

Web Text Corpus for Natural Language Processing

Vinci Liu and James R. Curran

School of Information Technologies

University of Sydney

NSW 2006, Australia

{vinci, james}@it.usyd.edu.au

Abstract

Web text has been successfully used as training data for many NLP applications. While most previous work accesses web text through search engine hit counts, we created a Web Corpus by downloading web pages to create a topic-diverse collection of 10 billion words of English. We show that for context-sensitive spelling correction the Web Corpus results are better than using a search engine. For thesaurus extraction, it achieved similar overall results to a corpus of newspaper text. With many more words available on the web, better results can be obtained by collecting much larger web corpora.

1 Introduction

Traditional written corpora for linguistics research are created primarily from printed text, such as newspaper articles and books. With the growth of the World Wide Web as an information resource, it is increasingly being used as training data in Natural Language Processing (NLP) tasks.

There are many advantages to creating a corpus from web data rather than printed text. All web data is already in electronic form and therefore readable by computers, whereas not all printed data is available electronically. The vast amount of text available on the web is a major advantage, with Keller and Lapata (2003) estimating that over 98 billion words were indexed by Google in 2003.

The performance of NLP systems tends to improve with increasing amount of training data. Banko and Brill (2001) showed that for context-sensitive spelling correction, increasing the training data size increases the accuracy, for up to 1 billion words in their experiments.

To date, most NLP tasks that have utilised web data have accessed it through search engines, using only the hit counts or examining a limited number of results pages. The tasks are reduced to determining n-gram probabilities which are then estimated by hit counts from search engine queries. This method only gathers information from the hit counts but does not require the computationally expensive downloading of actual text for analysis. Unfortunately search engines were not designed for NLP research and the reported hit counts are subject to uncontrolled variations and approximations (Nakov and Hearst, 2005). Volk (2002) proposed a linguistic search engine to extract word relationships more accurately.

We created a 10 billion word topic-diverse *Web Corpus* by spidering websites from a set of seed URLs. The seed set is selected from the Open Directory to ensure that a diverse range of topics is included in the corpus. A process of text cleaning transforms the HTML text into a form useable by most NLP systems – tokenised words, one sentence per line. Text filtering removes unwanted text from the corpus, such as non-English sentences and most lines of text that are not grammatical sentences. We compare the vocabulary of the Web Corpus with newswire.

Our Web Corpus is evaluated on two NLP tasks. Context-sensitive spelling correction is a disambiguation problem, where the correction word in a confusion set (e.g. {their, they're}) needs to be selected for a given context. Thesaurus extraction is a similarity task, where synonyms of a target word are extracted from a corpus of unlabelled text. Our evaluation demonstrates that web text can be used for the same tasks as search engine hit counts and newspaper text. However, there is a much larger quantity of freely available web text to exploit.

2 Existing Web Corpora

The web has become an indispensable resource with a vast amount of information available. Many NLP tasks have successfully utilised web data, including machine translation (Grefenstette, 1999), prepositional phrase attachment (Volk, 2001), and other-anaphora resolution (Modjeska et al., 2003).

2.1 Search Engine Hit Counts

Most NLP systems that have used the web access it via search engines such as Altavista and Google. N-gram counts are approximated by literal queries “ $w_1 \dots w_n$ ”. Relations between two words are approximated in Altavista by the NEAR operator (which locates word pairs within 10 tokens of each other). The overall coverage of the queries can be expanded by morphological expansion of the search terms.

Keller and Lapata (2003) demonstrated a high degree of correlation between n-gram estimates from search engine hit counts and n-gram frequencies obtained from traditional corpora such as the British National Corpus (BNC). The hit counts also had a higher correlation to human plausibility judgements than the BNC counts.

The web count method contrasts with traditional methods where the frequencies are obtained from a corpus of locally available text. While the corpus is much smaller than the web, an accurate count and further text processing is possible because all of the contexts are readily accessible. The web count method obtains only an approximate number of matches on the web, with no control over which pages are indexed by the search engines and with no further analysis possible.

There are a number of limitations in the search engine approximations. As many search engines discard punctuation information (especially when using the NEAR operator), words considered adjacent to each other could actually lie in different sentences or paragraphs. For example in Volk (2001), the system assumes that a preposition attaches to a noun simply when the noun appears within a fixed context window of the preposition. The preposition and noun could in fact be related differently or be in different sentences altogether.

The speed of querying search engines is another concern. Keller and Lapata (2003) needed to obtain the frequency counts of 26,271 test adjective pairs from the web and from the BNC for the task of prenominal adjective ordering. While extract-

ing this information from the BNC presented no difficulty, making so many queries to the Altavista was too time-consuming. They had to reduce the size of the test set to obtain a result.

Lapata and Keller (2005) performed a wide range of NLP tasks using web data by querying Altavista and Google. This included variety of generation tasks (e.g. machine translation candidate selection) and analysis tasks (e.g. prepositional phrase attachment, countability detection). They showed that while web counts *usually* outperformed BNC counts and *consistently* outperformed the baseline, the best performing system is usually a supervised method trained on annotated data. Keller and Lapata concluded that having access linguistic information (accurate n-gram counts, POS tags, and parses) outperforms using a large amount of web data.

2.2 Spidered Web Corpora

A few projects have utilised data downloaded from the web. Ravichandran et al. (2005) used a collection of 31 million web pages to produce noun similarity lists. They found that most NLP algorithms are unable to run on web scale data, especially those with quadratic running time. Halacsy et al. (2004) created a Hungarian corpus from the web by downloading text from the .hu domain. From a 18 million page crawl of the web a 1 billion word corpus is created (removing duplicates and non-Hungarian text).

A terabyte-sized corpus of the web was collected at the University of Waterloo in 2001. A breadth first search from a seed set of university home pages yielded over 53 billion words, requiring 960GB of storage. Clarke et al. (2002) and Terra and Clarke (2003) used this corpus for their question answering system. They obtained increasing performance with increasing corpus size but began reaching asymptotic behaviour at the 300-500GB range.

3 Creating the Web Corpus

There are many challenges in creating a web corpus, as the World Wide Web is unstructured and without a definitive directory. No simple method exists to collect a large representative sample of the web. Two main approaches exist for collecting representative web samples – IP address sampling and random walks. The *IP address sampling* technique randomly generates IP addresses

and explores any websites found (Lawrence and Giles, 1999). This method requires substantial resources as many attempts are made for each website found. Lawrence and Giles reported that 1 in 269 tries found a web server.

Random walk techniques attempt to simulate a regular undirected web graph (Henzinger et al., 2000). In such a graph, a random walk would produce a uniform sample of the nodes (i.e. the web pages). However, only an approximation of such a graph is possible, as the web is directed (i.e. you cannot easily determine all web pages linking to a particular page). Most implementations of random walks approximate the number of backward links by using information from search engines.

3.1 Web Spidering

We created a 10 billion word Web Corpus by spidering the web. While the corpus is not designed to be a representative sample of the web, we attempt to sample a *topic-diverse* collection of web sites. Our web spider is seeded with links from the *Open Directory*¹.

The Open Directory has a broad coverage of many topics on the web and allows us to create a topic-diverse collection of pages. Before the directory can be used, we had to address several coverage skews. Some topics have many more links in the Open Directory than others, simply due to the availability of editors for different topics. For example, we found that the topic University of Connecticut has roughly the same number of links as Ontario Universities. We would normally expect universities in a whole province of Canada to have more coverage than a single university in the United States. The directory was also constructed without keeping more general topics higher in the tree. For example, we found that Chicken Salad is higher in the hierarchy than Catholicism. The Open Directory is flattened by a rule-based algorithm which is designed to take into account the coverage skews of some topics to produce a list of 358 general topics.

From the seed URLs, the spider performs a breadth-first search. It randomly selects a topic node from the list and next unvisited URL from the node. It visits the website associated from the link and samples pages within the same section of the website until a minimum number of words have been collected or all of the pages were visited.

¹The Open Directory Project, <http://www.dmoz.org>

External links encountered during this process are added to the link collection of the topic node regardless of the actual topic of the link. Although websites of one topic tends to link to other websites of the same topic, this process contributes to a topic drift. As the spider traverses away from the original seed URLs, we are less certain of the topic included in the collection.

3.2 Text Cleaning

Text cleaning is the term we used to describe the overall process of converting raw HTML found on the web into a form useable by NLP algorithms – white space delimited words, separated into one sentence per line. It consists of many low-level processes which are often accomplished by simple rule-based scripts. Our text cleaning process is divided into four major steps.

First, different character encoding of HTML pages are transform into ISO Latin-1 and HTML named-entities (e.g. ` ` and `&`) translated into their single character equivalents.

Second, sentence boundaries are marked. Such boundaries are difficult to identify on web text as it does not always consists of grammatical sentences. A section of a web page may be mathematical equations or lines of C++ code. Grammatical sentences need to be separated from each other and from other non-sentence text. Sentence boundary detection for web text is a much harder problem than newspaper text.

We use a machine learning approach to identifying sentence boundaries. We trained a Maximum Entropy classifier following Ratnaparkhi (1998) to disambiguate sentence boundary on web text, training on 153 manually marked web pages. Systems for newspaper text only use regular text features, such as words and punctuations. Our system for web text uses HTML tag features in addition to regular text features. HTML tag features are essential for marking sentence boundaries in web text, as many boundaries in web text are only indicated by HTML tags and not by the text. Our system using HTML tag features achieves 95.1% accuracy in disambiguating sentence boundaries in web text compared to 88.9% without using such features.

Third, tokenisation is accomplished using the *sed* script used for the Penn Treebank project (MacIntyre, 1995), modified to correctly tokenise URLs, emails, and other web-specific text.

The final step is filtering, where unwanted text is removed from the corpus. A rule-based component analyses each web page and each sentence within a page to identify sections that are unlikely to be useful text. Our rules are similar to those employed by Halacsy et al. (2004), where the percentage of non-dictionary words in a sentence or document helps identify non-Hungarian text. We classify tokens into dictionary words, word-like tokens, numbers, punctuation, and other tokens. Sentences or documents with too few dictionary words or too many numbers, punctuation, or other tokens are discarded.

4 Corpus Statistics

Comparing the vocabulary of the Web Corpus and existing corpora is revealing. We compared with the Gigaword Corpus, a 2 billion token collection (1.75 billion words before tokenisation) of newspaper text (Graff, 2003). For example, what types of tokens appears more frequently on the web than in newspaper text? From each corpus, we randomly select a 1 billion word sample and classified the tokens into seven disjoint categories:

Numeric – At least one digit and zero or more punctuation characters, e.g. 2, 3.14, \$5.50

Uppercase – Only uppercase, e.g. REUTERS

Title Case – An uppercase letter followed by one or more lowercase letters, e.g. Dilbert

Lowercase – Only lowercase, e.g. violin

Alphanumeric – At least one alphabetic and one digit (allowing for other characters), e.g. B2B, mp3, RedHat-9

Hyphenated Word – Alphabetic characters and hyphens, e.g. serb-dominated, vis-a-vis

Other – Any other tokens

4.1 Token Type Analysis

An analysis by *token type* shows big differences between the two corpora (see Table 1). The same size samples of the Gigaword and the Web Corpus have very different number of token types. Title case tokens is a significant percentage of the token types encountered in both corpora, possibly representing named-entities in the text. There are also a significant number of tokens classified as others in the Web Corpus, possibly representing URLs and email addresses. While 2.2 million token types are found in the 1 billion word sample of the Gigaword, about twice as many (4.8 million) are found in an equivalent sample of the Web Corpus.

	Gigaword		Web Corpus	
Tokens	1 billion		1 billion	
Token Types	2.2 million		4.8 million	
Numeric	343k	15.6%	374k	7.7%
Uppercase	95k	4.3%	241k	5.0%
Title Case	645k	29.3%	946k	19.6%
Lowercase	263k	12.0%	734k	15.2%
Alpha-numeric	165k	7.6%	417k	8.6%
Hyphenated	533k	24.3%	970k	20.1%
Other	150k	6.8%	1,146k	23.7%

Table 1: Classification of corpus token by type

Gigaword	Web Corpus	
rreceive	receive	receieve
receieve	recesive	recive
receieve	recieive	recevieve
recive	receivce	receivve
receiv	receieve	receve
	receivea	receiv
	rceive	reyceive
1.7 misspellings per dictionary word	3.7 misspellings per dictionary word	
3.1m misspellings in 699m dict. words	5.6m misspellings in 669m dict. words	

Table 2: Misspellings of receive

4.2 Misspelling

One factor contributing to the larger number of token types in the Web Corpus, as compared with the Gigaword, is the misspelling of words. Web documents are authored by people with a widely varying command of English and their pages are not as carefully edited as newspaper articles. Thus, we anticipate a significantly larger number of misspellings and typographical errors.

We identify some of the misspellings by letter combinations that are one transformation away from a correctly spelled word. Consider a target word, correctly spelled. Misspellings can be generated by inserting, deleting, or substituting one letter, or by reordering any two adjacent letters (although we keep the first letter of the original word, as very few misspellings change the first letter).

Table 2 shows some of the misspellings of the word receive found in the Gigaword and the Web Corpus. While only 5 such misspellings were found in the Gigaword, 16 were found in the Web

Algorithm	Training	Testing	AA	WAA
Unpruned Winnow	Brown 80%	Brown 20%	94.1	96.4
Unpruned Winnow	Brown 80%	WSJ 40%	89.5	94.5
Winnow Semi-Sup.	Brown 80%*	WSJ 40%	93.1	96.6
Search Engine	Altavista	Brown 100%	89.3	N/A

Table 3: Context-sensitive spelling correction (* denotes also using 60% WSJ, 5% corrupted)

Corpus. For all words found in the Unix dictionary, an average of 1.7 misspellings are found per word in the Gigaword by type. The proportion of mistakes found in the Web Corpus is roughly double that of the Gigaword, at 3.7 misspellings per dictionary word. However, misspellings only represent a small portion of tokens (5.6 million out of 699 million instances of dictionary word are misspellings in the Web Corpus).

5 Context-Sensitive Spelling Correction

A confusion set is a collection of words which are commonly misused by even native speakers of a language because of their similarity. For example, the words {it's, its}, {affect, effect}, and {weather, whether} are often mistakenly interchanged. *Context-sensitive spelling correction* is the task of selecting the correct confusion word in a given context. Two different metrics have been used to evaluate the performance of context-sensitive spelling correction algorithms. The Average Accuracy (AA) is the performance by type whereas the Weighted Average Accuracy (WAA) is the performance by token.

5.1 Related Work

Golding and Roth (1999) used the Winnow multiplicative weight-updating algorithm for context-sensitive spelling correction. They found that when a system is tested on text from a different from the training set the performance drops substantially (see Table 3). Using the same algorithm and 80% of the Brown Corpus, the WAA dropped from 96.4% to 94.5% when tested on 40% WSJ instead of 20% Brown.

For cross corpus experiments, Golding and Roth devised a semi-supervised algorithm that is

trained on a fixed training set but also extracts information from the same corpus as the testing set. Their experiments showed that even if up to 20% of the testing set is corrupted (using wrong confusion words), a system that trained on both the training and testing sets outperformed the system that only trained on the training set. The Winnow Semi-Supervised method increases the WAA back up to 96.6%.

Lapata and Keller (2005) utilised web counts from Altavista for confusion set disambiguation. Their unsupervised method uses collocation features (one word to the left and right) where co-occurrence estimates are obtained from web counts of bigrams. This method achieves a stated accuracy of 89.3% AA, similar to the cross corpus experiment for Unpruned Winnow.

5.2 Implementation

Context-sensitive spelling correction is an ideal task for unannotated web data as unmarked text is essentially labelled data for this particular task, as words in a reasonably well-written text are positive examples of the correct usage of confusion words.

To demonstrate the utility of a large collection of web data on a disambiguation problem, we implemented the simple memory-based learner from Banko and Brill (2001). The learner trains on simple collocation features, keeping a count of (w_{i-1}, w_{i+1}) , w_{i-1} , and w_{i+1} for each confusion word w_i . The classifier first chooses the confusion word which appears with the context bigram most frequently, followed by the left unigram, right unigram, and then the most frequent confusion word.

Three data sets were used in the experiments: the 2 billion word Gigaword Corpus, a 2 billion word sample of our 10 billion word Web Corpus, and the full 10 billion word Web Corpus.

5.3 Results

Our experiments compare the results when the three corpora were trained using the same algorithm. The memory-based learner was tested using the 18 confusion word sets from Golding (1995) on the WSJ section of the Penn Treebank and the Brown Corpus.

For the WSJ testing set, the 2 billion word Web Corpus does not achieve the performance of the Gigaword (see Table 4). However, the 10 billion word Web Corpus results approach that of the Gigaword. Training on the Gigaword and testing

Training	Testing	AA	WAA
Gigaword 2 billion	WSJ 100%	93.7	96.1
Web Corpus 2 billion	WSJ 100%	92.7	94.1
Web Corpus 10 billion	WSJ 100%	93.3	95.1
Gigaword 2 billion	Brown 100%	90.7	94.6
Web Corpus 2 billion	Brown 100%	90.8	94.8
Web Corpus 10 billion	Brown 100%	91.8	95.4

Table 4: Memory-based learner results

on WSJ is not considered a true cross-corpus experiment, as the two corpora belong to the same genre of newspaper text. Compared to the Window method, the 10 billion word Web Corpus outperforms the cross corpus experiment but not the semi-supervised method.

For the Brown Corpus testing set, the 2 billion word Web Corpus and the 2 billion word Gigaword achieved similar results. The 10 billion word Web Corpus achieved 95.4% WAA, higher than the 94.6% from the 2 billion Gigaword. This and the above result with the WSJ suggests that the Web Corpus approach is comparable with training on a corpus of printed text such as the Gigaword.

The 91.8% AA of the 10 billion word Web Corpus testing on the WSJ is better than the 89.3% AA achieved by Lapata and Keller (2005) using the Altavista search engine. This suggests that a web collected corpus may be a more accurate method of estimating n-gram frequencies than through search engine hit counts.

6 Thesaurus Extraction

Thesaurus extraction is a word similarity task. It is a natural candidate for using web corpora as most systems extract synonyms of a target word from an unlabelled corpus. Automatic thesaurus extraction is a good alternative to manual construction methods, as such thesauri can be updated more easily and quickly. They do not suffer from bias, low coverage, and inconsistency that human creators of thesauri introduce.

Thesauri are useful in many NLP and Information Retrieval (IR) applications. Synonyms help

expand the coverage of system but providing alternatives to the input search terms. For n-gram estimation using search engine queries, some NLP applications can boost the hit count by offering alternative combination of terms. This is especially helpful if the initial hit counts are too low to be reliable. In IR applications, synonyms of search terms help identify more relevant documents.

6.1 Method

We use the thesaurus extraction system implemented in Curran (2004). It operates on the *distributional hypothesis* that *similar words appear in similar contexts*. This system only extracts one word synonyms of nouns (and not multi-word expressions or synonyms of other parts of speech). The extraction process is divided into two parts. First, target nouns and their surrounding contexts are encoded in relation pairs. Six different types of relationships are considered:

- Between a noun and a modifying adjective
- Between a noun and a noun modifier
- Between a verb and its subject
- Between a verb and its direct object
- Between a verb and its indirect object
- Between a noun and the head of a modifying prepositional phrase

The nouns (including subject and objects) are the target headwords and the relationships are represented in *context vectors*. In the second stage of the extraction process, a comparison is made between context vectors of headwords in the corpus to determine the most similar terms.

6.2 Evaluation

The evaluation of a list of synonyms of a target word is subject to human judgement. We use the evaluation method of Curran (2004), against gold standard thesauri results. The gold standard list is created by combining the terms found in four thesauri: Macquarie, Moby, Oxford and Roget's.

The *inverse rank* (InvR) metric allows a comparison to be made between the extracted rank list of synonyms and the unranked gold standard list. For example, if the extracted terms at ranks 3, 5, and 28 are found in the gold standard list, then $InvR = \frac{1}{3} + \frac{1}{5} + \frac{1}{28} \cong 0.569$.

Corpus	INVR	INVR MAX
Gigaword	1.86	5.92
Web Corpus	1.81	5.92

Table 5: Average INVR for 300 headwords

	Word	INVR Scores		Diff.
1	picture	3.322	to 0.568	2.754
2	star	2.380	to 0.119	2.261
3	program	3.218	to 1.184	2.034
4	aristocrat	2.056	to 0.031	2.025
5	box	3.194	to 1.265	1.929
6	cent	2.389	to 0.503	1.886
7	home	2.306	to 0.523	1.783
⋮	⋮	⋮	⋮	⋮
296	game	1.097	to 2.799	-1.702
297	bloke	0.425	to 2.445	-2.020
298	point	1.477	to 3.540	-2.063
299	walk	0.774	to 3.184	-2.410
300	chain	0.224	to 3.139	-2.915

Table 6: INVR scores ranked by difference, Gigaword to Web Corpus

Gigaword (24 matches out of 200)
house apartment building run office resident residence headquarters victory native place mansion room trip mile family night hometown town win neighborhood life suburb school restaurant hotel store city street season area road homer day car shop hospital friend game farm facility center north child land weekend community loss return hour ...
Web Corpus (18 matches out of 200)
page loan contact house us owner search finance mortgage office map links building faq equity news center estate privacy community info business car site web improvement extension heating rate directory room apartment family service rental credit shop life city school property place location job online vacation store facility library free ...

Table 7: Synonyms for home

Gigaword (9 matches out of 200)
store retailer supermarket restaurant outlet operator shop shelf owner grocery company hotel manufacturer retail franchise clerk maker discount business sale superstore brand clothing food giant shopping firm retailing industry drugstore distributor supplier bar insurer inc. conglomerate network unit apparel boutique mall electronics carrier division brokerage toy producer pharmacy airline inc ...
Web Corpus (53 matches out of 200)
necklace supply bracelet pendant rope belt ring ear-ring gold bead silver pin wire cord reaction clasp jewelry charm frame bangle strap sterling loop timing plate metal collar turn hook arm length string retailer repair brooch plug diamond wheel industry tube surface neck brooch store molecule ribbon pump choker shaft body ...

Table 8: Synonyms for chain

6.3 Results

We used the same 300 evaluation headwords as Curran (2004) and extracted the top 200 synonyms for each headword. The evaluation headwords were extracted from two corpora for comparison – a 2 billion word sample of our Web Corpus and the 2 billion words in the Gigaword Corpus. Table 5 shows the average INVR scores over the 300 headwords for the two corpora – one of web text and the other newspaper text. The INVR values differ by a negligible 0.05 (out of a maximum of 5.92).

6.4 Analysis

However on a per word basis one corpus can significantly outperform the other. Table 6 ranks the 300 headwords by difference in the INVR score. While much better results were extracted for words like home from the Gigaword, much better results were extracted for words like chain from the Web Corpus.

Table 7 shows the top 50 synonyms extracted for the headword home from the Gigaword and the Web Corpus. While similar number of correct synonyms were extracted from both corpora, the Gigaword matches were higher in the extracted list and received a much higher INVR score. In the list extracted from the Web Corpus, web-related collocations such as home page and search home appear.

Table 8 shows the top 50 synonyms extracted for the headword chain from both corpora. While there are only a total of 9 matches from the Gigaword Corpus, there are 53 matches from the Web Corpus. A closer examination shows that the synonyms extracted from the Gigaword belong only to one sense of the word chain, as in chain stores. The gold standard list and the Web Corpus results both contain the necklace sense of the word chain. The Gigaword results show a skew towards the business sense of the word chain, while the Web Corpus covers both senses of the word.

While individual words can achieve better results in either the Gigaword or the Web Corpus than the other, the aggregate results of synonym extraction for the 300 headwords are the same. For this task, the Web Corpus can replace the Gigaword without affecting the overall result. However, as some words are performed better under different corpora, an aggregate of the Web Corpus and the Gigaword may produce the best result.

7 Conclusion

In this paper, the accuracy of natural language application training on a 10 billion word Web Corpus is compared with other methods using search engine hit counts and corpora of printed text.

In the context-sensitive spelling correction task, a simple memory-based learner trained on our Web Corpus achieved better results than method based on search engine queries. It also rival some of the state-of-the-art systems, exceeding the accuracy of the Unpruned Winnow method (the only other true cross-corpus experiment). In the task of thesaurus extraction, the same overall results are obtained extracting from the Web Corpus as a traditional corpus of printed texts.

The Web Corpus contrasts with other NLP approaches that access web data through search engine queries. Although the 10 billion words Web Corpus is smaller than the number of words indexed by search engines, better results have been achieved using the smaller collection. This is due to the more accurate n-gram counts in the downloaded text. Other NLP tasks that require further analysis of the downloaded text, such a PP attachment, may benefit more from the Web Corpus.

We have demonstrated that carefully collected and filtered web corpora can be as useful as newswire corpora of equivalent sizes. Using the same framework describe here, it is possible to collect a much larger corpus of freely available web text than our 10 billion word corpus. As NLP algorithms tend to perform better when more data is available, we expect state-of-the-art results for many tasks will come from exploiting web text.

Acknowledgements

We like to thank our anonymous reviewers and the Language Technology Research Group at the University of Sydney for their comments. This work has been supported by the Australian Research Council under Discovery Project DP0453131.

References

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the ACL*, pages 26–33, Toulouse, France.

Charles L.A. Clarke, Gordon V. Cormack, M. Laszlo, Thomas R. Lynam, and Egidio Terra. 2002. The impact of corpus size on question answering performance. In *Proceedings of the ACM SIGIR*, pages 369–370, Tampere, Finland.

James Curran. 2004. *From Distributional to Semantic Similarity*. PhD thesis, University of Edinburgh, UK.

Andrew R. Golding and Dan Roth. 1999. A winnow-based approach to context-sensitive spelling correction. *Machine Learning*, 34(1-3):107–130.

Andrew R. Golding. 1995. A bayesian hybrid method for context-sensitive spelling correction. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 39–53, Somerset, NJ USA. ACL.

David Graff. 2003. English Gigaword. Technical Report LDC2003T05, Linguistic Data Consortium, Philadelphia, PA USA.

Gregory Grefenstette. 1999. The WWW as a resource for example-based MT tasks. In *the ASLIB Translating and the Computer Conference*, London, UK, October.

Peter Halacsy, Andras Kornai, Laszlo Nemeth, Andras Rung, Istvan Szakadat, and Vikto Tron. 2004. Creating open language resources for Hungarian. In *Proceedings of the LREC*, Lisbon, Portugal.

M. R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. 2000. On near-uniform URL sampling. In *Proceedings of the 9th International World Wide Web Conference*.

Frank Keller and Mirella Lapata. 2003. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3):459–484.

Mirella Lapata and Frank Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*.

Steve Lawrence and C. Lee Giles. 1999. Accessibility of information on the web. *Nature*, 400:107–109.

Robert MacIntyre. 1995. Sed script to produce Penn Treebank tokenization on arbitrary raw text. From <http://www.cis.upenn.edu/treebank/tokenizer.sed>.

Natalia N. Modjeska, Katja Markert, and Malvina Nissim. 2003. Using the web in machine learning for other-anaphora resolution. In *Proceedings of the EMNLP*, pages 176–183, Sapporo, Japan.

Preslav Nakov and Marti Hearst. 2005. A study of using search engine page hits as a proxy for n-gram frequencies. In *Recent Advances in Natural Language Processing*.

Adwait Ratnaparkhi. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD thesis, University of Pennsylvania, Philadelphia, PA USA.

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and NLP: Using locality sensitive hash functions for high speed noun clustering. In *Proceedings of the ACL*, pages 622–629.

E. L. Terra and Charles L.A. Clarke. 2003. Frequency estimates for statistical word similarity measures. In *Proceedings of the HLT*, Edmonton, Canada.

Martin Volk. 2001. Exploiting the WWW as a corpus to resolve PP attachment ambiguities. In *Proceedings of the Corpus Linguistics 2001*, Lancaster, UK, March.

Martin Volk. 2002. Using the web as corpus for linguistic research. *Tähendusepüüja. Catcher of the Meaning. A Festschrift for Professor Haldur Õim*.