Lexical categories for improved parsing of web data

Lilja ØVRELID *Arne SKJÆRHOLT*
Department of Informatics
University of Oslo
{liljao,arnskj}@ifi.uio.no

ABSTRACT
We investigate the use of features expressing lexical generalizations over word forms when parsing web data and experiment with a range of web text samples, taken from the Ontonotes corpus, as well as the web 2.0 data sets described in Foster et al. (2011b). We obtain significant improvements for a standard data-driven dependency parser when incorporating features expressing these lexical categories, and in fact find that we may dispense with word form features altogether and still observe the same levels of improvement.

KEYWORDS: Syntactic parsing, data-driven dependency parsing, web language, clustering, lemmatization, delexicalization.

Proceedings of COLING 2012: Posters, pages 903–912,
COLING 2012, Mumbai, December 2012.

1 Introduction

Syntactic analysis of web language has been shown to pose several challenges for traditional parsers trained on edited news text. First of all, web text does not represent a uniform domain or genre, but varies greatly, both in terms of topics and level of formality, where texts may range from edited articles to increasingly informal genres like blogs, user fora and tweets. It is well known that lexical statistics employed by the parser become less reliable when moving to a new domain (Gildea, 2001), and for web text like user forums and twitter data, the amount of unknown words may be as high as 17% (Foster et al., 2011a). In the same way, the performance of other post processing tools, such as part-of-speech taggers, also suffer (Foster et al., 2011b).

Even though parser lexicalization has been a topic of some debate, features incorporating information regarding lexical co-occurrence are employed in most state-of-the-art syntactic parsers in some form or other. Since the task of assigning word-to-word relations is at the core of dependency parsing, statistics regarding relations between different word forms in the training data provide vital information. These lexical statistics are, however, often sparse, and there exists a growing body of work which examines various strategies for generalizing over the distributions of words and using different kinds of lexical categories in syntactic parsing. Word clusters derived from unlabeled data have been shown to improve parsing accuracy for dependency parsing of English (Koo et al., 2008; Suzuki et al., 2009) and so have clusters derived from parsed data (Sagae and Gordon, 2009). Zhou et al. (2011) show that co-occurence based measures of word-to-word selectional preference derived from web-scale data sets can improve statistical dependency parsing. Furthermore, other types of lexical semantic information, such as named entity classes (Ciaramita and Attardi, 2007) and word sense information from WordNet (Agirre et al., 2011), have recently been shown to improve dependency parsing for English.

In this article, we investigate the use of different lexical categories when parsing a range of different types of web data with a state-of-the-art data-driven dependency parser. We examine the effect of enriching the parser with features detailing information about word cluster labels as well as lemma information. We furthermore revisit the role of parser lexicalization in the light of our findings.

The article is structured as follows: Section 2 presents the data used in our experiments and its enrichment with two types of lexical categories, whereas Section 3 describes the extended feature models used in order to enable the parser to take these categories into account. In Section 4 we go on to describe the experiments investigating the effects of these categories, as well as the effect of delexicalization. Finally, we conclude and outline some plans for future work.

2 Data

In the following, we present the different corpora used in our experiments, the preprocessing performed prior to experimentation and the enrichment of the data with automatically derived cluster labels and lemma information.

2.1 Corpora

We use the Wall Street Journal portion of the Penn Treebank sections 2-23, with the standard splits for training (2-21) and testing (23). Due to tokenization differences we train on both the original LDC version, as well as the version released with the Ontonotes corpus. Moreover,

we use a wide range of treebanked web data in our experiments. First, the Ontonotes corpus, release 4.0, contains web data from different sources. This portion of the Ontonotes corpus amounts to around 500,000 tokens (including punctuation) and 23,000 sentences split into six different data sets: translated Arabic-to-English (a2e; 55,000 tokens) and Chinese-to-English (c2e; 74,000 tokens) web text, P2.5 translated Arabic-to-English (p2.5_a2e; 16,000 tokens) and Chinese-to-English (p2.5_c2e; 22,000 tokens), as well as general English web data (eng; 71,500 tokens) and a large set of sentences originally selected to improve sense coverage in the corpus (sel; 279,000 tokens). Second, we use the user forum and twitter data sets described in Foster et al. (2011b), which contain a total of 1000 sentences split into development and test sets for user forums (on football topics) and twitter data.

As mentioned earlier, the amount of unknown words for a parser trained on the standard training sections of the Wall Street Journal has been reported to increase notably when moving to web data. For the data sets described above, this is also the case. Compared to a 2.5% proportion of unknown words for the test section of Wall Street Journal (section 23), we observe proportions ranging from 5.5% to 8.1% for the Ontonotes web data. The user forum data on football has 8% unknown words, whereas the twitter data has as much as 17.9% unknown words.

2.2 Preprocessing

The treebank data sets are converted to dependency representations using the Stanford parser, version 2.0, and its *basic* setting which performs a conversion of PTB-style phrase structure trees and provides a dependency graph which is a directed tree (de Marneffe et al., 2006). The dependency representations result from a conversion of PTB-style phrase structure trees, combining 'classic' head finding rules with rules that target specific linguistic constructions.

The data is subsequently PoS-tagged using SVMTool (Gimenez and Marquez, 2004) and the pretrained model for English available from the tool web page. In the Ontonotes data set all hyphens were in addition converted to the HYPH tag which is used in this data set.

2.3 Lexical categories

The data sets described above were enriched with information about the *lemma* of each token using the NLTK WordNet lemmatizer (Bird et al., 2009). The lemmatizer requires information about part-of-speech, hence lemmatization was performed separately on gold and automatically tagged data sets.

Following lemmatization, the data sets were further enriched with the *cluster labels* described in Turian et al. (2010), created using the Brown clustering algorithm (Brown et al., 1992) and induced from the RCV1 corpus, a corpus containing Reuters English newswire text, with approximately 63 million words and 3.3 million sentences. The Brown algorithm is a hierarchical clustering algorithm which clusters words by maximizing the mutual information of bigrams. Since the algorithm is hierarchical, cluster labels are simply unique identifiers of each node within the tree, expressing the path from the root, where 0 indicates a right branch and 1 a left branch. Furthermore, clusters may be extracted at various depths, giving clusters of different sizes. Brown clusters have previously been shown to improve statistical dependency parsing (Koo et al., 2008; Suzuki et al., 2009), as well as other NLP tasks such as Chunking and Named Entity Recognition (Turian et al., 2010).

Feature model	Features
Baseline	S_0p, S_1p, S_2p, S_3p, L_0p, L_1p, L_2p, I_0p, $S_{0l}p$, $S_{0r}p$, $S_{1r}p$, $S_{0l}d$, $S_{1r}d$, S_0w, S_1w, S_2w, L_0w, L_1w, $S_{0l}w$, $S_{1r}w$, S_0pS_1p, S_0wL_0w, S_0pS_0w, S_1pS_1w, L_0pL_0w, $S_{1r}dS_{0l}d$, $S_{1r}pS_{1l}p$, $S_0pS_1pL_0p$, $S_0pS_1pS_2p$, $S_0pL_0pL_1p$, $L_0pL_1pL_2p$, $L_1pL_2pL_3p$, $S_0pL_0pI_0p$, $S_1pS_{1l}dS_{1r}d$
+ PoS simple	S_0l, S_l1l, S_2l, S_3l, L_0l, L_1l, L_2l, I_0l, $S_{0l}l$, $S_{0r}l$, $S_{1r}l$
+ Form simple	S_0l, S_1l, S_2l, L_0l, L_1l, $S_{0l}l$, $S_{1r}l$
+ Form all	S_0l, S_1l, S_2l, L_0l, L_1l, $S_{0l}l$, $S_{1r}l$, S_0lL_0l, S_0pS_0l, S_1pS_1l, L_0pL_0l,

Table 1: Baseline and extended feature models, where p=PoS-tag, w=word form, d=dependency label in the graph constructed so far (if any), and l=lexical category, i.e., either cluster labels or lemma.

We experiment with several techniques for the introduction of cluster labels. First of all, we vary the number of clusters to be either 100, 320, 1000 or 3200 clusters. This means that the number of clusters is fixed prior to clustering. Koo et al. (2008) found the use prefixes of the cluster labels of various lengths (4 to 6) to be beneficial for parsing, so we adopt this approach in addition to using full-length labels. A third method for generalizing over the cluster labels is to use the lemma information directly in the assignment of clusters, so that all word forms with the same lemma are assigned identical cluster labels.

3 Parser features

We use Maltparser (Nivre et al., 2006) (v. 1.4.1), a system for data-driven dependency parsing which is based on a deterministic parsing strategy in combination with treebank-induced classifiers for predicting parse transitions. It supports a rich feature representation of the parse history and may easily be extended to take additional features into account. We choose to use Maltparser primarily due to its extendible feature model which facilitates experimentation with additional features during parsing.

As our baseline parser, we use the parse model described in Foster et al. (2011a), where Maltparser was employed to parse web 2.0 data. It employs the stacklazy algorithm (Nivre, 2009), along with the liblinear package (Fan et al., 2008) for inducing parse transition classifiers. The stacklazy algorithm operates over three data structures: a stack (S) of partially processed tokens, a list (I) of nodes that have been on the stack, and a "lookahead" list (L) of nodes that have not been on the stack. We refer to the top of the stack using S_0 and subsequent nodes using S_1, S_2, etc., and the leftmost/rightmost dependent of S_0 with S_{0l}/S_{0r}.

	Lexical categories – Ontonotes						
Parser	**wsj23$_{onto}$**	**a2e**	**c2e**	**p2.5a2e**	**p2.5c2e**	**eng**	**sel**
BaseGold	**89.27**	**84.85**	**82.22**	**84.99**	**86.11**	**83.89**	**83.61**
ClstGold	89.05$_{fa320}$	84.46$_{fs100}$	82.02$_{fs320}$	84.59$_{fs320}$	86.07$_{fs320}$	83.36$_{fs320}$	83.09$_{fs100}$
LemmGold	88.91$_{ps}$	84.38$_{ps}$	81.93$_{ps}$	84.46$_{ps}$	85.81$_{fa}$	83.41$_{ps}$	82.96$_{ps}$
BaseTag	86.24	78.35	75.38	79.40	79.38	76.99	74.84
ClstTag	**86.67**$_{fa320}$	**79.97**$_{fa100}$	**76.71**$_{fa100}$	**80.48**$_{fa100}$	**80.78**$_{fs100}$	**78.30**$_{fa320}$	**75.82**$_{fs100}$
LemmTag	86.49$_{ps}$	79.50$_{fs}$	76.41$_{ps}$	80.17$_{fa}$	80.60$_{fs}$	78.02$_{ps}$	75.43$_{fs}$

Table 2: Labeled accuracy scores (proportion of tokens with correct head *and* dependency label) for parsers trained on wsj02-21 and tested on wsj23 and Ontonotes web data sets, as well as web 2.0 (football and twitter) data sets using gold (Gold) and automatic (Tag) PoS-tags, for baseline (Base), as well as extended (Clst, Lemm) parsers, where results indicate the best configuration of feature model (ps=pos simple, fs=form simple, fa=form all) and cluster set size (100, 320, 1000 or 3200).

Table 1 provides the baseline feature model, along with three sets of additional features (PoS simple, Form simple, Form all), which are constructed by copying the full feature set ("all") or only the features that pertain to a single token ("simple") and involve either the PoS-tag or the word form. Note that the "PoS all" feature set proved to be too large for practical experimentation with lexical features derived from clusters or lemmas.

4 Experiments

We train two pairs of baseline parsers on the standard sections 2-21 of the WSJ data both with gold PoS-tags and automatically assigned PoS-tags, for the original tokenization and for the Ontonotes tokenization. The results for these parsers, evaluated on section 23 with original and Ontonotes tokenization are presented in Table 2 and 3, respectively (BaseGold, BaseTag). All results are provided as labeled accuracy scores (LAS), which express the proportion of tokens which were assigned correct head *and* dependency label by the parser. Statistical significance is checked using Bikel's randomized parsing evaluation comparator.

4.1 Lexical categories

The baseline parsers are subsequently applied to the different web data sets, as detailed in Section 2 above. We find that results vary with the degree of formality, ranging from LAS 85-86% for some of the more edited web data (Table 4), to LAS 76% for the twitter data (Table 4)). Just like Foster et al. (2011b), we find that the drop in performance following PoS tagging is considerably larger for the web data than the WSJ data (5-10 vs. 2-3 percentage points, respectively).

Tables 2 and 3 also show the results for the parsers with additional features: cluster labels (Clst) and lemmas (Lemm) over gold and tagged data, indicating the feature model (ps=pos simple, fs=form simple, fa=form all) and cluster set size (100, 320, 1000 or 3200) that produced the result. The addition of the cluster and lemma features are beneficial largely for

Lexical categories – Web 2.0			
Parser	**wsj23**$_{orig}$	**football**	**twitter**
BaseGold	89.83	79.62	**76.15**
ClstGold	**90.13**$_{fa1000}$	79.87$_{fs1000}$	75.93$_{fs3200}$
LemmGold	89.92$_{ps}$	**79.89**$_{fs}$	76.05$_{fs}$
BaseTag	87.83	73.86	65.57
ClstTag	**87.94**$_{fa100}$	**74.50**$_{fa100}$	**66.18**$_{fs100}$
LemmTag	87.76$_{fs}$	74.23$_{fs}$	65.55$_{fa}$

Table 3: Labeled accuracy scores (proportion of tokens with correct head *and* dependency label) for parsers trained on wsj02-21 and tested on wsj23 and web 2.0 data sets, using gold (Gold) and automatic (Tag) PoS-tags, for baseline (Base), as well as extended (Clst, Lemm) parsers, where results indicate the best configuration of feature model (ps=pos simple, fs=form simple, fa=form all) and cluster set size (100, 320, 1000 or 3200).

the parsers trained and tested with automatically assigned PoS-tags, and more so on the web data than the WSJ data. For the web data, the addition of cluster labels lead to improvements of 1-1.5 percentage points for the Ontonotes web data (Table 4), all differences being statistically significant ($p < 0.0001$).

We furthermore find that the cluster features provide significant improvements for the user forum data (football; $p < 0.05$), and small, but non-significant, improvements for the twitter data, see Table 3. The models that perform the best are the models copying the form features and using the smaller cluster sizes (100, 320) and exclusively the models which use the lemmatized assignment of full cluster labels described above. The prefix labels do not perform as well on these data sets and show best results on average 0.5 percentage points lower than the results presented in Table 2 and 3.

4.2 Delexicalization

Seeing that the lexical categories employed above gave clear improvements and knowing that the proportion of unknown words typically rises dramatically for web language texts, we investigate the role of lexicalization in the parsing of web language. We therefore train *delexicalized* parsers, i.e., where we modify the baseline feature model in Table 1 by removing all features involving word forms (w).

As shown by the results in Table 4 and 5, delexicalization causes an expected drop in performance over all data sets. We then add our cluster features and lemma features, using a fixed feature model, the "Form simple" model, and vary the cluster sizes as before. Not surprisingly, the results show that with a delexicalized model, the largest cluster size (3200) provides the best performing model throughout.

We furthermore observe that the delexicalized models including either clusters or lemmas significantly out-perform the lexicalized baseline for the automatically tagged web data sets ($p < 0.0001$) (Table 4) and the user forum data ($p < 0.01$) (Table 5), indicating that the

Delexicalization – Ontonotes							
Parser	**wsj23$_{onto}$**	**a2e**	**c2e**	**p2.5a2e**	**p2.5c2e**	**eng**	**sel**
BaseGold	**89.27**	**84.85**	**82.22**	**84.99**	**86.11**	**83.89**	**83.61**
DelexGold	81.02	77.07	73.72	77.75	76.36	75.65	75.92
DelexClstGold	88.99	84.44	81.74	84.41	85.41	83.27	82.86
DelexLemmGold	88.94	84.47	81.81	84.63	85.77	83.66	83.22
BaseTag	86.24	78.35	75.38	79.40	79.38	76.99	74.84
DelexTag	78.15	70.57	66.75	72.31	70.01	68.52	67.34
DelexClstTag	86.31	79.48	**76.51**	79.88	**80.55**	77.70	74.90
DelexLemmTag	**86.34**	**79.66**	76.32	**80.24**	80.44	**78.23**	**75.48**

Table 4: Labeled accuracy scores (proportion of tokens with correct head *and* dependency label) for delexicalized parsers trained on wsj02-21 and tested on wsj23 and Ontonotes web data sets, using gold (Gold) and automatic (Tag) PoS-tags, for baseline (Base), as well as extended (Clst, Lemm) parsers. All delexicalized extended experiments were performed using the form simple feature model and a cluster set size of 3200.

Delexicalization – Web 2.0			
Parser	**wsj23$_{orig}$**	**football**	**twitter**
BaseGold	89.83	79.62	76.15
DelexGold	81.37	70.84	68.72
DelexClstGold	89.91	79.53	76.19
DelexLemmGold	**89.97**	**79.93**	**76.37**
BaseTag	**87.83**	73.86	**65.57**
DelexTag	79.14	65.90	58.77
DelexClstTag	87.60	73.98	65.18
DelexLemmTag	87.83	**74.47**	65.51

Table 5: Labeled accuracy scores (proportion of tokens with correct head *and* dependency label) for delexicalized parsers trained on wsj02-21 and tested on wsj23, Ontonotes web data and web 2.0 (football and twitter) data sets, using gold (Gold) and automatic (Tag) PoS-tags, for baseline (Base), as well as extended (Clst, Lemm) parsers. All delexicalized extended experiments were performed using the form simple feature model and a cluster set size of 3200.

generalizations provided by clustering and/or lemmatization help overcome some of the sparsity problems mentioned initially. We furthermore observe that the delexicalized models including clusters and/or lemmas perform only marginally worse than their lexicalized counterparts. Seeing that word token features are used in most state-of-the-art parsers today, the finding that we may dispense of these completely and still observe the same level of improvements using the cluster label and/or lemma information is highly interesting. Our work indicates that these types of lexical categories capture many important properties of word tokens and even generalize over these so that lexical constraints may be acquired even when individual word features prove too sparse due to domain and genre differences.

Conclusion and future work

We have shown how lexical features derived from clusters and lemmas may improve data-driven dependency parsing of web data and even replace individual word forms during parsing. The addition of the cluster and lemma features are beneficial largely for the parsers trained and tested with automatically assigned PoS-tags, and more so on the web data than the WSJ data. We furthermore find that the delexicalized models including information about either clusters or lemmas significantly out-perform the lexicalized baseline for the automatically tagged web data sets.

In terms of future work, we plan to experiment with other parsers and other clustering algorithms. We would also like to perform similar experiment with data taken from other genres and/or domains.

Acknowledgments

Our thanks go to Jennifer Foster for sharing her Web 2.0 (football and twitter) data. Thanks also to our colleagues at UiO and the anonymous reviewers for their comments.

References

Agirre, E., Bengoetxa, K., Gojenola, K., and Nivre, J. (2011). Improving dependency parsing with semantic classes. In *Proceedings of the 49th Meeting of the Association for Computational Linguistics*.

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly, Beijing.

Brown, P., deSouza, P., Mercer, R., Pietra, V., and Lai, J. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18.

Ciaramita, M. and Attardi, G. (2007). Dependency parsing with second-order feature maps and annotated semantic information. In *Proceedings of the 10th International Conference on Parsing Technologies*.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Fan, R., Chang, K., Hsieh, C., Wang, X., and Lin, C. (2008). Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, (9).

Foster, J., Çetinoglu, Ö., Wagner, J., Roux, J. L., Hogan, S., Nivre, J., Hogan, D., and van Genabith, J. (2011a). hardtoparse: Pos tagging and parsing the twitterverse. In *Proceedings of AAAI-11 Workshop on Analysing Microtext*.

Foster, J., Çetinoglu, Ö., Wagner, J., Roux, J. L., Nivre, J., Hogan, D., and van Genabith, J. (2011b). From news to comment: Resources and benchmarks for parsing the language of web 2.0. In *Proceedings of IJCNLP*.

Gildea, D. (2001). Corpus variation and parser performance. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 167–202.

Gimenez, J. and Marquez, L. (2004). SVMTool: A general POS tagger generator based on Support Vector Machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of the 46th Meeting of the Association for Computational Linguistics*.

Nivre, J. (2009). Non-projective dependency parsing in expected linear time. In *Proceedings of the 47th Meeting of the Association for Computational Linguistics*.

Nivre, J., Hall, J., and Nilsson, J. (2006). MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Sagae, K. and Gordon, A. S. (2009). Clustering words by syntactic similarity improves dependency parsing of predicate argument structures. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT)*, pages 192–201.

Suzuki, J., Isozaki, H., Carreras, X., and Collins, M. (2009). An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Meeting of the Association for Computational Linguistics*.

Zhou, G., Zhao, J., Liu, K., and Cai, L. (2011). Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of the 49th Meeting of the Association for Computational Linguistics*.