5	87.07	86.08	86.57	**84.72**
Regress+ALL	0	**87.28**	**86.30**	**86.79**	**84.89**
	2	86.73	85.77	86.25	84.43
	5	87.04	86.06	86.55	84.66

Table 3: 50-best reranking performance on Section 00 of CCGbank for various combinations of features, pruning values t, and classification and regression experiments.

	LP	LR	LF	AF
Best	87.60	86.67	87.13	85.22
-CoPar	87.47	86.57	87.02	85.11
-CoLenPar	87.53	86.59	87.06	85.17
-Heavy	87.44	86.55	86.99	85.09
-RightBranch	87.59	86.67	87.13	85.17
-SubjVerbAgr	**87.26**	**86.26**	**86.76**	**84.88**
-Edges	**87.11**	**86.22**	**86.67**	**84.87**
-Heads	87.55	86.65	87.10	85.26
-HeadTree	87.61	86.64	87.12	85.22
-Neighbours	87.50	86.59	87.05	85.16
-NGramTree	87.51	86.55	87.03	85.08
-Rule	87.54	86.58	87.05	85.14
-SynSemHeads	87.42	86.47	86.94	85.07
-Word	87.44	86.51	86.97	85.11
-WProj	87.44	86.55	86.99	85.09
-Balance	87.44	86.53	86.98	85.17
-CoHeads	87.38	86.47	86.93	84.90
-LexDep	87.52	86.58	87.04	85.15
-NumDeps	87.40	86.54	86.97	85.04
-TypeRaising	87.41	86.48	86.95	85.05
-UnaryRule	87.58	86.68	87.13	85.27
-BiUnaryRule	87.55	86.64	87.09	85.20
-C&C	87.44	86.48	86.96	84.97

Table 4: Subtractive analysis on the top performing 10-best model on Section 00. Bold indicates a statistically significant change from the baseline.

results in much poorer performance than the 1-best baseline. All our experiments produced results that were significantly higher than the randomized result, indicating that our approaches were learning useful features from the training data. Even though the oracle scores increase with n (as shown in Table 1), the overall parse quality deteriorates.

Regression was generally more successful for 10-best reranking, while classification was better for 50-best reranking. However, there were very few cases where a statistically significant difference in performance was observed between regression and classification approaches.

8.2 Features

We investigated the performance of three sets of features: those adapted from Charniak and Johnson (2005) (CJ), our new features (CCG), and the union of the two sets (ALL). The log score and rank of each parse was included as core features in every experiment. In general, more features improved performance. The best results were produced using all of the possible features in the reranker model. In terms of the top F-score for

	LP	LR	LF	AF
Baseline	87.19	86.32	86.75	84.80
SubjVerbAgr	86.76	85.87	86.31	84.33
Edges	86.29	85.45	85.87	83.95
Both	86.30	85.49	85.89	83.95

Table 5: 10-best isolation experiments for the SubjVerbAgr and Edges features on Section 00 using regression and no pruning.

	LP	LR	LF	AF
Baseline	87.75	86.98	87.36	85.07
Reranker	**87.98**	**87.21**	**87.59**	**85.36**

Table 6: Baseline and final reranker performance over Section 23 of CCGbank with the normal-form model.

each set of features, the CCG-specific features were better than the Charniak and Johnson (2005) features by a statistically significant margin. This held in all experiments except one (10-best classification), indicating that features tailored to CCG trees and dependency evaluation were more discriminative between good and bad CCG parses. This also implies that for reranking to improve the accuracy of a parser, the features must target that parser and the nature of its evaluation. Features producing state-of-the-art performance for the Charniak reranker had no positive impact on CCG parsing in isolation.

We conducted subtractive feature analysis on our best performing model (10-best regression with all features and no pruning) to investigate the contribution of individual features. Features were individually removed and the reranker was retrained and retested on Section 00. The removal of the SubjVerbAgr and Edges features are statistically significant, while the removal of any other single feature results in a non-significant decrease in F-score. We then performed an isolation experiment, training and testing the reranker using just the SubjVerbAgr and Edge features with the log score and rank from the parser. Table 5 shows that these features do significantly worse than the baseline in isolation, indicating that it is the combination of features together which produces the improved performance.

8.3 Pruning

We found that increased feature pruning had a negative impact on parsing accuracy. None of our experiments showed a significant improvement with higher pruning values, as opposed to Charniak and Johnson (2005) who found the count-based pruning to be useful. The best performing systems overall used pruning values of 0 or 2, implying that the pruning strategy is ineffective with respect to performance over such a varied set of features. One area where pruning does help is in the training times for the reranker: some experiments are nearly twice as fast with a pruning value $t = 5$ compared to $t = 0$. However, as this cost must only be paid once, the benefit of pruning with respect to actual parsing time is negligible.

8.4 Final Results

Table 6 summarises the performance of our best reranker model against the baseline normal-form model on Section 23 of CCGbank. We achieve statistical significant improvement in F-score over the baseline. However, in real terms the change in F-score is small, indicating that reranking may not guarantee performance improvements even if it is carefully targeted to the parser.

9 Conclusion

We have implemented a maximum entropy reranker for the C&C CCG parser, building on the methodology and features of Charniak and Johnson (2005) and extending the approach with new features. We have found that performance improvements from reranking stem from targeting the reranker features at the parser and its evaluation: features tailored to CCG perform better than PCFG-style features in isolation. Our best system achieves an of 87.59%, which is a statistically significant improvement over the baseline parser.

The reranker scales with the efficiency of calculating features on parse trees. The features described in this paper require time linear in the number of nodes in the tree. However, the reranker is currently implemented as an external post-processing step. This leads to an order of magnitude speed decrease; future work will include integrating the reranker into the parser itself to mitigate this speed impact.

The improvement in accuracy that we achieve is small in absolute terms, showing that reranking is a considerably difficult task. However, continued

improvements such as this one in parsing accuracy will benefit the variety of downstream applications that utilise parsing for practical NLP tasks.

Acknowledgments

This work was supported by Australian Research Council Discovery grants DP0665973 and DP1097291, the Capital Markets CRC and a University of Sydney Merit Scholarship.

References

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. 2004. Wide-Coverage Semantic Representations from a CCG Parser. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*, pages 1240–1246, Geneva, Switzerland, August.

Johan Bos, James R. Curran, and Edoardo Guzzetti. 2007. The Pronto QA system at TREC-2007: Harvesting Hyponyms, Using Nominalisation Patterns, and Computing Answer Cardinality. In *Proceedings of the Sixteenth Text REtrieval Conference (TREC 2007)*, pages 726–732, Gaitersburg, Maryland, USA.

Forrest Brennan. 2008. k-best Parsing Algorithms for a Natural Language Parser. Master's thesis, University of Oxford.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 173–180, Ann Arbor, Michigan, USA, June.

Nancy Chinchor. 1992. The statistical significance of the MUC-4 results. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 30–50, McLean, Virginia, June.

Stephen Clark and James R. Curran. 2007. Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models. *Computational Linguistics*, 33(4):493–552.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania, Philadelphia, Pennsylvania, USA.

Michael Collins. 2000. Discriminative Reranking for Natural Language Parsing. In *Proceedings of the 17th International Conference on Machine Learning (ICML-00)*, pages 175–182, Palo Alto, California, USA, June.

Steve DeNeefe and Kevin Knight. 2009. Synchronous Tree Adjoining Machine Translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-09)*, pages 727–736, Singapore, August.

Abdessamad Echihabi and Daniel Marcu. 2003. A Noisy-Channel Approach to Question Answering. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 16–23, Sapporo, Japan, July.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Computational Linguistics*, 28(3):245–288.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Julia Hockenmaier. 2003. Parsing with Generative Models of Predicate-Argument Structure. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 359–366, Sapporo, Japan, July.

Liang Huang and David Chiang. 2005. Better k-best Parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology (IWPT-05)*, pages 53–64, Vancouver, British Columbia, Canada, October.

Liang Huang. 2008. Forest Reranking: Discriminative Parsing with Non-Local Features. In *Proceedings of the Human Language Technology Conference at the 45th Annual Meeting of the Association for Computational Linguistics (HLT/ACL-08)*, pages 586–594, Columbus, Ohio, June.

Mark Johnson and Ahmet Engin Ural. 2010. Reranking the Berkeley and Brown Parsers. In *Proceedings of Human Language Technologies: the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-10)*, pages 665–668, Los Angeles, California, USA, June.

Dan Klein and Christopher D. Manning. 2001. Parsing and Hypergraphs. In *Proceedings of the 7th International Workshop on Parsing Technologies (IWPT-01)*, Beijing, China, October.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective Self-Training for Parsing. In *Proceedings of the 2006 Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT/NAACL-06)*, pages 152–159, New York City, New York, USA, June.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, pages 433–440, Sydney, Australia, July.

Laura Rimell, Stephen Clark, and Mark Steedman. 2009. Unbounded Dependency Recovery for Parser Evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-09)*, pages 813–821, Singapore, August.

Federico Sangati, Willem Zuidema, and Rens Bod. 2009. A generative re-ranking model for dependency parsing. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT-09)*, pages 238–241, Paris, France, October.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative Reranking for Machine Translation. In *Proceedings of the 2004 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-04)*, pages 177–184, Boston, Massachusetts, USA, May.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, Massachusetts, USA.

Michael White and Rajakrishnan Rajkumar. 2009. Perceptron Reranking for CCG Realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-09)*, pages 410–419, Singapore, August.

Repurposing Corpora for Speech Repair Detection: Two Experiments

Simon Zwarts, Mark Johnson, Robert Dale
Center for Language Technology,
Department of Computing,
Macquarie University
{simon.zwarts, mark.johnson, robert.dale}@mq.edu.au

Abstract

Unrehearsed spoken language often contains many disfluencies. If we want to correctly interpret the content of spoken language, we need to be able to detect these disfluencies and deal with them appropriately. In the work described here, we use a statistical noisy channel model to detect disfluencies in transcripts of spoken language. Like all statistical approaches, this is naturally very data-hungry; however, corpora containing transcripts of unrehearsed spoken language with disfluencies annotated are a scarce resource, which makes training difficult.

We address this issue in the following ways: First, since written textual corpora are much more abundant than speech corpora, we see whether using a large text corpus to increase the data available to our language model component delivers an improvement. Second, given that most spoken language corpora are not annotated with disfluencies, we explore the use of Expectation Maximisation to mark the disfluencies in such corpora, so as to increase the data availability for our complete model.

In neither case do we see an improvement in our results. We discuss these results and the possible reasons for the negative outcome.

1 Introduction

We are interested in improving speech disfluency detection in transcripts of spontaneous spoken language. Many models have been proposed for this task in the literature; the best performing models so far are statistical by nature and have large data needs.

A statistical natural language processing algorithm typically has two important components: a model that describes the behaviour of interest, and the training data which is necessary to guide that model. It has been observed that simple algorithms can outperform more complex models when these simple algorithms have the advantage in terms of the amount of data available; so, for example, Brill and Banko (2001) argue that more data is more important than better algorithms for some natural language processing tasks. It is this insight that drives the work described in this paper.

Our current approach to speech disfluency detection is trained on manually-constructed spoken language corpora which contain annotations of all disfluencies as part of the transcription process. Our model is based on the noisy channel model and consists of a language model and a channel model. As we have reported elsewhere (Zwarts et al., 2010), we are able to achieve reasonable results when using Switchboard data: we obtain an F-score of 0.757 in determining which constituents of an utterance belong to a disfluency.

We would like to see if we can improve on our previously reported performance by adding more data. Our language model does not need any special annotation, and so our first set of experiments investigates whether we can improve results by vastly increasing the training data for the language model. The task of increasing the training data for the channel model is a more difficult one, since here we require the annotation of disfluencies. Our second set of experiments therefore investigates whether, given our existing annotated data, we can use Expectation Maximisation in a semi-supervised approach to automatically anno-

Simon Zwarts, Mark Johnson and Robert Dale. 2010. Repurposing Corpora for Speech Repair Detection: Two Experiments. In *Proceedings of Australasian Language Technology Association Workshop*, pages 99−106

tate a larger collection of unannotated speech data, by learning what sentences typically look like around disfluencies and what the typical structure of disfluencies is.

The remainder of this paper is structured as follows. In Section 2 we first present some background on disfluencies and their structure in spontaneous speech. Section 3 discusses the current state of the art in disfluency detection models, motivates the choice of the model we use, and describes some of its intricacies and the data sets we use. Section 4 investigates the language model component of our model and explores whether we can improve this component; we provide the results obtained when using a language model that is several orders of magnitude larger than the language model used in our previous work. Section 5 investigates a more radical approach to address our data needs: we alter the training data for both the language model and the channel model.

It turns out that neither of these experiments results in an improvement in disfluency detection. Section 6 draws some conclusions from our results, and suggests some ways forward based on this experience.

2 Speech Repairs

We adopt the terminology and definitions introduced by Shriberg (1994) to discuss disfluencies. We are particularly interested in those disfluencies which are categorised as **repairs**. These are the most interesting and also the hardest disfluencies to identify, since they are not marked by a characteristic vocabulary. Shriberg (1994) identifies and defines three distinct parts of a such a disfluency, referred to as the **reparandum**, the **interregnum** and the **repair**. Consider the following utterance:

$$\underbrace{\textit{I want a flight}\ \overbrace{\textit{to Boston,}}^{\text{reparandum}}}_{\textit{uh, I mean}\ \textit{to Denver}\ \textit{on Friday}} \qquad (1)$$

$$\underbrace{\phantom{\textit{uh, I mean}}}_{\text{interregnum}}\ \underbrace{\phantom{\textit{to Denver}}}_{\text{repair}}$$

The reparandum *to Boston* is the part of the utterance that is being 'edited out'; the interregnum *uh, I mean* is a filler, which may not always be present; and the repair *to Denver* replaces the reparandum.

Given an utterance that contains such a disfluency, we want to be able to correctly detect the start and end positions of each of these three components. We can think of each word in an utterance as belonging to one of four categories: fluent material, reparandum, filler, or repair. We can then assess the accuracy of techniques that attempt to detect disfluencies by computing precision and recall values for the assignment of the correct categories to each of the words in the utterance, as compared to the gold standard as indicated by annotations in the corpus.

3 Disfluency Detection Models

3.1 Related Work

A number of different techniques have been proposed for automatic disfluency detection. Schuler et al. (2010) propose a Hierarchical Hidden Markov Model approach; this is a statistical approach which builds up a syntactic analysis of the sentence and marks those subtrees which it considers to be made up of disfluent material. Although this is one of the few models that actually builds up a syntactic analysis of the utterance being analysed, its final F-score for fluency detection is lower than that of other models.

Snover et al. (2004) investigate the use of purely lexical features combined with part-of-speech tags to detect disfluencies. This approach is compared against approaches which use primarily prosodic cues, and appears to perform equally well. However, the authors note that this model finds it difficult to identify disfluencies which by themselves are very fluent. The edit repairs which are the focus of our work typically have this characteristic: when a speaker edits her speech for meaning-related reasons, rather than errors that arise from performance, the resulting disfluency can be by itself fluent. We can see this in Example (1): the repair and the reparandum are equally fluent. This makes it difficult to distinguish reparanda as being part of disfluencies when only lexical cues are available. Since the transcripts we work with do not have prosodic cues annotated, we need to look elsewhere for a solution to this problem.

Noisy Channel models have done very well in this area; the work of Johnson and Charniak (2004) explores such an approach. This approach performs very well when compared

with other approaches. Johnson et al. (2004) adds some handwritten rules to the noisy channel model, providing the current state of the art in disfluency detection. Lease and Johnson (2006) also use this approach, but they are particularly interested in finding fillers; they use early filler detection and deletion in this model.

The following section describes the noisy channel approach in more detail.

3.2 The Noisy Channel Approach

The approach we build on is that first introduced by Johnson and Charniak (Johnson and Charniak, 2004). This approach is modular by nature, making it possible to interchange different sub-components. The original paper explores the use of different types of language models, and demonstrates how some models provide better overall performance than others. In the remainder of this section we describe the basics of this approach.

To find repair disfluencies, a noisy channel model is used. For an observed utterance with disfluencies y, we wish to find the most likely source utterance, $\hat{x}$, where:

$$\begin{aligned} \hat{x} &= argmax_x\ p(x \mid y) \\ &= argmax_x\ p(y \mid x)\, p(x) \end{aligned} \tag{2}$$

Here we have a channel model $p(y|x)$ which generates an utterance y given a source x and a language model $p(x)$. We assume that x is a substring of y, i.e., the source utterance can be obtained by marking words in y as being disfluent elements and effectively removing them from this utterance.

The task of the language model is to assess the fluency of the sentence when the reparandum and the interregnum have been removed. As noted above, Johnson and Charniak (2004) experiment with variations on the language model; they report results for a bigram model, a trigram model, and a language model using the Charniak Parser (Charniak, 2001). Their results demonstrate that the parser model outperforms the bigram model by 5%.

The channel model is based on the intuition that a reparandum and a repair are generally very alike; it is often the case that the repair is almost a copy of the reparandum. In the training data, over 60% of the words in a reparandum are lexically identical to the words in the corresponding repair. Example (1) again provides an example of this: half of the repair is lexically identical to the reparandum. The channel model therefore gives the highest probability when the reparandum and repair are identical. When the potential reparandum and potential repair are not identical, the channel model performs deletion, insertion or substitution operations. The probabilities for these operations are defined on a lexical level and are derived from the training set text. This channel model is formalised using a Synchronous Tree Adjoining Grammar (S-TAG) (Shieber and Schabes, 1990), which matches words from the reparandum to the repair. The weights for these S-TAG rules are learnt from the training text, where reparanda and repairs are aligned to each other using a minimum edit-distance string aligner.

For a given utterance, every possible utterance position might be the start of a reparandum, and every given utterance position thereafter might be the start of a repair (to limit complexity, a maximum distance between these two points is imposed). Every disfluency in turn can have an arbitrary length (again up to some maximum to limit complexity). After every possible disfluency other new reparanda and repairs might occur; the model does not attempt to generate crossing or nested disfluencies, although they do very occasionally occur in practice. To find the optimal selection for reparanda and repairs, all possibilities are calculated and the one with the highest probability is selected.

A chart is filled with all the possible start and end positions of reparanda, interregna and repairs; each entry consists of a tuple $\langle rm_{\text{begin}}, ir_{\text{begin}}, rr_{\text{begin}}, rr_{\text{end}} \rangle$, where rm is the reparandum, ir is the interregnum and rr is the repair. A Viterbi algorithm is used to find the optimal path through the utterance, ranking each chart entry using the language model and channel model. The language model, a bigram model, can be easily calculated given the start and end positions of all disfluency components. The channel model is slightly more complicated because an optimal alignment be-

tween reparandum and repair needs to be calculated. This is done by extending each partial analysis by adding a word to the reparandum, the repair or both. The start position and end position of the reparandum and repair are given for this particular entry. The task of the channel model is to calculate the highest probable alignment between reparandum and repair. This is done by initialising with an empty reparandum and repair and 'growing' the analysis one word at a time. Using a similar approach to that used in calculating the edit-distance between reparandum and repair, the reparandum and repair can both be extended with one of four operations: deletion (only the reparandum grows), insertion (only the repair grows), substitution (both grow), or copy (both grow). When the reparandum and the repair have their length corresponding to the current entry in the chart, the channel probability can be calculated. Since there are multiple alignment possibilities, we use dynamic programming to select the most probable solutions. The probabilities for insertion, deletion and substitution are estimated from the training corpus. We use a beam-search strategy to find the final optimum when combining the channel model and the language model.

3.3 The Data Set

As a data set to work with, we use the Switchboard part of the Penn Treebank 3 corpus. The Switchboard Corpus is made up of transcriptions of spontaneous conversations between two partners during a telephone call. The Penn Treebank 3 corpus adds manual annotation of disfluencies to the Switchboard corpus; additionally it provides part-of-speech information for all the words.

The disfluency annotation distinguishes between repair disfluencies and filled pauses. When repair disfluencies are present the structure of the disfluency is annotated: these annotations indicate which part of the disfluency is the reparandum, which part is the interregnum and which part is the repair. The following is an example:

```
[ i/NN think/VBP it/PRP was/VBD +
{F yeah/UH } i/NN think/VBP that/WDT
was/VBD ] the/DT only/JJ question/NN
E_S
```

Here we see the reparandum (*I think it was*), the interregnum (*yeah*) and the repair (*I think that was*) annotated.

Following Johnson and Charniak (2004), we use all of sections 2 and 3 of the corpus for training; we use conversations 4[5-9]* for a held-out training set; and conversations 40*, 41[0-4]* and 415[0-3]* as the held-out test set.

The corpus is not immense: a little over 100K sentences are present in the training data. This means that in the the held-out training set, and presumably also in the test set, there are many out-of-vocabulary words and a very large incidence of low frequency vocabulary items, for which we struggle to find the appropriate statistical values.

Our earlier work just used this data. When we use the noisy channel model as described in Section 3.2 using the Switchboard data, as described above, we can compute precision and recall over a held-out test set. Comparing our output against the gold standard annotation, we can compute performance over disfluencies detected. This results in an F-score of 0.757.[1]

4 Extending The Language Model

4.1 Background

As we noted earlier, previous work by Johnson and Charniak (2004) has shown that the language model component of the model has an important role: when more sophisticated language models are used, the overall performance can be increased significantly.

An important aspect of our earlier work is that we were particularly interested in processing incoming speech incrementally, detecting disfluencies as soon after they happen as possible. However, incremental processing makes the use of a reranker, as adopted in Johnson and Charniak's more sophisticated model, a less viable option. Our initial language model was trained on the fluent part of the Switchboard Corpus: this consists of the utterances with the reparanda and the interregna removed. The bigram model is trained on the counts from the same data and, as mentioned above, this contains approximately 100k sentences. This would not typically be considered a large data set in terms of language modelling

[1] The F-score reported here is the harmonic mean between precision and recall.

(Harb et al., 2009); consequently, we look to increasing the amount of data used in our language model as an alternative means of improving results.

4.2 Motivation

Using a larger set of data for the language model allows us to answer two questions:

1. Has the current bigram model reached its limit? Previous research has shown that reranking the results of a model using bigrams still leaves room for improvement. We assume that the bigram model itself also has scope for improvement, since there is still a large set of out-of-vocabulary words in the held-out training set, and an even larger set of low frequency words for which it is difficult to calculate the proper probabilities accurately. Can we improve the bigram model when we increase its training data?

2. Does the nature of the data used matter? Our language model is currently specifically trained on the fluent parts of transcribed spontaneous speech. Most language models, however, are built on primarily written texts, given their greater availability. Would the use of a vastly greater quantity of written data offset the impact of the change in the nature of that data?

4.3 Experimental Setup

We decided to use the Google Web 1T corpus, which contains English word n-grams and their observed frequency counts. The n-gram counts were generated from approximately 1 trillion word tokens of text from publicly accessible Web pages, much larger than the number of words in Switchboard (roughly 700K). In their description of this corpus, the authors suggest the corpus should be useful for language models and for speech recognition; our experiments are one test of this claim.

The Web 1T corpus records counts for unigrams up to 5-grams. We only use the bigram part of this corpus, but this still introduces memory problems. The entire bigram counts take up more than 8.8GB, which is more than we can fit into memory. This dataset is also vastly larger than the test portion of the corpus. Since our evaluation is only carried out over the test portion, we do not need to memorise any bigrams which are not present in this portion; so, we can use the process of prefiltering (Goodman, 2001) the bigrams of the larger corpus against the test set. This process does not mean we are using test data during our experiments: it is only an optimisation strategy that avoids loading into memory bigrams which will not be used later. After this process of prefiltering we are left with only 10MB of bigram data, which easily fits into memory.

Our baseline model is the model as described by Johnson and Charniak (2004), using the traditional Switchboard part of the Penn Treebank 3 data to derive the language model. Our alternative model has the language model replaced with the Web 1T bigram probabilities. If this approach proves to be successful, we might consider using a language model which is a hybrid consisting of both the data derived from the Switchboard part (which is arguably closer in nature to the data we ultimately want to process), and the Web 1T data (which might deliver statistics for the tail end of the Zipfian curve). We can use the held-out training set for tuning purposes to decide on the relative weight to be accorded to these two language models.

4.4 Results

The baseline model, using only the Switchboard data with a bigram language model, results in an F-score of 0.757. Our new model, which uses a vastly larger data set for bigram modelling, results in an F-score of 0.739.

The most obvious explanation for this is that text derived from Web pages is not a good source of data for building a language model for spoken language: Even when disfluencies are removed from spontaneous spoken speech, the language used is still very different from written text. In general terms, this, of course, is not a new or surprising result; Biber (1988), and many others since, have drawn attention to the differences between spoken and written language. What is perhaps more surprising is that these differences appear to impact not only, for example, at the syntactic level, but also at the level of bigram occurrences.

5 A Semi-supervised Learning Approach

5.1 Background

Our noisy channel approach has two components, the language model and the channel model. The approach in the previous section investigate whether it would be possible to use a very large data set for the language model. In this section we investigate whether it is possible to address the data needs for both the language model and the channel model.

5.2 Motivation

Our objective here is to use a data set of transcribed spontaneous speech which is more than an order of magnitude larger than the data available in the Switchboard part of the Penn Treebank 3 corpus. With this approach we would hope to answer the following three questions:

1. Is it possible to significantly increase the performance of this model, without the application of a more complicated approach? As noted above, complications like reranking via parser results are difficult to apply in our incremental processing scenario.

2. What does the performance curve of this model look like? When we increase training data, how does the overall performance increase? Our interest here is in providing a more definitive assessment as to how much data is needed to reach the upper limit of performance with the current model.

3. Can we use a Expectation Maximisation approach in order to increase our data needs? Disfluency-annotated data is very costly to develop; we want to see whether we can avoid this by automatically deriving such annotations using a semi-supervised approach.

5.3 Experimental Setup

In the experiments described here, we explore increasing the training data by using additional speech corpora.

The Fisher English Training Speech Transcripts represent the collection of conversational telephone speech (CTS) that was created at the LDC during 2003. It contains transcript data for 5,850 complete conversations, each lasting up to 10 minutes. The Fisher Speech Corpora Part I and II together contain a little over 2 million sentences, which is considerably more than is present in the Switchboard part of Penn Treebank 3. However, the only disfluency annotation the corpus contains is the marking of partially uttered words. Filled pauses and the more complicated repair disfluencies are not annotated.

Besides lacking disfluency annotations, the Fisher corpora also lacks part-of-speech tags. Our channel model uses these tags to build up an alignment between reparandum and repair: since it assumes reparandum and repair are a rough copy of each other, it uses the part-of-speech tags to inspect how similar these parts are, and these tags are especially useful when the words in the reparandum and repair are not exact lexical copies. Since it is too costly to obtain manually-annotated tags for our corpus, we use the Brill Tagger (Brill, 1993) to automatically annotate the Fisher corpus with part-of-speech tags, using the same tag set as is used in the Penn Treebank 3 data.

Once the part-of-speech tags are available, we can use our original noisy channel model to annotate this corpus for disfluencies. We can then add this newly acquired data to the existing training data. In this way, we hope to acquire new statistical insights into what types of disfluencies are common, and what sentences typically look like around these disfluencies.

In order not to dominate the manually-annotated data from the Penn Treebank 3 data with the more noisy Fisher data, we would as a first step like to use them in similar proportions. We initially only use the first part of the Fisher data, of a similar size to the Penn Treebank 3 data. When this approach results in increased performance, we can re-annotate this same part with the newly built model, which hopefully will result in a better analysis of the Fisher corpus. When this iterative process reaches its maximum score, we can then investigate whether we can use more of the Fisher data. Because the original Penn Treebank 3 data is hand-annotated and

is more accurate, it might prove to be helpful to not weight counts from both corpora equally: doing so might make the model drift away from disfluency detection to another annotation scheme which fits the data better, but which ultimately could be meaningless. We can use the held-out training data to properly decide on a weighting scheme between both corpora.

The baseline which we compare against is the standard model as described by (Johnson and Charniak, 2004), using the Penn Treebank 3 data set only.

5.4 Results

The baseline model using only the Switchboard data part results in an F-score of 0.757 using the bigram language model. When we add the Fisher data as part of our training data we expect to achieve a higher performance; however in our experimental set-up we reached a final F-score of 0.742, which is actually a slight decrease in performance. This is disappointing, since Expectation Maximisation has proven to be a successful strategy in other area of natural language processing.

There are several possible reasons as to why this approach turned out to be less fruitful here. First, note that the training process heavily relies on part-of-speech information. However, the Brill Tagger was not initially built for spontaneous speech, and may have introduced errors which impact on our final results. An alternative explanation could be that the Fisher corpus and Switchboard corpus exhibit a different type of language use, although this seems to be less likely. Finally, it could be the case that our model does not perform well enough on the Fisher data to actually help out in a new iteration, although for the expectation maximisation step an F-score of around 0.75 should not be a hindrance to building a new model for a next iteration. Significant gains using Expectation Maximisation have been achieved in other spoken language processing tasks starting from this absolute score (Sandrini and Federico, 2003). We are not yet convinced, therefore, that this direction is a dead-end.

6 Conclusions and Future Work

Statistical models are typically data-hungry, and so a problem arises in any domain where data is scarce. In this paper, we have explored two different approaches that aim to increase the amount of data usable by our disfluency detection model. We have investigated the use of Google 1T, the largest written text corpus available to date for language modelling. This proved to have a negative impact on our results. We hypothesise that this is most likely because of the differences between written and spoken language. The result means that one should be cautious about using corpora derived from textual sources when working with conversational speech.

In our second set of experiments, we tried to use Expectation Maximisation to provide more data for use in our channel model. Again, the results here were negative.

Ultimately, although it may be true that more data can be more important than smarter algorithms, it needs to be the right data.

For future work we intend to experiment with a different part-of-speech tagger. We also suspect that a different source of data may require retuning of our model: currently our model is trained towards the Switchboard data, and even though this is the only data for which we have gold standard annotations, we would like to retune the model parameters when using the Fisher corpus. We can still use the held-out Switchboard data set to retune the model operating on Switchboard and Fisher. The current approach uses a noisy channel model, in which the language model and channel model are weighted equally. We could transform this into a log linear model which will allow us not only to weight the language model and channel model differently, but also will allow us to use multiple models. We can develop separate language models from different sources (Web1T, Fisher, Switchboard) and separate channel models derived from different sources (Fisher via EM training, Switchboard) and use them simultaneously. Using a log linear approach we can individually weight these components using the held-out training set to achieve optimal performance. This almost guarantees that perfor-

mance will not degrade, as in a worst case scenario the learner can turn off new data sources and use the old model; but even when there is a little information in any of the additional sources, performances is expected to go up. Finally, using such a model will allow us to add any computable feature, making it possible to go beyond language and channel models. As an additional advantage, the individual learnt weights will be a good indication of the relative value of each data source.

Acknowledgements

This work was supported by the Australian Research Council as part of the Thinking Head Project, ARC/NHMRC Special Research Initiative Grant # TS0669874. We thank the anonymous reviewers for their helpful comments.

References

Douglas Biber. 1988. *Variation across speech and writing*. Cambridge University Press.

Erik Brill and Michele Banko. 2001. Mitigating the Paucity-of-Data Problem: Exploring the Effect of Training Corpus Size on Classifier Performance for Natural Language Processing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Eric Brill. 1993. *A corpus-based approach to language learning*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 124–131.

Joshua T. Goodman. 2001. A bit of progress in language modeling. Technical report, Microsoft Research.

Boulos Harb, Ciprian Chelba, Jeffrey Dean, and Sanjay Ghemawat. 2009. Back-Off Language Model Compression. In *Proceedings of Interspeech*, pages 325–355.

Mark Johnson and Eugene Charniak. 2004. A tag-based noisy channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 33–39.

M. Johnson, E. Charniak, and M. Lease. 2004. An improved model for recognizing disfluencies. In *in Proceedings of Conversational Speech Rich Transcription Fall Workshop*.

Matthew Lease and Mark Johnson. 2006. Early deletion of fillers in processing conversational speech. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 73–76.

Vanessa Sandrini and Marcello Federico. 2003. Spoken Information Extraction from Italian Broadcast News. *Advances in Information Retrieval Lecture Notes in Computer Science*, 2633.

William Schuler, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2010. Broad-Coverage Parsing using Human-Like Memory Constraints. *Computational Linguistics*, 36(1):1–30.

Stuart M. Shieber and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, pages 253–258.

Elizabeth Shriberg. 1994. *Preliminaries to a Theory of Speech Disuencies*. Ph.D. thesis, University of California, Berkeley.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2004. A Lexically-Driven Algorithm for Disfluency Detection. In *Proceedings of Human Language Technologies and North American Association for Computational Linguistics*, pages 157–160.

Simon Zwarts, Mark Johnson, and Robert Dale. 2010. Detecting speech repairs incrementally using a noisy channel approach. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1371–1378, Beijing, China, August. Coling 2010 Organizing Committee.