Better Automatic Treebank Conversion Using A Feature-Based Approach

Muhua Zhu **Jingbo Zhu** **Minghan Hu**
Natural Language Processing Lab.
Northeastern University, China
zhumuhua@gmail.com
zhujingbo@mail.neu.edu.cn huminghan@ise.neu.edu.cn

Abstract

For the task of automatic treebank conversion, this paper presents a feature-based approach which encodes bracketing structures in a treebank into features to guide the conversion of this treebank to a different standard. Experiments on two Chinese treebanks show that our approach improves conversion accuracy by 1.31% over a strong baseline.

1 Introduction

In the field of syntactic parsing, research efforts have been put onto the task of automatic conversion of a treebank (*source treebank*) to fit a different standard which is exhibited by another treebank (*target treebank*). Treebank conversion is desirable primarily because source-style and target-style annotations exist for non-overlapping text samples so that a larger target-style treebank can be obtained through such conversion. Hereafter, source and target treebanks are named as heterogenous treebanks due to their different annotation standards. In this paper, we focus on the scenario of conversion between phrase-structure heterogeneous treebanks (Wang et al., 1994; Zhu and Zhu, 2010).

Due to the availability of annotation in a source treebank, it is natural to use such annotation to guide treebank conversion. The motivating idea is illustrated in Fig. 1 which depicts a sentence annotated with standards of Tsinghua Chinese Treebank (TCT) (Zhou, 1996) and Penn Chinese Treebank (CTB) (Xue et al., 2002), respectively. Suppose that the conversion is in the direction from the TCT-style parse (left side) to the CTB-style parse (right side). The constituents vp:[将/will 投降/surrender], dj:[敌人/enemy 将/will 投降/surrender], and np:[情报/intelligence 专家/experts] in the TCT-style parse strongly suggest a resulting CTB-style parse also bracket the words as constituents. Zhu and Zhu (2010) show the effectiveness of using bracketing structures in a source treebank (source-side bracketing structures in short) as parsing constraints during the decoding phase of a target treebank-based parser.

However, using source-side bracketing structures as parsing constraints is problematic in some cases. As illustrated in the shadow part of Fig. 1, the TCT-style parse takes "认为/deems" as the right boundary of a constituent while in the CTB-style parse, "认为" is the left boundary of a constituent. According to the criteria used in Zhu and Zhu (2010), any CTB-style constituents with "认为" being the left boundary are thought to be *inconsistent* with the bracketing structure of the TCT-style parse and will be pruned. However, if we prune such "inconsistent" constituents, the correct conversion result (right side of Fig. 1) has no chance to be generated.

The problem comes from binary distinctions used in the approach of Zhu and Zhu (2010). With binary distinctions, constituents generated by a target treebank-based parser are judged to be either consistent or inconsistent with source-side bracketing structures. That approach prunes inconsistent constituents which instead might be correct conversion results[1]. In this paper, we insist on using source-side bracketing structures as guiding information. Meanwhile, we aim to avoid using binary distinctions. To achieve such a goal, we propose to use a feature-based approach to treebank conversion and to encode source-side bracketing structures as a set

[1]To show how severe this problem might be, Section 3.1 presents statistics on inconsistence between TCT and CTB.

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:shortpapers, pages 715–719,
Portland, Oregon, June 19-24, 2011.

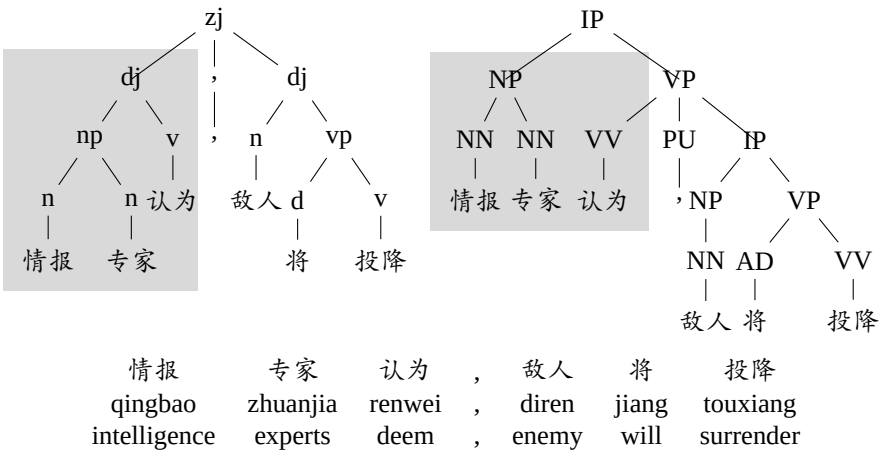

Figure 1: An example sentence with TCT-style annotation (left) and CTB-style annotation (right).

of features. The advantage is that inconsistent constituents can be scored with a function based on the features rather than ruled out as impossible.

To test the efficacy of our approach, we conduct experiments on conversion from TCT to CTB. The results show that our approach achieves a 1.31% absolute improvement in conversion accuracy over the approach used in Zhu and Zhu (2010).

2 Our Approach

2.1 Generic System Architecture

To conduct treebank conversion, our approach, overall speaking, proceeds in the following steps.

Step 1: Build a parser (named *source parser*) on a source treebank, and use it to parse sentences in the training data of a target treebank.

Step 2: Build a parser on pairs of golden target-style and auto-assigned (in Step 1) source-style parses in the training data of the target treebank. Such a parser is named *heterogeneous parser* since it incorporates information derived from both source and target treebanks, which follow different annotation standards.

Step 3: In the testing phase, the heterogeneous parser takes golden source-style parses as input and conducts treebank conversion. This will be explained in detail in Section 2.2.

To instantiate the generic framework described above, we need to decide the following three factors: (1) a parsing model for building a source parser, (2) a parsing model for building a heterogeneous parser, and (3) features for building a heterogeneous parser. In principle, any off-the-shelf parsers can be used to build a source parser, so we focus only on the latter two factors. To build a heterogeneous parser, we use feature-based parsing algorithms in order to easily incorporate features that encode source-side bracketing structures. Theoretically, any feature-based approaches are applicable, such as Finkel et al. (2008) and Tsuruoka et al. (2009). In this paper, we use the shift-reduce parsing algorithm for its simplicity and competitive performance.

2.2 Shift-Reduce-Based Heterogeneous Parser

The heterogeneous parser used in this paper is based on the shift-reduce parsing algorithm described in Sagae and Lavie (2006a) and Wang et al. (2006). Shift-reduce parsing is a state transition process, where a state is defined to be a tuple $\langle S, Q \rangle$. Here, S is a stack containing partial parses, and Q is a queue containing word-POS pairs to be processed. At each state transition, a shift-reduce parser either *shifts* the top item of Q onto S, or *reduces* the top one (or two) items on S.

A shift-reduce-based heterogeneous parser proceeds similarly as the standard shift-reduce parsing algorithm. In the training phase, each target-style parse tree in the training data is transformed into a binary tree (Charniak et al., 1998) and then decomposed into a (golden) action-state sequence. A classifier can be trained on the set of action-states,

where each state is represented as a feature vector. In the testing phase, the trained classifier is used to choose actions for state transition. Moreover, beam search strategies can be used to expand the search space of a shift-reduce-based heterogeneous parser (Sagae and Lavie, 2006a). To incorporate information on source-side bracketing structures, in both training and testing phases, feature vectors representing states $\langle S, Q \rangle$ are augmented with features that bridge the current state and the corresponding source-style parse.

2.3 Features

This section describes the feature functions used to build a heterogeneous parser on the training data of a target treebank. The features can be divided into two groups. The first group of features are derived solely from target-style parse trees so they are referred to as *target side features*. This group of features are completely identical to those used in Sagae and Lavie (2006a).

In addition, we have features extracted jointly from target-style and source-style parse trees. These features are generated by consulting a source-style parse (referred to as t_s) while we decompose a target-style parse into an action-state sequence. Here, s_i denote the i_{th} item from the top of the stack, and q_i denote the i_{th} item from the front end of the queue. We refer to these features as *heterogeneous features*.

Constituent features $F_c(s_i, t_s)$
This feature schema covers three feature functions: $F_c(s_1, t_s)$, $F_c(s_2, t_s)$, and $F_c(s_1 \circ s_2, t_s)$, which decide whether partial parses on stack S correspond to a constituent in the source-style parse t_s. That is, $F_c(s_i, t_s) = +$ if s_i has a bracketing match (ignoring grammar labels) with any constituent in t_s. $s_1 \circ s_2$ represents a concatenation of spans of s_1 and s_2.

Relation feature $F_r(N_s(s_1), N_s(s_2))$
We first position the lowest node $N_s(s_i)$ in t_s, which dominates the span of s_i. Then a feature function $F_r(N_s(s_1), N_s(s_2))$ is defined to indicate the relationship of $N_s(s_1)$ and $N_s(s_2)$. If $N_s(s_1)$ is identical to or a sibling of $N_s(s_2)$, we say $F_r(N_s(s_1), N_s(s_2)) = +$.

Features Bridging Source and Target Parses
$F_c(s_1, t_s) = -$
$F_c(s_2, t_s) = +$
$F_c(s_1 \circ s_2, t_s) = +$
$F_r(N_s(s_1), N_s(s_2)) = -$
$F_f(RF(s_1), q_1) = -$
$F_p(RF(s_1), q_1) =$ "$v \uparrow dj \uparrow zj \downarrow$,"

Table 1: An example of new features. Suppose we are considering the sentence depicted in Fig. 1.

Frontier-words feature $F_f(RF(s_1), q_1)$
A feature function which decides whether the right frontier word of s_1 and q_1 are in the same base phrase in t_s. Here, a base phrase is defined to be any phrase which dominates no other phrases.

Path feature $F_p(RF(s_1), q_1)$
Syntactic path features are widely used in the literature of semantic role labeling (Gildea and Jurafsky, 2002) to encode information of both structures and grammar labels. We define a string-valued feature function $F_p(RF(s_1), q_1)$ which connects the right frontier word of s_1 to q_1 in t_s.

To better understand the above feature functions, we re-examine the example depicted in Fig. 1. Suppose that we use a shift-reduce-based heterogeneous parser to convert the TCT-style parse to the CTB-style parse and that stack S currently contains two partial parses: s_2:[NP (NN 情报) (NN 专家)] and s_1: (VV 认为). In such a state, we can see that spans of both s_2 and $s_1 \circ s_2$ correspond to constituents in t_s but that of s_1 does not. Moreover, $N_s(s_1)$ is *dj* and $N_s(s_2)$ is *np*, so $N_s(s_1)$ and $N_s(s_2)$ are neither identical nor sisters in t_s. The values of these features are collected in Table 1.

3 Experiments

3.1 Data Preparation and Performance Metric

In the experiments, we use two heterogeneous treebanks: CTB 5.1 and the TCT corpus released by the CIPS-SIGHAN-2010 syntactic parsing competition[2]. We actually only use the training data of these two corpora, that is, articles 001-270 and 400-1151 (18,100 sentences, 493,869 words) of CTB 5.1 and

[2]http://www.cipsc.org.cn/clp2010/task2_en.htm

the training data (17,529 sentences, 481,061 words) of TCT.

To evaluate conversion accuracy, we use the same test set (named *Sample-TCT*) as in Zhu and Zhu (2010), which is a set of 150 sentences with manually assigned CTB-style and TCT-style parse trees. In Sample-TCT, 6.19% (215/3473) CTB-style constituents are inconsistent with respect to the TCT standard and 8.87% (231/2602) TCT-style constituents are inconsistent with respect to the CTB standard.

For all experiments, *bracketing F1* is used as the performance metric, provided by *EVALB* [3].

3.2 Implementation Issues

To implement a heterogeneous parser, we first build a Berkeley parser (Petrov et al., 2006) on the TCT training data and then use it to assign TCT-style parses to sentences in the CTB training data. On the "updated" CTB training data, we build two shift-reduce-based heterogeneous parsers by using maximum entropy classification model, without/with beam search. Hereafter, the two heterogeneous parsers are referred to as *Basic-SR* and *Beam-SR*, respectively.

In the testing phase, Basic-SR and Beam-SR convert TCT-style parse trees in Sample-TCT to the CTB standard. The conversion results are evaluated against corresponding CTB-style parse trees in Sample-TCT. Before conducting treebank conversion, we apply the POS adaptation method proposed in Jiang et al. (2009) to convert TCT-style POS tags in the input to the CTB standard. The POS conversion accuracy is 96.2% on Sample-TCT.

3.3 Results

Table 2 shows the results achieved by Basic-SR and Beam-SR with heterogeneous features being added incrementally. Here, baseline represents the systems which use only target side features. From the table we can see that heterogeneous features improve conversion accuracy significantly. Specifically, adding the *constituent (F_c)* features to Basic-SR (Beam-SR) achieves a 2.79% (3%) improvement, adding the *relation (F_r)* and *frontier-word (F_f)* features yields a 0.79% (0.98%) improvement, and adding the *path (F_p)* feature achieves a 0.14% (0.13%) improvement. The path feature is not so effective as expected, although it manages to achieve improvements. One possible reason lies on the data sparseness problem incurred by this feature.

System	Features	$<= 40$ words	Unlimited
Basic-SR	baseline	83.34	80.33
	$+F_c$	85.89	83.12
	$+F_r$, $+F_f$	85.47	83.91
	$+F_p$	86.01	84.05
Beam-SR	baseline	84.40	81.27
	$+F_c$	86.30	84.27
	$+F_r$, $+F_f$	87.00	85.25
	$+F_p$	87.27	85.38

Table 2: Adding new features to baselines improve treebank conversion accuracy significantly on Sample-TCT.

Since we use the same training and testing data as in Zhu and Zhu (2010), we can compare our approach directly with the informed decoding approach used in that work. We find that Basic-SR achieves very close conversion results (84.05% vs. 84.07%) and Beam-SR even outperforms the informed decoding approach (85.38% vs. 84.07%) with a 1.31% absolute improvement.

4 Related Work

For phrase-structure treebank conversion, Wang et al. (1994) suggest to use source-side bracketing structures to select conversion results from k-best lists. The approach is quite generic in the sense that it can be used for conversion between treebanks of different grammar formalisms, such as from a dependency treebank to a constituency treebank (Niu et al., 2009). However, it suffers from limited variations in k-best lists (Huang, 2008). Zhu and Zhu (2010) propose to incorporate bracketing structures as parsing constraints in the decoding phase of a CKY-style parser. Their approach shows significant improvements over Wang et al. (1994). However, it suffers from binary distinctions (consistent or inconsistent), as discussed in Section 1.

The approach in this paper is reminiscent of co-training (Blum and Mitchell, 1998; Sagae and Lavie, 2006b) and up-training (Petrov et al., 2010). Moreover, it coincides with the stacking method used for dependency parser combination (Martins

[3] http://nlp.cs.nyu.edu/evalb

et al., 2008; Nivre and McDonald, 2008), the Pred method for domain adaptation (Daumé III and Marcu, 2006), and the method for annotation adaptation of word segmentation and POS tagging (Jiang et al., 2009). As one of the most related works, Jiang and Liu (2009) present a similar approach to conversion between dependency treebanks. In contrast to Jiang and Liu (2009), the task studied in this paper, phrase-structure treebank conversion, is relatively complicated and more efforts should be put into feature engineering.

5 Conclusion

To avoid binary distinctions used in previous approaches to automatic treebank conversion, we proposed in this paper a feature-based approach. Experiments on two Chinese treebanks showed that our approach outperformed the baseline system (Zhu and Zhu, 2010) by 1.31%.

Acknowledgments

We thank Kenji Sagae for helpful discussions on the implementation of shift-reduce parser and the three anonymous reviewers for comments. This work was supported in part by the National Science Foundation of China (60873091; 61073140), Specialized Research Fund for the Doctoral Program of Higher Education (20100042110031), the Fundamental Research Funds for the Central Universities and Natural Science Foundation of Liaoning Province of China.

References

Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of COLT 1998*.

Eugene Charniak, Sharon Goldwater, and Mark Johnson. 1998. Edge-Based Best-First Chart Parsing. In *Proceedings of the Six Workshop on Very Large Corpora*, pages 127-133.

Hal Daumé III and Daniel Marcu. 2006. Domain Adaptation for Statistical Classifiers. *Journal of Artifical Intelligence Research*, 26:101-166.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, Feature-Based Conditional Random Fileds Parsing. In *Proceedings of ACL 2008*, pages 959-967.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling for Semantic Roles. *Computational Linguistics*, 28(3):245-288.

Liang Huang. 2008. Forest Reranking: Discriminative Parsing with Non-local Features. In *Proceedings of ACL*, pages 824-831.

Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic Adaptation of Annotation Standards: Chinese Word Segmentation and POS Tagging - A Case Study. In *Proceedings of ACL 2009*, pages 522-530.

Wenbin Jiang and Qun Liu. 2009. Automatic Adaptation of Annotation Standards for Dependency Parsing – Using Projected Treebank As Source Corpus. In *Proceedings of IWPT 2009*, pages 25-28.

André F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stack Dependency Parsers. In *Proceedings of EMNLP 2008*, pages 157-166.

Zheng-Yu Niu, Haifeng Wang, and Hua Wu. 2009. Exploiting Heterogeneous Treebanks for Parsing. In *Proceedings of ACL 2009*, pages 46-54.

Joakim Nivre and Ryan McDonald. 2008. Integrating Graph-Based and Transition-Based Dependency Parsers. In *Proceedings of ACL 2008*, pages 950-958.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of ACL 2006*, pages 433-440.

Slav Petrov, Pi-Chuan Chang, Michael Ringgaard, and Hiyan Alshawi. 2010. Uptraining for Accurate Deterministic Question Parsing. In *Proceedings of EMNLP 2010*, pages 705-713.

Kenji Sagae and Alon Lavie. 2006. A Best-First Probabilistic Shift-Reduce Parser. In *Proceedings of ACL-COLING 2006*, pages 691-698.

Kenji Sagae and Alon Lavie. 2006. Parser Combination by Reparsing. In *Proceedings of NAACL 2006*, pages 129-132.

Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. 2009. Fast Full Parsing by Linear-Chain Conditional Random Fields. In *Proceedings of EACL 2009*, pages 790-798.

Jong-Nae Wang, Jing-Shin Chang, and Keh-Yih Su. 1994. *An Automatic Treebank Conversion Algorithm for Corpus Sharing*. In *Proceedings of ACL 1994*, pages 248-254.

Mengqiu Wang, Kenji Sagae, and Teruk Mitamura. 2006. A Fast, Deterministic Parser for Chinese. In *Proceedings of ACL-COLING 2006*, pages 425-432.

Nianwen Xue, Fu dong Chiou, and Martha Palmer. 2002. Building a Large-Scale Annotated Chinese Corpus. In *Proceedings of COLING 2002*, pages 1-8.

Qiang Zhou. 1996. Phrase Bracketing and Annotation on Chinese Language Corpus (in Chinese). Ph.D. thesis, Peking University.

Muhua Zhu, and Jingbo Zhu. 2010. Automatic Treebank Conversion via Informed Decoding. In *Porceedings of COLING 2010*, pages 1541-1549.