A Simple Approach for HPSG Supertagging Using Dependency Information

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, UK
{yaozhong.zhang, matuzaki, tsujii}@is.s.u-tokyo.ac.jp

Abstract

In a supertagging task, sequence labeling models are commonly used. But their limited ability to model long-distance information presents a bottleneck to make further improvements. In this paper, we modeled this long-distance information in dependency formalism and integrated it into the process of HPSG supertagging. The experiments showed that the dependency information is very informative for supertag disambiguation. We also evaluated the improved supertagger in the HPSG parser.

1 Introduction

Supertagging is a widely used speed-up technique for lexicalized grammar parsing. It was first proposed for lexicalized tree adjoining grammar (LTAG) (Bangalore and Joshi, 1999), then extended to combinatory categorial grammar (CCG) (Clark, 2002) and head-driven phrase structure grammar (HPSG) (Ninomiya et al., 2006). For deep parsing, supertagging is an important preprocessor: an accurate supertagger greatly reduces search space of a parser. Not limited to parsing, supertags can be used for NP chunking (Shen and Joshi, 2003), semantic role labeling (Chen and Rambow, 2003) and machine translation (Birch et al., 2007; Hassan et al., 2007) to explore rich syntactic information contained in them.

Generally speaking, supertags are lexical templates extracted from a grammar. These templates encode possible syntactic behavior of a word. Although the number of supertags is far larger than the 45 POS tags defined in Penn Treebank, sequence labeling techniques are still effective for supertagging. Previous research (Clark, 2002) showed that a POS sequence is very informative for supertagging, and some extent of local syntactic information can be captured by the context of surrounding words and POS tags. However, since the context window length is limited for the computational cost reasons, there are still long-range dependencies which are not easily captured in sequential models (Zhang et al., 2009). In practice, the multi-tagging technique proposed by Clark (2002) assigned more than one supertag to each word and let the ambiguous supertags be selected by the parser. As for other NLP applications which use supertags, resolving more supertag ambiguities in supertagging stage is preferred. With this consideration, we focus on supertagging and aim to make it as accurate as possible.

In this paper, we incorporated long-distance information into supertagging. First, we used dependency parser formalism to model long-distance relationships between the input words, which is hard to model in sequence labeling models. Then, we combined the dependency information with local context in a simple point-wise model. The experiments showed that dependency information is very informative for supertagging and we got a competitive 93.70% on supertagging accuracy (fed golden POS). In addition, we also evaluated the improved supertagger in the HPSG parser.

2 HPSG Supertagging and Dependency

2.1 HPSG Supertags

HPSG (Pollard and Sag, 1994) is a lexicalist grammar framework. In HPSG, a large number of lexical entries is used to describe word-specific syntactic characteristics, while only a small number of schemas is used to explain general construction rules. These lexical entries are called "HPSG supertags". For example, one possible supertag for the word "like" is written like "[NP.nom<V.bse>NP.acc]_lxm", which indicates

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 645–648,
Los Angeles, California, June 2010.

the head syntactic category of "like" is verb in base form. It has a NP subject and a NP complement. With such fine-grained grammatical type distinctions, the number of supertags is much larger than the number of tags used in other sequence labeling tasks. The HPSG grammar used in our experiment includes 2,308 supertags. This increases computational cost of sequence labeling models.

2.2 Why Use Dependency in Supertagging

By analyzing the internal structure of the supertags, we found that subject and complements are two important syntactic properties for each supertag. If we could predict subject and complements of the word well, supertagging would be an easier job to do. However, current widely used sequence labeling models have the limited ability to catch these long-distance syntactic relations. In supertagging stage, tree structures are still not constructed. Dependency formalism is an alternative way to describe these two syntactic properties. Based on this observation, we think dependency information could assist supertag prediction.

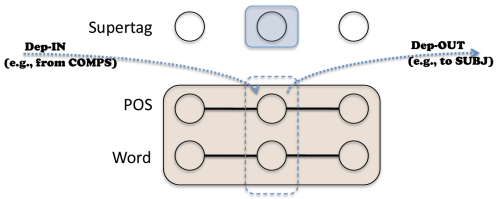

Figure 1: Model structure of incorporating dependency information into the supertagging stage. Dotted arrows describe the augmented long distance dependency information provided for supertag prediction.

3 Our Method

3.1 Modeling Dependency for Supertags

First of all, we need to characterize the dependency between words for supertagging. Since exact dependency locations are not encoded in supertags, to make use of state-of-the-art dependency parser, we recover HPSG supertag dependencies with the aid of HPSG treebanks. The dependencies are extracted from each branch in the HPSG trees by regarding the non-head daughter as the modifier of the head-daughter. HPSG schemas are expressed in dependency arcs.

To model the dependency, we follow mainstream dependency parsing formalism. Two representative methods for dependency parsing are transition-based model like MaltParser (Nivre, 2003) and graph-based model like MSTParser[1] (McDonald et al., 2005). Previous research (Nivre and McDonald, 2008) showed that MSTParser is more accurate than MaltParser for long dependencies. Since our motivation is to capture long-distance dependency as a complement for local supertagging models, we use the projective MSTParser formalism to model dependencies.

MOD-IN	$\{(p_i \Leftarrow p_j)\&s_j \vert (j,i) \in E\}$ $\{(p_i \Leftarrow w_j)\&s_j \vert (j,i) \in E\}$ $\{(w_i \Leftarrow p_j)\&s_j \vert (j,i) \in E\}$ $\{(w_i \Leftarrow w_j)\&s_j \vert (j,i) \in E\}$
MOD-OUT	$\{(p_i \Rightarrow p_j)\&s_i \vert (i,j) \in E\}$ $\{(p_i \Rightarrow w_j)\&s_i \vert (i,j) \in E\}$ $\{(w_i \Rightarrow p_j)\&s_i \vert (i,j) \in E\}$ $\{(w_i \Rightarrow w_j)\&s_i \vert (i,j) \in E\}$

Table 1: Non-local feature templates used for supertagging. Here, p, w and s represent POS, word and schema respectively. Direction (Left/Right) from MODIN/MODOUT word to the current word is also considered in the feature templates.

3.2 Integrating Dependency into Supertagging

There are several ways to combine long-distance dependency into supertagging. Integrating dependency information into training process would be more intuitive. Here, we use feature-based integration. The base model is a point-wise averaged perceptron (PW-AP) which has been shown very effective (Zhang et al., 2009). The improved model structure is described in Figure 1. The long-distance information is formalized as first-order dependency. For the word being predicted, we extract its modifiers (MODIN) and its head (MODOUT) (Table 1) based on first-order dependency arcs. Then MODIN and MODOUT relations are combined as features with local context for supertag prediction. To compare with previous work, the basic local context features are the same as in Matsuzaki et al. (2007).

[1] http://sourceforge.net/projects/mstparser/

4 Experiments

We evaluated dependency-informed supertagger (PW-DEP) both by supertag accuracy [2] and by a HPSG parser. The experiments were conducted on WSJ-HPSG treebank (Miyao, 2006). Sections 02-21 were used to train the dependency parser, the dependency-informed supertagger and the HPSG parser. Section 23 was used as the testing set. The evaluation metric for HPSG parser is the accuracy of predicate-argument relations in the parser's output, as in previous work (Sagae et al., 2007).

Model	Dep Acc%†	Acc%
PW-AP	/	91.14
PW-DEP	90.98	**92.18**
PW-AP (gold POS)	/	92.48
PW-DEP (gold POS)	92.05	**93.70**
	100	**97.43**

Table 2: Supertagging accuracy on section 23. (†) Dependencies are given by MSTParser evaluated with labeled accuracy. PW-AP is the baseline point-wise averaged perceptron model. PW-DEP is point-wise dependency-informed model. The automatically tagged POS tags were given by a maximum entropy tagger with 97.39% accuracy.

4.1 Results on Supertagging

We first evaluated the upper-bound of dependency-informed supertagging model, given gold standard first-order dependencies. As shown in Table 2, with such long-distance information supertagging accuracy can reach 97.43%. Comparing to point-wise model (PW-AP) which only used local context (92.48%), this absolute 4.95% gain indicated that dependency information is really informative for supertagging. When automatically predicted dependency relations were given, there still were absolute 1.04% (auto POS) and 1.22% (gold POS) improvements from baseline PW-AP model.

We also compared supertagging results with previous works (reported on section 22). Here we mainly compared the dependency-informed point-wise models with perceptron-based Bayes point machine (BPM) plus CFG-filter (Zhang et al., 2009). To the best of our knowledge, these are the state-of-the-art results on the same dataset with gold POS tags. CFG-filtering can be considered as an alternative way of incorporating long-distance constraints on supertagging results. Although our baseline system was slightly behind (PW-AP: 92.16% vs. BPM:92.53%), the final accuracies of grammatically constrained models were very close (PW-DEP: 93.53% vs. BPM-CFG: 93.60%); They were not statistically significantly different (P-value is 0.26). As the result of oracle PW-DEP indicated, supertagging accuracy can be further improved with better dependency modeling (e.g., with a semi-supervised dependency parser), which makes it more extensible and attractive than using CFG-filter after the supertagging process.

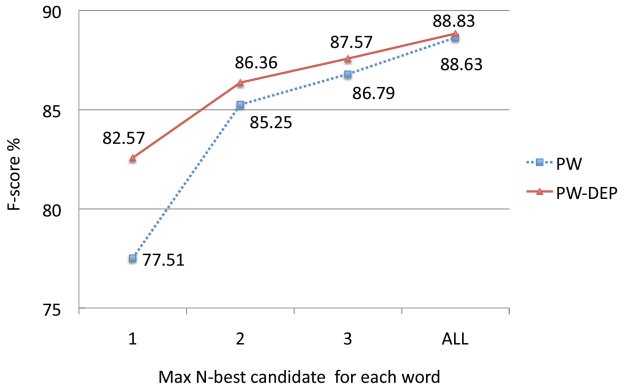

Figure 2: HPSG Parser F-score on section 23, given automatically tagged POS.

4.2 HPSG parsing results

We also evaluated the dependency-informed supertagger in a HPSG parser. Considering the efficiency, we use the HPSG parser[3] described by Matsuzaki et al. (2007).

In practice, several supertag candidates are reserved for each word to avoid parsing failure. To evaluate the quality of the two supertaggers, we restricted the number of each word's supertag candidates fed to the HPSG parser. As shown in Figure 2, for the case when only one supertag was predicted for each word, F-score of the HPSG parser using dependency-informed supertagger is 5.06% higher than the parser using the baseline supertagger module. As the candidate number increased, the gap narrowed: when all candidates were given, the gains gradually came down to 0.2%. This indicated that

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

[3] Enju v2.3.1, http://www-tsujii.is.s.u-tokyo.ac.jp/enju.

improved supertagger can optimize the search space of the deep parser, which may contribute to more accurate and fast deep parsing. From another aspect, supertagging can be viewed as an interface to combine different types of parsers.

As for the overall parsing time, we didn't optimize for speed in current setting. The parsing time[4] saved by using the improved supertagger (around 6.0 ms/sen, 21.5% time reduction) can not compensate for the extra cost of MSTParser (around 73.8 ms/sen) now. But there is much room to improve the final speed (e.g., optimizing the dependency parser for speed or reusing acquired dependencies for effective pruning). In addition, small beam-size can be "safely" used with improved supertagger for speed.

Using shallow dependencies in deep HPSG parsing has been previously explored by Sagae et al. (2007), who used dependency constraints in schema application stage to guide HPSG tree construction (F-score was improved from 87.2% to 87.9% with a single shift-reduce dependency parser). Since the baseline parser is different, we didn't make a direct comparison here. However, it would be interesting to compare these two different ways of incorporating the dependency parser into HPSG parsing. We left it as further work.

5 Conclusions

In this paper, focusing on improving the accuracy of supertagging, we proposed a simple but effective way to incorporate long-distance dependency relations into supertagging. The experiments mainly showed that these long-distance dependencies, which are not easy to model in traditional sequence labeling models, are very informative for supertag predictions. Although these were preliminary results, the method shows its potential strength for related applications. Not limited to HPSG, it can be extended to other lexicalized grammar supertaggers.

Acknowledgments

Thanks to the anonymous reviewers for valuable comments. We also thank Goran Topic for his selfless help. 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).

[4]Tested on section 23 (2291 sentences) using an AMD Opteron 2.4GHz server, given all supertag candidates.

References

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

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

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

Stephen Clark. 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.

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

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2007. Efficient hpsg parsing with supertagging and cfg-filtering. In *Proceedings of IJCAI-07*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL-05*.

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

Takashi Ninomiya, 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.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*.

J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of IWPT-03*, pages 149–160. Citeseer.

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

Kenji Sagae, Yusuke Miyao, and Jun'ichi Tsujii. 2007. Hpsg parsing with shallow dependency constraints. In *Proceedings of ACL-07*.

Libin Shen and Aravind K. Joshi. 2003. A snow based supertagger with application to np chunking. In *Proceedings of ACL 2003*, pages 505–512.

Yao-zhong Zhang, Takuya Matsuzaki, and Jun'ichi Tsujii. 2009. Hpsg supertagging: A sequence labeling view. In *Proceedings of IWPT-09*, Paris, France.