Exploiting Web-Derived Selectional Preference to Improve Statistical Dependency Parsing

Guangyou Zhou, Jun Zhao*, Kang Liu, and Li Cai
National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, China
{gyzhou,jzhao,kliu,lcai}@nlpr.ia.ac.cn

Abstract

In this paper, we present a novel approach which incorporates the web-derived selectional preferences to improve statistical dependency parsing. Conventional selectional preference learning methods have usually focused on word-to-class relations, e.g., a verb selects as its subject a given nominal class. This paper extends previous work to word-to-word selectional preferences by using web-scale data. Experiments show that web-scale data improves statistical dependency parsing, particularly for long dependency relationships. There is no data like more data, performance improves log-linearly with the number of parameters (unique N-grams). More importantly, when operating on new domains, we show that using web-derived selectional preferences is essential for achieving robust performance.

1 Introduction

Dependency parsing is the task of building dependency links between words in a sentence, which has recently gained a wide interest in the natural language processing community. With the availability of large-scale annotated corpora such as Penn Treebank (Marcus et al., 1993), it is easy to train a high-performance dependency parser using supervised learning methods.

However, current state-of-the-art statistical dependency parsers (McDonald et al., 2005; McDonald and Pereira, 2006; Hall et al., 2006) tend to have lower accuracies for longer dependencies (McDonald and Nivre, 2007). The length of a dependency from word w_i to word w_j is simply equal to $|i-j|$. Longer dependencies typically represent the modifier of the root or the main verb, internal dependencies of longer NPs or PP-attachment in a sentence. Figure 1 shows the F_1 score[1] relative to the dependency length on the development set by using the graph-based dependency parsers (McDonald et al., 2005; McDonald and Pereira, 2006). We note that the parsers provide very good results for adjacent dependencies (96.89% for dependency length =1), while the dependency length increases, the accuracies degrade sharply. These longer dependencies are therefore a major opportunity to improve the overall performance of dependency parsing. Usually, these longer dependencies can be parsed dependent on the specific words involved due to the limited range of features (e.g., a verb and its modifiers). Lexical statistics are therefore needed for resolving ambiguous relationships, yet the lexicalized statistics are sparse and difficult to estimate directly. To solve this problem, some information with different granularity has been investigated. Koo et al. (2008) proposed a semi-supervised dependency parsing by introducing lexical intermediaries at a coarser level than words themselves via a cluster method. This approach, however, ignores the selectional preference for word-to-word interactions, such as head-modifier relationship. Extra resources

*Correspondence author: jzhao@nlpr.ia.ac.cn

[1] Precision represents the percentage of predicted arcs of length d that are correct, and recall measures the percentage of gold-standard arcs of length d that are correctly predicted. $F_1 = 2 \times precision \times recall/(precision + recall)$

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 1556–1565,
Portland, Oregon, June 19-24, 2011.

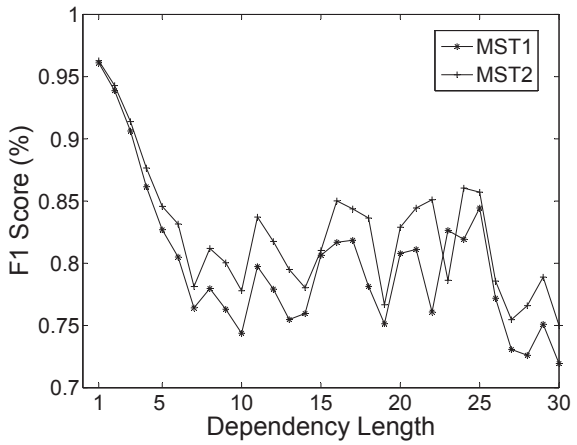

Figure 1: F score relative to dependency length.

beyond the annotated corpora are needed to capture the bi-lexical relationship at the word-to-word level.

Our purpose in this paper is to exploit web-derived selectional preferences to improve the supervised statistical dependency parsing. All of our lexical statistics are derived from two kinds of web-scale corpus: one is the web, which is the largest data set that is available for NLP (Keller and Lapata, 2003). Another is a web-scale N-gram corpus, which is a N-gram corpus with N-grams of length 1-5 (Brants and Franz, 2006), we call it **Google V1** in this paper. The idea is very simple: web-scale data have large coverage for word pair acquisition. By leveraging some assistant data, the dependency parsing model can directly utilize the additional information to capture the word-to-word level relationships. We address two natural and related questions which some previous studies leave open:

Question I: Is there a benefit in incorporating web-derived selectional preference features for statistical dependency parsing, especially for longer dependencies?

Question II: How well do web-derived selectional preferences perform on new domains?

For Question I, we systematically assess the value of using web-scale data in state-of-the-art supervised dependency parsers. We compare dependency parsers that include or exclude selectional preference features obtained from web-scale corpus. To the best of our knowledge, none of the existing studies directly address long dependencies of dependency parsing by using web-scale data.

Most statistical parsers are highly domain dependent. For example, the parsers trained on WSJ text perform poorly on Brown corpus. Some studies have investigated domain adaptation for parsers (McClosky et al., 2006; Daumé III, 2007; McClosky et al., 2010). These approaches assume that the parsers know which domain it is used, and that it has access to representative data in that domain. However, in practice, these assumptions are unrealistic in many real applications, such as when processing the heterogeneous genre of web texts. In this paper we incorporate the web-derived selectional preference features to design our parsers for robust open-domain testing.

We conduct the experiments on the English Penn Treebank (PTB) (Marcus et al., 1993). The results show that web-derived selectional preference can improve the statistical dependency parsing, particularly for long dependency relationships. More importantly, when operating on new domains, the web-derived selectional preference features show great potential for achieving robust performance (Section 4.3).

The remainder of this paper is divided as follows. Section 2 gives a brief introduction of dependency parsing. Section 3 describes the web-derived selectional preference features. Experimental evaluation and results are reported in Section 4. Finally, we discuss related work and draw conclusion in Section 5 and Section 6, respectively.

2 Dependency Parsing

In dependency parsing, we attempt to build head-modifier (or head-dependent) relations between words in a sentence. The discriminative parser we used in this paper is based on the *part-factored* model and features of the MSTParser (McDonald et al., 2005; McDonald and Pereira, 2006; Carreras, 2007). The parsing model can be defined as a conditional distribution $p(y|\mathbf{x};\mathbf{w})$ over each projective parse tree y for a particular sentence $\mathbf{x}$, parameterized by a vector $\mathbf{w}$. The probability of a parse tree is

$$p(y|\mathbf{x};\mathbf{w}) = \frac{1}{Z(\mathbf{x};\mathbf{w})}\exp\Big\{\sum_{\rho\in y}\mathbf{w}\cdot\mathbf{\Phi}(\mathbf{x},\rho)\Big\} \quad (1)$$

where $Z(\mathbf{x};\mathbf{w})$ is the partition function and $\mathbf{\Phi}$ are *part-factored* feature functions that include *head-*

modifier parts, *sibling* parts and *grandchild* parts. Given the training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, parameter estimation for log-linear models generally resolve around optimization of a regularized conditional log-likelihood objective $\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{w})$ where

$$L(\mathbf{w}) = -C \sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i; \mathbf{w}) + \frac{1}{2}||\mathbf{w}||^2 \quad (2)$$

The parameter $C > 0$ is a constant dictating the level of regularization in the model. Since objective function $L(\mathbf{w})$ is smooth and convex, which is convenient for standard gradient-based optimization techniques. In this paper we use the dual exponentiated gradient (EG)[2] descent, which is a particularly effective optimization algorithm for log-linear models (Collins et al., 2008).

3 Web-Derived Selectional Preference Features

In this paper, we employ two different feature sets: a baseline feature set[3] which draw upon "normal" information source, such as word forms and part-of-speech (POS) without including the web-derived selectional preference[4] features, a feature set conjoins the baseline features and the web-derived selectional preference features.

3.1 Web-scale resources

All of our selectional preference features described in this paper rely on probabilities derived from unlabeled data. To use the largest amount of data possible, we exploit web-scale resources. one is web, N-gram counts are approximated by **Google hits**. Another we use is **Google V1** (Brants and Franz, 2006). This N-gram corpus records how often each unique sequence of words occurs. N-grams appearing 40 times or more (1 in 25 billion) are kept, and appear in the n-gram tables. All n-grams with lower counts are discarded. Co-occurrence probabilities can be calculated directly from the N-gram counts.

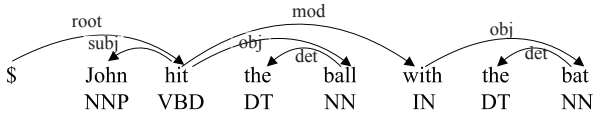

Figure 2: An example of a labeled dependency tree. The tree contains a special token "$" which is always the root of the tree. Each arc is directed from head to modifier and has a label describing the function of the attachment.

3.2 Web-derived N-gram features

3.2.1 PMI

Previous work on noun compounds bracketing has used *adjacency model* (Resnik, 1993) and *dependency model* (Lauer, 1995) to compute association statistics between pairs of words. In this paper we generalize the adjacency and dependency models by including the pointwise mutual information (Church and Hanks, 1900) between all pairs of words in the dependency tree:

$$\text{PMI}(x, y) = \log \frac{p(\text{“x y”})}{p(\text{“x”})p(\text{“y”})} \quad (3)$$

where $p(\text{“x y”})$ is the co-occurrence probabilities. When use the **Google V1** corpus, this probabilities can be calculated directly from the N-gram counts, while using the **Google hits**, we send the queries to the search engine *Google*[5] and all the search queries are performed as exact matches by using quotation marks.[6]

The value of these features is the PMI, if it is defined. If the PMI is undefined, following the work of (Pitler et al., 2010), we include one of two binary features:

$$p(\text{“x y”}) = 0 \text{ or } p(\text{“x”}) \vee p(\text{“y”}) = 0$$

Besides, we also consider the trigram features be-

[2] http://groups.csail.mit.edu/nlp/egstra/

[3] This kind of feature sets are similar to other feature sets in the literature (McDonald et al., 2005; Carreras, 2007), so we will not attempt to give a exhaustive description.

[4] Selectional preference tells us which arguments are plausible for a particular predicate, one way to determine the selectional preference is from co-occurrences of predicates and arguments in text (Bergsma et al., 2008). In this paper, the selectional preferences have the same meaning with N-grams, which model the word-to-word relationships, rather than only considering the predicates and arguments relationships.

[5] http://www.google.com/

[6] Google only allows automated querying through the Google Web API, this involves obtaining a license key, which then restricts the number of queries to a daily quota of 1000. However, we obtained a quota of 20,000 queries per day by sending a request to api-support@google.com for research purposes.

PMI("hit with")
x_i-word="hit", x_j-word="with", PMI("hit with")
x_i-word="hit", x_j-word="with", x_j-pos="IN", PMI("hit with")
x_i-word="hit", x_i-pos="VBD", x_j-word="with", PMI("hit with")
x_i-word="hit", b-pos="ball", x_j-word="with", PMI("hit with")
x_i-word="hit", x_j-word="with", PMI("hit with"), dir=R, dist=3
...

Table 1: An example of the N-gram PMI features and the conjoin features with the baseline.

tween the three words in the dependency tree:

$$\text{PMI}(x, y, z) = \log \frac{p(\text{"x y z"})}{p(\text{"x y"})p(\text{"y z"})} \quad (4)$$

This kinds of trigram features, for example in MSTParser, which can directly capture the sibling and grandchild features.

We illustrate the PMI features with an example of dependency parsing tree in Figure 2. In deciding the dependency between the main verb *hit* and its argument headed preposition *with*, an example of the N-gram PMI features and the conjoin features with the baseline are shown in Table 1.

3.2.2 PP-attachment

Propositional phrase (PP) attachment is one of the hardest problems in English dependency parsing. An English sentence consisting of a subject, a verb, and a nominal object followed by a prepositional phrase is often ambiguous. Ambiguity resolution reflects the selectional preference between the verb and noun with their prepositional phrase. For example, considering the following two examples:

(1) John **hit** the ball **with** the *bat*.

(2) John hit the **ball with** the red *stripe*.

In sentence (1), the preposition **with** depends on the main verb **hit**; but in sentence (2), the prepositional phrase is a noun attribute and the preposition **with** needs to depends on the word **ball**. To resolve this kind of ambiguity, there needs to measure the attachment preference. We thus have PP-attachment features that determine the PMI association across the preposition word "*IN*"[7]:

$$\text{PMI}_{IN}(x, z) = \log \frac{p(\text{"x } IN \text{ z"})}{p(\text{x})} \quad (5)$$

[7]Here, the preposition word "*IN*" (e.g., "with", "in", ...) is any token whose part-of-speech is IN

N-gram feature templates
hw, mw, PMI(hw,mw)
hw, ht, mw, PMI(hw,mw)
hw, mw, mt, PMI(hw,mw)
hw, ht, mw, mt, PMI(hw,mw)
...
hw, mw, sw
hw, mw, sw, PMI(hw, mw, sw)
hw, mw, gw
hw, mw, gw, PMI(hw, mw, gw)

Table 2: Examples of N-gram feature templates. Each entry represents a class of indicator for tuples of information. For example, "hw, mw" reprsents a class of indicator features with one feature for each possible combination of head word and modifier word. Abbreviations: hw=head word, ht= head POS. st, gt=likewise for sibling and grandchild.

$$\text{PMI}_{IN}(y, z) = \log \frac{p(\text{"y } IN \text{ z"})}{p(\text{y})} \quad (6)$$

where the word x and y are usually verb and noun, z is a noun which directly depends on the preposition word "*IN*". For example in sentence (1), we would include the features $\text{PMI}_{with}(hit, bat)$ and $\text{PMI}_{with}(ball, bat)$. If both PMI features exist and $\text{PMI}_{with}(hit, bat) > \text{PMI}_{with}(ball, bat)$, indicating to our dependency parsing model that the preposition word *with* depends on the verb *hit* is a good choice. While in sentence (2), the features include $\text{PMI}_{with}(hit, stripe)$ and $\text{PMI}_{with}(ball, stripe)$.

3.3 N-gram feature templates

We generate N-gram features by mimicking the template structure of the original baseline features. For example, the baseline feature set includes indicators for word-to-word and tag-to-tag interactions between the head and modifier of a dependency. In the N-gram feature set, we correspondingly introduce N-gram PMI for word-to-word interactions.

The N-gram feature set for MSTParser is shown in Table 2. Following McDonald et al. (2005), all features are conjoined with the direction of attachment as well as the distance between the two words creating the dependency. In between N-gram features, we include the form of word trigrams and PMI of the trigrams. The surrounding word N-gram features represent the local context of the selectional preference. Besides, we also present the second-order feature templates, including the sibling and grandchild features. These features are designed to disambiguate cases like coordinating conjunctions and prepositional attachment. Consider the examples we have shown in section 3.2.2, for sentence (1), the dependency graph path feature $ball \rightarrow with \rightarrow bat$ should have a lower weight since *ball* rarely is modified by *bat*, but is often seen through them (e.g., a higher weight should be associated with $hit \rightarrow with \rightarrow bat$). In contrast, for sentence (2), our N-gram features will tell us that the prepositional phrase is much more likely to attach to the noun since the dependency graph path feature $ball \rightarrow with \rightarrow stripe$ should have a high weight due to the high strength of selectional preference between *ball* and *stripe*.

Web-derived selectional preference features based on PMI values are trickier to incorporate into the dependency parsing model because they are continuous rather than discrete. Since all the baseline features used in the literature (McDonald et al., 2005; Carreras, 2007) take on binary values of 0 or 1, there is a "mis-match" between the continuous and binary features. Log-linear dependency parsing model is sensitive to inappropriately scaled feature. To solve this problem, we transform the PMI values into a more amenable form by replacing the PMI values with their $z\text{-}score$. The $z\text{-}score$ of a PMI value x is $\frac{x-\mu}{\sigma}$, where μ and σ are the mean and standard deviation of the PMI distribution, respectively.

4 Experiments

In order to evaluate the effectiveness of our proposed approach, we conducted dependency parsing experiments in English. The experiments were performed on the Penn Treebank (PTB) (Marcus et al., 1993), using a standard set of head-selection rules (Yamada and Matsumoto, 2003) to convert the phrase structure syntax of the Treebank into a dependency tree representation, dependency labels were obtained via the "Malt" hard-coded setting.[8] We split the Treebank into a training set (Sections 2-21), a development set (Section 22), and several test sets (Sections 0,[9] 1, 23, and 24). The part-of-speech tags for the development and test set were automatically assigned by the MXPOST tagger[10], where the tagger was trained on the entire training corpus.

Web page hits for word pairs and trigrams are obtained using a simple heuristic query to the search engine *Google*.[11] Inflected queries are performed by expanding a bigram or trigram into all its morphological forms. These forms are then submitted as literal queries, and the resulting hits are summed up. John Carroll's suite of morphological tools[12] is used to generate inflected forms of verbs and nouns. All the search terms are performed as exact matches by using quotation marks and submitted to the search engines in lower case.

We measured the performance of the parsers using the following metrics: unlabeled attachment score (UAS), labeled attachment score (LAS) and complete match (CM), which were defined by Hall et al. (2006). All the metrics are calculated as mean scores per word, and punctuation tokens are consistently excluded.

4.1 Main results

There are some clear trends in the results of Table 3. First, performance increases with the order of the parser: *edge-factored* model (dep1) has the lowest performance, adding sibling and grandchild relationships (dep2) significantly increases performance. Similar observations regarding the effect of model order have also been made by Carreras (2007) and Koo et al. (2008).

Second, note that the parsers incorporating the N-gram feature sets consistently outperform the models using the baseline features in all test data sets, regardless of model order or label usage. Another

[8] http://w3.msi.vxu.se/ nivre/research/MaltXML.html

[9] We removed a single 249-word sentence from Section 0 for computational reasons.

[10] http://www.inf.ed.ac.uk/resources/nlp/local_doc/MXPOST.html

[11] http://www.google.com/

[12] http://www.cogs.susx.ac.uk/lab/nlp/carroll/morph.html.

Sec	dep1	+hits	+V1	dep2	+hits	+V1	dep1-L	+hits-L	+V1-L	dep2-L	+hits-L	+V1-L
00	90.39	**90.94**	90.91	91.56	**92.16**	**92.16**	90.11	**90.69**	90.67	91.94	**92.47**	92.42
01	91.01	**91.60**	**91.60**	92.27	**92.89**	92.86	90.77	**91.39**	**91.39**	91.81	**92.38**	92.37
23	90.82	**91.46**	91.39	91.98	**92.64**	92.59	90.30	**90.98**	90.92	91.24	**91.83**	91.77
24	89.53	**90.15**	90.13	90.81	**91.44**	91.41	89.42	**90.03**	90.02	90.30	**90.91**	90.89

Table 3: Unlabeled accuracies (UAS) and labeled accuracies (LAS) on Section 0, 1, 23, 24. Abbreviation: dep1/dep2=first-order parser and second-order parser with the baseline features; +hits=N-gram features derived from the Google hits; +V1=N-gram features derived from the Google V1; suffix-L=labeled parser. Unlabeled parsers are scored using unlabeled parent predictions, and labeled parsers are scored using labeled parent predictions.

finding is that the N-gram features derived from Google hits are slightly better than Google V1 due to the large N-gram coverage, we will discuss later. As a final note, all the comparisons between the integration of N-gram features and the baseline features in Table 3 are mildly significant using the Z-test of Collins et al. (2005) ($p < 0.08$).

Type	Systems	UAS	CM
D	Yamada and Matsumoto (2003)	90.3	38.7
	McDonald et al. (2005)	90.9	37.5
	McDonald and Pereira (2006)	91.5	42.1
	Corston-Oliver et al. (2006)	90.9	37.5
	Hall et al. (2006)	89.4	36.4
	Wang et al. (2007)	89.2	34.4
	Carreras et al. (2008)	**93.5**	-
	GoldBerg and Elhadad (2010)†	91.32	40.41
	Ours	*92.64*	46.61
C	Nivre and McDonald (2008)†	92.12	44.37
	Martins et al. (2008)†	**92.87**	45.51
	Zhang and Clark (2008)	92.1	45.4
S	Koo et al. (2008)	93.16	-
	Suzuki et al. (2009)	**93.79**	-
	Chen et al. (2009)	93.16	47.15

Table 4: Comparison of our final results with other best-performing systems on the whole Section 23. Type D, C and S denote discriminative, combined and semi-supervised systems, respectively. † These papers were not directly reported the results on this data set, we implemented the experiments in this paper.

To put our results in perspective, we also compare them with other best-performing systems in Table 4. To facilitate comparisons with previous work, we only use Section 23 as the test data. The results show that our second order model incorporating the N-gram features (92.64) performs better than most previously reported discriminative systems trained on the Treebank. Carreras et al. (2008) reported a very high accuracy using information of constituent structure of TAG grammar formalism, while in our system, we do not use such knowledge. When compared to the combined systems, our system is better than Nivre and McDonald (2008) and Zhang and Clark (2008), but a slightly worse than Martins et al. (2008). We also compare our method with the semi-supervised approaches, the semi-supervised approaches achieved very high accuracies by leveraging on large unlabeled data directly into the systems for joint learning and decoding, while in our method, we only explore the N-gram features to further improve supervised dependency parsing performance.

Table 5 shows the details of some other N-gram sources, where **NEWS**: created from a large set of news articles including the Reuters and Gigword (Graff, 2003) corpora. For a given number of unique N-gram, using any of these sources does not have significant difference in Figure 3. Google hits is the largest N-gram data and shows the best performance. The other two are smaller ones, accuracies increase linearly with the log of the number of types in the auxiliary data set. Similar observations have been made by Pitler et al. (2010). We see that the relationship between accuracy and the number of N-gram is not monotonic for Google V1. The reason may be that Google V1 does not make detailed preprocessing, containing many mistakes in the corpus. Although Google hits is noisier, it has very much larger coverage of bigrams or trigrams.

Some previous studies also found a log-linear relationship between unlabeled data (Suzuki and Isozaki, 2008; Suzuki et al., 2009; Bergsma et al., 2010; Pitler et al., 2010). We have shown that this trend continues well for dependency parsing by using web-scale data (NEWS and Google V1).

[13]Google indexes about more than 8 billion pages and each contains about 1,000 words on average.

Corpus	# of tokens	θ	# of types
NEWS	3.2B	1	3.7B
Google V1	1,024.9B	40	3.4B
Google hits[13]	8,000B	100	-

Table 5: N-gram data, with total number of words in the original corpus (in billions, B). Following (Brants and Franz, 2006; Pitler et al., 2010), we set the frequency threshold to filter the data θ, and total number of unique N-gram (types) remaining in the data.

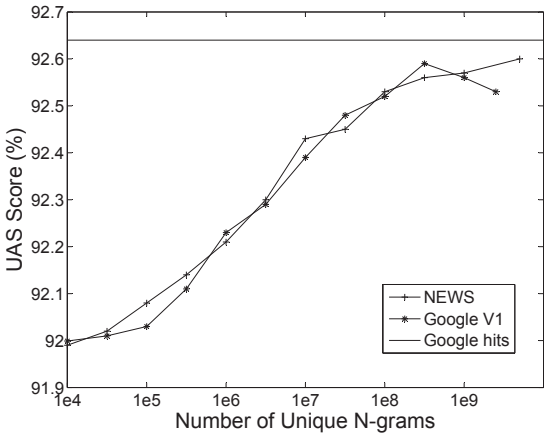

Figure 3: There is no data like more data. UAS accuracy improves with the number of unique N-grams but still lower than the Google hits.

4.2 Improvement relative to dependency length

The experiments in (McDonald and Nivre, 2007) showed a negative impact on the dependency parsing performance from too long dependencies. For our proposed approach, the improvement relative to dependency length is shown in Figure 4. From the Figure, it is seen that our method gives observable better performance when dependency lengths are larger than 3. The results here show that the proposed approach improves the dependency parsing performance, particularly for long dependency relationships.

4.3 Cross-genre testing

In this section, we present the experiments to validate the robustness the web-derived selectional preferences. The intent is to understand how well the web-derived selectional preferences transfer to other sources.

The English experiment evaluates the performance of our proposed approach when it is trained

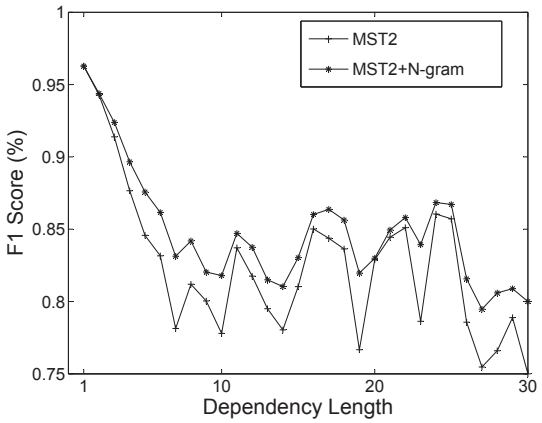

Figure 4: Dependency length vs. F_1 score.

on annotated data from one genre of text (WSJ) and is used to parse a test set from a different genre: the biomedical domain related to cancer (PennBioIE., 2005) with 2,600 parsed sentences. We divided the data into 500 for training, 100 for development and others for testing. We created five sets of training data with 100, 200, 300, 400, and 500 sentences respectively. Figure 5 plots the UAS accuracy as function of training instances. *WSJ* is the performance of our second-order dependency parser trained on section 2-21; WSJ+N-gram is the performance of our proposed approach trained on section 2-21; *WSJ+BioMed* is the performance of the parser trained on WSJ and biomedical data. *WSJ+BioMed+N-gram* is the performance of our proposed approach trained on WSJ and biomedical data. The results show that incorporating the web-scale N-gram features can significantly improve the dependency parsing performance, and the improvement is much larger than the in-domain testing presented in Section 4.1, the reason may be that web-derived N-gram features do not depend directly on training data and thus work better on new domains.

4.4 Discussion

In this paper, we present a novel method to improve dependency parsing by using web-scale data. Despite the success, there are still some problems which should be discussed.

(1) Google hits is less sparse than Google V1 in modeling the word-to-word relationships, but Google hits are likely to be noisier than Google V1. It is very appealing to carry out a correlation anal-

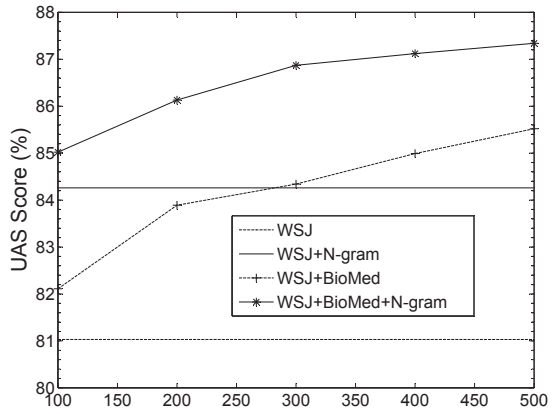

Figure 5: Adapting a WSJ parser to biomedical text. *WSJ*: performance of parser trained only on WSJ; *WSJ+N-gram*: performance of our proposed approach trained only on WSJ; *WSJ+BioMed*: parser trained on WSJ and biomedical text; *WSJ+BioMed+N-gram*: our approach trained on WSJ and biomedical text.

ysis to determine whether Google hits and Google V1 are highly correlated. We will leave it for future research.

(2) Veronis (2005) pointed out that there had been a debate about reliability of Google hits due to the inconsistencies of page hits estimates. However, this estimate is scale-invariant. Assume that when the number of pages indexed by Google grows, the number of pages containing a given search term goes to a fixed fraction. This means that if pages indexed by Google doubles, then so do the bigrams or trigrams frequencies. Therefore, the estimate becomes stable when the number of indexed pages grows unboundedly. Some details are presented in Cilibrasi and Vitanyi (2007).

5 Related Work

Our approach is to exploit web-derived selectional preferences to improve the dependency parsing. The idea of this paper is inspired by the work of Suzuki et al. (2009) and Pitler et al. (2010). The former uses the web-scale data explicitly to create more data for training the model; while the latter explores the web-scale N-grams data (Lin et al., 2010) for compound bracketing disambiguation. Our research, however, applies the web-scale data (**Google hits** and **Google V1**) to model the word-to-word dependency relationships rather than compound bracketing disambiguation.

Several previous studies have exploited the web-scale data for word pair acquisition. Keller and Lapata (2003) evaluated the utility of using web search engine statistics for unseen bigram. Nakov and Hearst (2005) demonstrated the effectiveness of using search engine statistics to improve the noun compound bracketing. Volk (2001) exploited the WWW as a corpus to resolve PP attachment ambiguities. Turney (2007) measured the semantic orientation for sentiment classification using co-occurrence statistics obtained from the search engines. Bergsma et al. (2010) created robust supervised classifiers via web-scale N-gram data for adjective ordering, spelling correction, noun compound bracketing and verb part-of-speech disambiguation. Our approach, however, extends these techniques to dependency parsing, particularly for long dependency relationships, which involves more challenging tasks than the previous work.

Besides, there are some work exploring the word-to-word co-occurrence derived from the web-scale data or a fixed size of corpus (Calvo and Gelbukh, 2004; Calvo and Gelbukh, 2006; Yates et al., 2006; Drabek and Zhou, 2000; van Noord, 2007) for PP attachment ambiguities or shallow parsing. Johnson and Riezler (2000) incorporated the lexical selectional preference features derived from British National Corpus (Graff, 2003) into a stochastic unification-based grammar. Abekawa and Okumura (2006) improved Japanese dependency parsing by using the co-occurrence information derived from the results of automatic dependency parsing of large-scale corpora. However, we explore the web-scale data for dependency parsing, the performance improves log-linearly with the number of parameters (unique N-grams). To the best of our knowledge, web-derived selectional preference has not been successfully applied to dependency parsing.

6 Conclusion

In this paper, we present a novel method which incorporates the web-derived selectional preferences to improve statistical dependency parsing. The results show that web-scale data improves the dependency parsing, particularly for long dependency relationships. There is no data like more data, performance improves log-linearly with the num-

ber of parameters (unique N-grams). More importantly, when operating on new domains, the web-derived selectional preferences show great potential for achieving robust performance.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 60875041 and No. 61070106), and CSIDM project (No. CSIDM-200805) partially funded by a grant from the National Research Foundation (NRF) administered by the Media Development Authority (MDA) of Singapore. We thank the anonymous reviewers for their insightful comments.

References

T. Abekawa and M. Okumura. 2006. Japanese dependency parsing using co-occurrence information and a combination of case elements. In *Proceedings of ACL-COLING*.

S. Bergsma, D. Lin, and R. Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *Proceedings of EMNLP*, pages 59-68.

S. Bergsma, E. Pitler, and D. Lin. 2010. Creating robust supervised classifier via web-scale N-gram data. In *Proceedings of ACL*.

T. Brants and Alex Franz. 2006. The Google Web 1T 5-gram Corpus Version 1.1. LDC2006T13.

H. Calvo and A. Gelbukh. 2004. Acquiring selectional preferences from untagged text for prepositional phrase attachment disambiguation. In *Proceedings of VLDB*.

H. Calvo and A. Gelbukh. 2006. DILUCT: An open-source Spanish dependency parser based on rules, heuristics, and selectional preferences. In *Lecture Notes in Computer Science 3999*, pages 164-175.

X. Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of EMNLP-CoNLL*, pages 957-961.

X. Carreras, M. Collins, and T. Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of CoNLL*.

E. Charniak, D. Blaheta, N. Ge, K. Hall, and M. Johnson. 2000. BLLIP 1987-89 WSJ Corpus Release 1, LDC No. LDC2000T43.Linguistic Data Consortium.

W. Chen, D. Kawahara, K. Uchimoto, and Torisawa. 2009. Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of EMNLP*, pages 570-579.

K. W. Church and P. Hanks. 1900. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22-29.

R. L. Cilibrasi and P. M. B. Vitanyi. 2007. The Google similarity distance. IEEE Transaction on Knowledge and Data Engineering, 19(3):2007. pages 370-383.

M. Collins, A. Globerson, T. Koo, X. Carreras, and P. L. Bartlett. 2008. Exponentiated gradient algorithm for conditional random fields and max-margin markov networks. *Journal of Machine Learning Research*, pages 1775–1822.

M. Collins, P. Koehn, and I. Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL*, pages 531-540.

S. Corston-Oliver, A. Aue, Kevin. Duh, and E. Ringger. 2006. Multilingual dependency parsing using bayes point machines. In *Proceedings of NAACL*.

H. Daumé III. 2007. Frustrating easy domain adaptation. In *Proceedings of ACL*.

E. F. Drabek and Q. Zhou. 2000. Using co-occurrence statistics as an information source for partial parsing of Chinese. In *Proceedings of Second Chinese Language Processing Workshop, ACL*, pages 22-28.

Y. GoldBerg and M. Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Proceedings of NAACL*, pages 742-750.

D. Graff. 2003. English Gigaword, LDC2003T05.

J. Hall, J. Nivre, and J. Nilsson. 2006. Discriminative classifier for deterministic dependency parsing. In *Proceedings of ACL*, pages 316-323.

M. Johnson and S. Riezler. 2000. Exploiting auxiliary distribution in stochastic unification-based garmmars. In *Proceedings of NAACL*.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*, pages 595-603.

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

M. Lapata and F. Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1), pages 1-30.

M. Lauer. 1995. Corpus statistics meet the noun compound: some empirical results. In *Proceedings of ACL*.

D. K. Lin, H. Church, S. Ji, S. Sekine, D. Yarowsky, S. Bergsma, K. Patil, E. Pitler, E. Lathbury, V Rao, K. Dalwani, and S. Narsale. 2010. New tools for web-scale n-grams. In *Proceedings of LREC*.

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

A. F. T. Martins, D. Das, N. A. Smith, and E. P. Xing. 2008. Stacking dependency parsers. In *Proceedings of EMNLP*, pages 157-166.

D. McClosky, E. Charniak, and M. Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of ACL*.

D. McClosky, E. Charniak, and M. Johnson. 2010. Automatic Domain Adapatation for Parsing. In *Proceedings of NAACL-HLT*.

R. McDonald and J. Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of EMNLP-CoNLL*.

R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*, pages 81-88.

R. McDonald, K. Crammer, and F. Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91-98.

P. Nakov and M. Hearst. 2005. Search engine statistics beyond the n-gram: application to noun compound bracketing. In *Proceedings of CoNLL*.

J. Nivre and R. McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL*, pages 950-958.

G. van Noord. 2007. Using self-trained bilexical preferences to improve disambiguation accuracy. In *Proceedings of IWPT*, pages 1-10.

PennBioIE. 2005. Mining the bibliome project, 2005. http:bioie.ldc.upenn.edu/.

E. Pitler, S. Bergsma, D. Lin, and K. Church. 2010. Using web-scale N-grams to improve base NP parsing performance. In *Proceedings of COLING*, pages 886-894.

P. Resnik. 1993. *Selection and information: a class-based approach to lexical relationships*. Ph.D. thesis, University of Pennsylvania.

J. Suzuki, H. Isozaki, X. Carreras, and M. Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of EMNLP*, pages 551-560.

J. Suzuki and H. Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of ACL*, pages 665-673.

P. D. Turney. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4).

J. Veronis. 2005. Web: Google adjusts its counts. Jean Veronis' blog: http://aixtal.blogsplot.com/2005/03/web-google-adjusts-its-count.html.

M. Volk. 2001. Exploiting the WWW as corpus to resolve PP attachment ambiguities. In *Proceedings of the Corpus Linguistics*.

Q. I. Wang, D. Lin, and D. Schuurmans. 2007. Simple training of dependency parsers via structured boosting. In *Proceedings of IJCAI*, pages 1756-1762.

Yamada and Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*, pages 195-206.

A. Yates, S. Schoenmackers, and O. Etzioni. 2006. Detecting parser errors using web-based semantic filters. In *Proceedings of EMNLP*, pages 27-34.

Y. Zhang and S. Clark. 2008. A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proceedings of EMNLP*, pages 562-571.