Forest-guided Supertagger Training

Yao-zhong Zhang † **Takuya Matsuzaki** † **Jun'ichi Tsujii**†‡§

† Department of Computer Science, University of Tokyo
‡ School of Computer Science, University of Manchester
§National Centre for Text Mining
{yaozhong.zhang, matuzaki, tsujii}@is.s.u-tokyo.ac.jp

Abstract

Supertagging is an important technique for deep syntactic analysis. A supertagger is usually trained independently of the parser using a sequence labeling method. This presents an inconsistent training objective between the supertagger and the parser. In this paper, we propose a forest-guided supertagger training method to alleviate this problem by incorporating global grammar constraints into the supertagging process using a CFG-filter. It also provides an approach to make the supertagger and the parser more tightly integrated. The experiment shows that using the forest-guided trained supertagger, the parser got an absolute 0.68% improvement from baseline in F-score for predicate-argument relation recognition accuracy and achieved a competitive result of 89.31% with a faster parsing speed, compared to a state-of-the-art HPSG parser.

1 Introduction

Deep syntactic analysis by lexicalized grammar parsing, which provides linguistic-rich information for many NLP tasks, has recently received more and more attention from the NLP community. To use a deep parser in real large-scale applications, speed is an important issue to take into consideration. Supertagging is one of the speed-up technique widely used for lexicalized grammar parsing. A supertagger is used to limit the number of plausible lexical entries fed to the parser, this can greatly reduce the search space for the parser. Supertagging was first proposed for Lexicalized Tree Adjoining Grammar (LTAG) (Bangalore and Joshi, 1999), and then successfully applied to Combinatory Categorial Grammar (CCG) (Clark, 2002) and Head-driven Phrase Structure Grammar (HPSG) (Ninomiya et al., 2006). In addition, supertags can also be used for other NLP tasks besides parsing, such as semantic role labeling (Chen and Rambow, 2003) and machine translation (Birch et al., 2007; Hassan et al., 2007) to utilize syntactic information in the supertags.

In lexicalized grammar parsing, supertagging is usually treated as a sequence labeling task independently trained from the parser. Previous research (Clark, 2002) showed that even a pointwise classifier not considering context edge features is effective when used as a supertagger. To make up for the insufficient accuracy as a single-tagger, more than one supertag prediction is reserved and the parser takes the burden of resolving the rest of the supertag ambiguities.

A non-trivial problem raised by the separate training of the supertagger is that the prediction score provided by the supertagger might not be suitable for direct use in the parsing process, since a separately trained supertagger that does not take into account grammar constraints has a training objective which is inconsistent with the parser. Although the scores provided by the supertagger can be ignored (e.g., in some CCG parsers), this may also discard some useful information for effective beam search and accurate disambiguation.

Based on this observation, we assume that considering global grammar constraints during the supertagger training process would make the supertagger and the parser more tightly integrated.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 1281–1289,
Beijing, August 2010

In this paper, we propose an on-line forest-guided training method for a supertagger to make the training objective of a supertagger more closely related to the parsing task. We implemented this method on a large-scale HPSG grammar. We used a CFG grammar to approximate the original HPSG grammar in the supertagging stage and applied best-first search to select grammar-satisfying supertag sequences for the parameter updating. The experiments showed that the HPSG parser is improved by considering structure constraints in the supertagging training process. For the standard test set (Penn Treebank Section 23), we accomplished an absolute 0.68% improvement from baseline in F-score for predicate-argument relation recognition and got a competitive result of 89.31% with a faster parsing speed, compared to a state-of-the-art HPSG parser.

The remainder of the paper is organized as follows: in section 2 we provide the necessary background regarding HPSG parsing. In section 3, we introduce the on-line forest-guided supertagger training method. Section 4 shows the experiment results and the related analysis. Section 5 compares the proposed approach with related work and section 6 presents our conclusions and future work.

2 Background

2.1 Statistical HPSG Parsing

HPSG (Pollard and Sag, 1994) is a lexicalist grammar framework. In HPSG, a large number of lexical entries are used to express word-specific characteristics, while only a small number of rule schemata are used to describe general construction rules. Typed feature structures named "signs" are used to represent both lexical entries and phrasal constituents. A classic efficient statistical HPSG parsing process is depicted in Figure 1. Given a word and part-of-speech sequence (w, p) as input, the first step (called "supertagging") in HPSG parsing is to assign possible lexical entries. In practice, for each word, more than one supertag is reserved for the parser. Then, the parser searches the given lexical entry space to construct a HPSG tree using the rule schemata to combine possible signs. Constituent-based methods and transition-based methods can be used for tree structure disambiguation. This parsing framework using supertagging is also used in other lexicalized grammars, such as LTAG and CCG.

2.2 HPSG Supertagging

Like other lexicalized grammar, the lexical entries defined in HPSG are referred to as "supertags". For example, the word "like" is assigned a lexical entry for transitive verbs in non-3rd person present form, which indicates that the head syntactic category of "like" is verb and it has an NP subject and an NP complement. With such fine-grained grammatical type distinctions, the number of supertags is very large. Compared to the 45 part-of-speech (POS) tags defined in the PennTreebank, the HPSG grammar we used contains 2,308 supertags. The large number and the complexity of the supertags makes supertagging harder than the POS tagging task.

Supertagging can be formulated as a sequence labeling task. Here, we follow the definition of Collins' perceptron (Collins, 2002). The training objective of supertagging is to learn the mapping from a POS-tagged word sentence $\boldsymbol{w} = (w_1/p_1, ..., w_n/p_n)$ to a sequence of supertags $\boldsymbol{s} = (s_1, ..., s_n)$. We use function $\boldsymbol{GEN(w)}$ to indicate all candidates of supertag sequences given input $\boldsymbol{w}$. Feature function $\boldsymbol{\Phi}$ maps a sample $(\boldsymbol{w}, \boldsymbol{s})$ to a point in the feature space R^d. $\boldsymbol{\theta}$ is the vector of feature weights. Given an input $\boldsymbol{w}$, the most plausible supertag sequence is found by the prediction function defined as follows:

$$F(\boldsymbol{w}) = \underset{\boldsymbol{s} \in \boldsymbol{GEN(w)}}{argmax} \ \theta \cdot \Phi(\boldsymbol{w}, \boldsymbol{s}) \qquad (1)$$

2.3 CFG-filtering

CFG-filtering (Kiefer and Krieger, 2000) is a technique to find a superset of (packed) HPSG parse trees that satisfy the constraints in a grammar. A CFG that approximates the original HPSG grammar is used for efficiently finding such trees without doing full-fledged HPSG parsing that is computationally demanding because the schema application involves unification operations among large feature structures (signs). The number of possible signs is infinite in general and hence

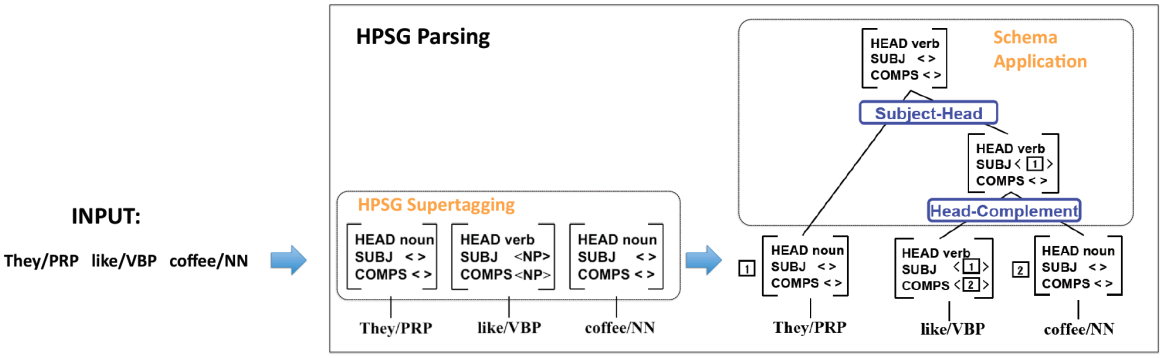

Figure 1: HPSG parsing for the sentence "They like coffee."

some features (e.g., the number agreement feature) are ignored in the approximating CFG so that the set of possible signs can be approximated by a finite set of non-terminal symbols in the CFG. By this construction, some illegal trees may be included in the set of trees licensed by the approximating CFG, but none of the well-formed trees (i.e., those satisfying all constraints in the grammar) are excluded by the approximation. We use the algorithm described by Kiefer and Krieger (2000) to obtain the approximating CFG for the original HPSG. The technical details regarding the algorithm can be found in Kiefer and Krieger (2000).

3 Forest-guided Training for Supertagging

3.1 Motivation

In lexicalized grammar parsing, a parser aims to find the most plausible syntactic structure for a given sentence based on the supertagging results. One efficient parsing approach is to use prediction scores provided by the supertagger. Usually, the supertagger is trained separately from the structure disambiguation in a later stage. This pipeline parsing strategy poses a potential problem in that the training objective of a supertagger can deviate from the final parser, if the global grammar constraints are not considered. For example, the supertag predictions for some words can contribute to high supertagging accuracy, but cause the parser to fail. Therefore, considering the global grammar constraints in the supertagging training stage can make the supertagger and the parser more tightly related, which will contribute towards the performance of the parser.

Algorithm 1: Forest-guided supertagger training

Input: Training Sample $(w_i, s_i)_{i=1,...,N}$,
Number of iterations T

$\theta \leftarrow (0, ..., 0)$, $\theta_{sum} \leftarrow (0, ..., 0)$
for $iterNum \leftarrow 1$ to T **do**
  for i $\leftarrow 1$ to N **do**
    Generate supertag lattice using the point-wise classifier with current θ
    Select $\hat{s}_i$ from the lattice which **can construct a tree** with largest sequence score
    if(No $\hat{s}_i$ satisfied grammar constraints)
    $\hat{s}_i \leftarrow \arg\max_{s \in \mathbf{GEN}(w_i)} \theta_i \cdot \Phi(w_i, s_i)$
    if $\hat{s}_i \neq s_i$ then
      $\theta_{i+1} \leftarrow \theta_i + \Phi(w_i, s_i) - \Phi(w_i, \hat{s}_i)$
      $\theta_{sum} \leftarrow \theta_{sum} + \theta_{i+1}$
Return: θ_{sum}/NT

3.2 Training Algorithm

Based on the motivation above, we propose a forest-guided supertagger training method to make the supertagger more tightly integrated with the parser. This method is based on the averaged perceptron training algorithm. The training process is given in Algorithm 1.

The most important difference of the proposed algorithm compared to the traditional supertagger training method is that the current best-scored supertag sequence is searched only within the space of the supertag sequences that are allowed by the grammar. As for whether the grammar

constraints are satisfied, we judge it by whether a possible syntactic tree can be constructed using the given supertag sequence. We do not require the constructed syntactic tree to be identical to the gold tree in the corpus. For this reason we call it "forest-guided".

In the forest-guided training of the supertagger, an approximating CFG is used to filter out the supertag sequences from which no well-formed tree can be built. It is implemented as a best-first CFG parser wherein the score of a constituent is the score of the supertag (sub-)sequence on the fringe of the constituent, which is calculated using the current value of the parameters. Note that the best-first parser can find the best-scored supertag sequence very efficiently given proper scoring for the candidate supertag set for each token; this is actually the case in the course of training except for the initial phase of the training, wherein the parameter values are not well-tuned. The efficiency is due to the sparseness of the approximating CFG (i.e., the production rule set includes only a tiny fraction of the possible parent-children combinations of symbols) and highest-scored supertags often have a well-formed tree on top of them.

As is clear from the above description, the use of CFG-filter in the forest-guided training of the supertagger is not essential but is only a subsidiary technique to make the training faster. The improvement by the forest-guided training should however depend on whether the CFG approximation is reasonably tight or not. Actually, we managed to obtain a manageable size out of a CFG grammar, which includes 80 thousand nonterminal symbols and 10 million rules, by eliminating only a small number of features (semantics, case and number agreement, and fine distinctions in nouns, adjectives and complementizers). We thus believe that the approximation is fairly tight.

This training algorithm can also be explained in a search-based learning framework (Hal Daumé III and Daniel Marcu, 2005). In this framework, the objective of learning is to optimize the $\boldsymbol{\theta}$ for the enqueue function to make the good hypotheses rank high in the search queue. The rank score $\boldsymbol{r}$ consists of two components: path score $\boldsymbol{g}$ and heuristic score $\boldsymbol{h}$. In the forest-guided training method, $\boldsymbol{r}$ can be rewritten as follows:

$$\begin{aligned} r &= g + h \\ &= \theta \cdot \Phi(x, \hat{y}) + 1_{[Tree(\hat{y})]} * Penalty \end{aligned} \quad (2)$$

The heuristic part $\boldsymbol{h}$ checks whether the supertag candidate sequence satisfies the grammar constraints: if no CFG tree can be constructed, -∞ penalty is imposed to the candidate sequence in the forest-guided training method.

4 Experiments

We mainly evaluated the proposed forest-guided supertagger training method on HPSG parsing. Supertagging accuracy[1] using different training methods was also investigated.

4.1 Corpus Description

The HPSG grammar used in the experiments is Enju version 2.3[2]. It is semi-automatically converted from the WSJ portion of PennTreebank (Miyao, 2006). The grammar consists of 2,308 supertags in total. Sections 02-21 were used to train different supertagging models and the HPSG parser. Section 22 and section 23 were used as the development set and the test set respectively. We evaluated the HPSG parser performance by labeled precision (LP) and labeled recall (LR) of predicate-argument relations of the parser's output as in previous works (Miyao, 2005). All experiments were conducted on an AMD Opteron 2.4GHz server.

Template Type	Template
Word	w_i,w_{i-1},w_{i+1}, $w_{i-1}\&w_i$, $w_i\&w_{i+1}$
POS	p_i, p_{i-1}, p_{i-2}, p_{i+1}, p_{i+2}, $p_{i-1}\&p_i$, $p_{i-2}\&p_{i-1}$, $p_{i-1}\&p_{i+1}$, $p_i\&p_{i+1}$, $p_{i+1}\&p_{i+2}$
Word-POS	$p_{i-1}\&w_i$, $p_i\&w_i$, $p_{i+1}\&w_i$

Table 1: Feature templates used for supertagging models.

[1] "UNK" supertags are ignored in evaluation as in previous works.

[2] http://www-tsujii.is.s.u-tokyo.ac.jp/enju/index.html

4.2 Baseline Models and Settings

We used a point-wise averaged perceptron (PW) to train a baseline supertagger. Point-wise classifiers have been reported to be very effective and with competitive results for the supertagging task (Clark, 2002; Zhang et al., 2009). The number of training iterations was set to 5. The features used in the supertaggers are described in Table 1. For comparison, these features are identical to the features used in the previous works (Matsuzaki et al., 2007; Ninomiya et al., 2007). To make the training efficient, we set the default chart size limit for the forest-guided supertagger training to be 20k by tuning it on the development set.

We combined the supertagger trained under forest-guidance with a supertagging-based HPSG parser (Matsuzaki et al., 2007) and evaluated the contribution of the improved supertagger training procedure for the final HPSG parsing by the accuracy of the predicate-argument relations output of the parser. The parser crucially depends on the supertagger's performance in that it outputs the first well-formed tree successfully constructed on the highest scored supertag sequence. The highest-scored supertag sequences are enumerated one by one in descending order in regards to their score. The enumeration is actually implemented as n-best parsing on the supertag candidates using an approximating CFG. The HPSG tree construction on a supertag sequence is done using a shift-reduce style parsing algorithm equipped with a classifier-based action selection mechanism.

The automatically assigned POS tags were given by a maximum entropy tagger with roughly 97% accuracy.

4.3 Supertagging Results

Although we mainly focused on improving the final HPSG parsing performance through the improved supertagger training, it is also very interesting to investigate the supertagger performance using different training methods. To evaluate the forest-guided training method for a supertagger, we also need to incorporate structure constraints in the test stage. To make fair comparisons, for the averaged perceptron trained supertagger we also add structure constraints in its testing.

	Model Name	Acc%
auto-POS	FT+CFG	**92.77**
	PW+CFG	92.47
	PW	91.14
	ME	91.45
gold-POS	FT+CFG	**93.98**
	PW+CFG	93.70
	PW	92.48
	ME	92.78

Table 2: Supertagging results in section 23. "FT" represents the forest-guided trained supertagger. "PW" is the baseline average perceptron trained supertagger. "ME" is the supertagger trained by using the maximum entropy method. "+CFG" indicates the use of the CFG-filter for the supertagger results. The accuracy of automatically assigned POS tags in this section is 97.39%.

For simplicity, throughout this paper, we call the forest-guided trained supertagger "FT" in short, while the "PW" is used to represent the baseline point-wise averaged perceptron supertagger. "ME" is the re-implemented maximum entropy supertagger described in Matsuzaki et al. (2007).

For the PW supertagger, the performance was roughly 0.3% below the ME supertagger. Similar results were reported by Zhang et al. (2009), which used a Bayes point machine to reduce the gap between the averaged perceptron supertagger and the maximum entropy supertagger. Although we expected the ME supertagger using CFG-filter to give better results than the PW supertagger, implementing forest-guided supertagger training in a maximum entropy framework is different and more sophisticated than the current on-line training method. Considering that the performance of the PW supertagger and the ME supertagger were at a similar level, we chose the PW supertagger as our baseline.

We used a CFG-filter to incorporate global grammar constraints into both the training and the testing phase. Compared to the PW supertagger, the PW+CFG supertagger incorporated global grammar constraints only in the test phase, while for the FT+CFG supertagger, the global grammar constraints were incorporated both in

Training Method \ Iter NUM	1	2	3	4	5	Total Time
FT	6684s	4189s	3524s	3285s	3086s	$\approx$ 5.8h
PW	99s	116s	117s	117s	117s	$\approx$ 10 min
ME	/					$\approx$ 3h

Table 3: Supertagger training time on section 02-21. "FT" and "PW" represent forest-guided training and point-wise averaged perceptron training separately. "ME" is the point-wise maximum entropy training reported in Matsuzaki et al. (2007).

the training and the testing stage. The supertagging accuracy for different models is shown in Table 2. Firstly, incorporating grammar constraints only in the testing phase (PW+CFG) gave an absolute 1.22% (gold POS) and 1.33% (auto POS) increase in F-score compared to the PW supertagger. Secondly, incorporating grammar constraints into both the training and the testing stage (FT+CFG) gave an additional 0.28% (gold POS) and 0.3% (auto POS) improvement over the PW+CFG supertagger with p-values 0.0018 (gold POS) and 0.0016 (auto POS).

This also indicates that the supertagger and the parser are closely related to each other. The original motivation for supertagging is using simple models to resolve lexical ambiguities, which can efficiently reduce the search space of the parser. A better supertagger can contribute to more efficient and more accurate lexicalized grammar parsing. Actually, a supertagger can act as a coarse parser for the whole parsing process as well, as long as the coarse parser is efficient. Since supertag disambiguation is highly constrained by the grammar, incorporating grammar constraints into supertagging (including training and testing) by using the CFG-filter can further improve the supertagging performance, as shown in Table 2.

As for the supertagger training time, incorporating grammar constraints inevitably increases the training time. As shown in Table 3, the total training time of forest-guided training (default settings, with chart size limited to 20k) was about 5.8 hours. For each iteration of the FT model, we find that the training time gradually decreases with each successive iteration. This hints that we can do better model initialization to further reduce the training time.

4.4 HPSG Parsing Results

We evaluated the HPSG parsers using different supertagger training methods. For the baseline HPSG parser, a CFG-filter is already incorporated to accelerate the parsing process. In the following experiments, we fed the parser all the possible supertag candidates with the prediction scores generated by the supertaggers. We controlled the upper bound of the chart size in the CFG-filter to make the parser more efficient.

Table 4 shows the results of the different parsing models. We first compared the baseline parsers using different supertaggers. The forest-guided supertagger improved the final FT parser's F-score by 0.68% (statistically significant) over the PW parser using the PW supertagger, which did not consider global grammar constraints during the supertagger training process. The parsing time of the FT parser was very close to that of the PW parser (108s vs. 106s), which was also efficient. The result empirically reflects that incorporating the global grammar constraints into the supertagger training process can refine supertag predicting scores, which become more consistent and compatible with the parser.

We also compared our results with a state-of-the-art HPSG parser using the same grammar. Enju (Miyao, 2005; Ninomiya et al., 2007) is a log-linear model based HPSG parser, which uses a maximum entropy model for the structure disambiguation. In contrast to our baseline parser, full HPSG grammar is directly used with CKY algorithm in the parsing stage. As for the parsing performance, our baseline PW parser using the PW supertagger was 0.23% below the Enju parser. However, by using the forest-guided trained supertagger, our improved FT parser per-

Parser	UP	UR	LP	LR	F-score	Time †
FT Parser	92.28	92.14	89.38	89.23	**89.31**	108s
PW Parser	91.88	91.63	88.75	88.51	88.63	106s
Enju 2.3	92.26	92.21	88.89	88.84	88.86	775s

Table 4: Parser performance on Section 23. "FT Parser" represents baseline parser which uses forest-guided trained supertagger. "PW Parser" represents the baseline parser which uses the point-wise averaged perceptron trained supertagger. (†) The time is the total time of both supertagging and parsing and it was calculated on all 2291 sentences of the Section 23.

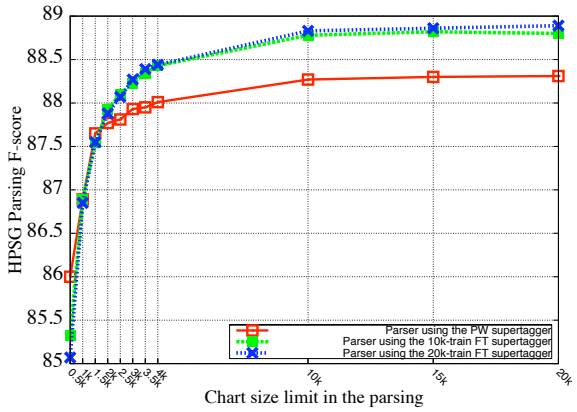

Figure 2: The F-score of the HPSG parsers on section 22 using different settings for the chart size limit in supertagger training and parsing.

formed 0.45% better than the Enju parser (default settings) in F-score. In addition, our shift-reduce style parser was faster than the Enju parser.

Beam size plays an important role for the forest-guided supertagger training method, since a larger beam size reduces the possibility of search errors. Precisely speaking, we control the beam size by limiting the number of edges in the chart in both the forest-guided supertagger training process and the final parsing. Figure 2 shows the results of setting different limits for the chart size during supertagger training and parsing on the development set. The X-axis represents the chart size limitation for the parsing. "10k-train" represents the chart size to be limited to 10k during FT supertagger training phase. A similar representation is used for "20k-train". There is no tree structure search process for the baseline PW supertagger. We evaluated the F-score of the parsers using different supertaggers. As shown in Figure 2, when the chart size of the parser was more than 10k, the benefit of using forest-guided supertaggers were obvious (around an absolute 0.5% improvement in F-score, compared to the parser using the baseline PW supertagger). The performance of the parser using "10k-train" FT supertagger was already approaching to that of the parser using "20k-train" FT supertagger. When the chart size of the parser was less than 2000, the forest-guided supertaggers were not work. Similar to the results showed in previous research (Hal Daumé III and Daniel Marcu, 2005), it is better to use the same chart size limit in the forest-guided supertagger training and the final parsing.

5 Related Work

Since the supertagging technique is well known to drastically improve the parsing speed and accuracy, there is work concerned with tightly integrating a supertagger with a lexicalized grammar parser. Clark and Curran (2004) investigated a multi-tagger supertagging technique for CCG. Based on the multi-tagging technique, supertagger and parser are tightly coupled, in the sense that the parser requests more supertags if it fails. They (Clark and Curran, 2007) also used the perceptron algorithm to train a CCG parser. Different from their work, we focused on improving the performance of the deep parser by refining the training method for supertagging. Ninomiya et al. (2007) used the supertagging probabilities as a reference distribution for the log-linear model for HPSG, which aimed to consistently integrate supertagging into probabilistic HPSG parsing. Prins et al. (2001) trained a POS-tagger on an automatic parser-generated lexical entry corpus as a filter for Dutch HPSG parsing to improve the parsing speed and accuracy.

The existing work most similar to ours is Boullier (2003). He presented a non-statistical parsing-based supertagger for LTAG. Similar to his method, we used a CFG to approximate the original lexicalized grammar. The main difference between these two methods is that we consider the grammar constraints in the training phase of the supertagger, not only in the supertagging test phase and our main objective is to improve the performance of the final parser.

6 Conclusions and Future Work

In this paper, based on the observation that supertaggers are commonly trained separately from lexicalized parsers without global grammar constraints, we proposed a forest-guided supertagger training method to integrate supertagging more tightly with deep parsing. We applied this method to HPSG parsing and made further significant improvement for both supertagging (0.28%) and the HPSG parsing (0.68%) compared to the baseline. The improved parser also achieved a competitive result (89.31%) with a faster parsing speed, compared to a state-of-the-art HPSG parser.

For future work, we will try to weight the forest trees for the supertagger training and extend this method to other lexicalized grammars, such as LTAG and CCG.

Acknowledgments

We are grateful to the anonymous reviewers for their valuable comments. We also thank Goran Topic and Pontus Stenetorp for their help proofreading this paper. The first author was supported by The University of Tokyo Fellowship (UT-Fellowship). This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan).

References

Bangalore, Srinivas and Aravind K. Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25:237–265.

Birch, Alexandra, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 9–16.

Boullier, P. 2003. Supertagging: A non-statistical parsing-based approach. In *In Proceedings IWPT-2003*, volume 3, pages 55–65.

Chen, John and Owen Rambow. 2003. Use of deep linguistic features for the recognition and labeling of semantic arguments. In *Proceedings of EMNLP-2003*, pages 41–48.

Clark, Stephen and James R. Curran. 2004. The importance of supertagging for wide-coverage ccg parsing. In *Proceedings of COLING-04*, pages 282–288.

Clark, S. and J.R. Curran. 2007. Perceptron training for a wide-coverage lexicalized-grammar parser. In *Proceedings of the Workshop on Deep Linguistic Processing*, pages 9–16.

Clark, Stephen. 2002. Supertagging for combinatory categorial grammar. In *Proceedings of the 6th International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+ 6)*, pages 19–24.

Collins, M. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP-2002*, pages 1–8.

Hal Daumé III and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *International Conference on Machine Learning (ICML)*, pages 169–176.

Hassan, Hany, Mary Hearne, and Andy Way. 2007. Supertagged phrase-based statistical machine translation. In *Proceedings of ACL-2007*, pages 288–295.

Kiefer, Bernd and Hans-Ulrich Krieger. 2000. A context-free approximation of head-driven phrase structure grammar. In *Proceedings of IWPT-2000*, pages 135–146.

Matsuzaki, Takuya, Yusuke Miyao, and Jun'ichi Tsujii. 2007. Efficient HPSG Parsing with Supertagging and CFG-filtering. In *Proceedings of IJCAI-07*, pages 1671–1676.

Miyao, Yusuke. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 83–90.

Miyao, Yusuke. 2006. *From Linguistic Theory to Syntactic Analysis: Corpus-Oriented Grammar Development and Feature Forest Model*. Ph.D. Dissertation, The University of Tokyo.

Ninomiya, Takashi, Yoshimasa Tsuruoka, Takuya Matsuzaki, and Yusuke Miyao. 2006. Extremely lexicalized models for accurate and fast HPSG parsing. In *Proceedings of EMNLP-2006*, pages 155–163.

Ninomiya, T., T. Matsuzaki, Y. Miyao, and J. Tsujii. 2007. A log-linear model with an n-gram reference distribution for accurate HPSG parsing. *In Proceedings of IWPT-2007*, pages 60–68.

Pollard, Carl and Ivan A. Sag. 1994. *Head-driven Phrase Structure Grammar*. University of Chicago / CSLI.

Prins, R. and G. Van Noord. 2001. Unsupervised Pos-Tagging Improves Parsing Accuracy And Parsing Efficiency. In *Proceedings of IWPT-2001*, pages 154–165.

Zhang, Yao-zhong, Takuya Matsuzaki, and Jun'ichi Tsujii. 2009. HPSG Supertagging: A Sequence Labeling View. In *Proceedings of IWPT-2009*, pages 210–213.