Exploiting Heterogeneous Treebanks for Parsing

Zheng-Yu Niu, Haifeng Wang, Hua Wu
Toshiba (China) Research and Development Center
5/F., Tower W2, Oriental Plaza, Beijing, 100738, China
{niuzhengyu,wanghaifeng,wuhua}@rdc.toshiba.com.cn

Abstract

We address the issue of using heterogeneous treebanks for parsing by breaking it down into two sub-problems, converting grammar formalisms of the treebanks to the same one, and parsing on these homogeneous treebanks. First we propose to employ an iteratively trained target grammar parser to perform grammar formalism conversion, eliminating predefined heuristic rules as required in previous methods. Then we provide two strategies to refine conversion results, and adopt a corpus weighting technique for parsing on homogeneous treebanks. Results on the Penn Treebank show that our conversion method achieves 42% error reduction over the previous best result. Evaluation on the Penn Chinese Treebank indicates that a converted dependency treebank helps constituency parsing and the use of unlabeled data by self-training further increases parsing f-score to 85.2%, resulting in 6% error reduction over the previous best result.

1 Introduction

The last few decades have seen the emergence of multiple treebanks annotated with different grammar formalisms, motivated by the diversity of languages and linguistic theories, which is crucial to the success of statistical parsing (Abeille et al., 2000; Brants et al., 1999; Bohmova et al., 2003; Han et al., 2002; Kurohashi and Nagao, 1998; Marcus et al., 1993; Moreno et al., 2003; Xue et al., 2005). Availability of multiple treebanks creates a scenario where we have a treebank annotated with one grammar formalism, and another treebank annotated with another grammar formalism that we are interested in. We call the first a source treebank, and the second a target treebank. We thus encounter a problem of how to use these heterogeneous treebanks for target grammar parsing. Here heterogeneous treebanks refer to two or more treebanks with different grammar formalisms, e.g., one treebank annotated with dependency structure (DS) and the other annotated with phrase structure (PS).

It is important to acquire additional labeled data for the target grammar parsing through exploitation of existing source treebanks since there is often a shortage of labeled data. However, to our knowledge, there is no previous study on this issue.

Recently there have been some works on using multiple treebanks for domain adaptation of parsers, where these treebanks have the same grammar formalism (McClosky et al., 2006b; Roark and Bacchiani, 2003). Other related works focus on converting one grammar formalism of a treebank to another and then conducting studies on the converted treebank (Collins et al., 1999; Forst, 2003; Wang et al., 1994; Watkinson and Manandhar, 2001). These works were done either on multiple treebanks with the same grammar formalism or on only one converted treebank. We see that their scenarios are different from ours as we work with multiple heterogeneous treebanks.

For the use of heterogeneous treebanks[1], we propose a two-step solution: (1) converting the grammar formalism of the source treebank to the target one, (2) refining converted trees and using them as additional training data to build a target grammar parser.

For grammar formalism conversion, we choose the DS to PS direction for the convenience of the comparison with existing works (Xia and Palmer, 2001; Xia et al., 2008). Specifically, we assume that the source grammar formalism is dependency

[1]Here we assume the existence of two treebanks.

grammar, and the target grammar formalism is phrase structure grammar.

Previous methods for DS to PS conversion (Collins et al., 1999; Covington, 1994; Xia and Palmer, 2001; Xia et al., 2008) often rely on predefined heuristic rules to eliminate converison ambiguity, e.g., minimal projection for dependents, lowest attachment position for dependents, and the selection of conversion rules that add fewer number of nodes to the converted tree. In addition, the validity of these heuristic rules often depends on their target grammars. To eliminate the heuristic rules as required in previous methods, we propose to use an existing target grammar parser (trained on the target treebank) to generate N-best parses for each sentence in the source treebank as conversion candidates, and then select the parse consistent with the structure of the source tree as the converted tree. Furthermore, we attempt to use converted trees as additional training data to retrain the parser for better conversion candidates. The procedure of tree conversion and parser retraining will be run iteratively until a stopping condition is satisfied.

Since some converted trees might be imperfect from the perspective of the target grammar, we provide two strategies to refine conversion results: (1) pruning low-quality trees from the converted treebank, (2) interpolating the scores from the source grammar and the target grammar to select better converted trees. Finally we adopt a corpus weighting technique to get an optimal combination of the converted treebank and the existing target treebank for parser training.

We have evaluated our conversion algorithm on a dependency structure treebank (produced from the Penn Treebank) for comparison with previous work (Xia et al., 2008). We also have investigated our two-step solution on two existing treebanks, the Penn Chinese Treebank (CTB) (Xue et al., 2005) and the Chinese Dependency Treebank (CDT)[2] (Liu et al., 2006). Evaluation on WSJ data demonstrates that it is feasible to use a parser for grammar formalism conversion and the conversion benefits from converted trees used for parser retraining. Our conversion method achieves 93.8% f-score on dependency trees produced from WSJ section 22, resulting in 42% error reduction over the previous best result for DS to PS conversion. Results on CTB show that score interpolation is more effective than instance pruning for the use of converted treebanks for parsing and converted CDT helps parsing on CTB. When coupled with self-training technique, a reranking parser with CTB and converted CDT as labeled data achieves 85.2% f-score on CTB test set, an absolute 1.0% improvement (6% error reduction) over the previous best result for Chinese parsing.

[2]Available at http://ir.hit.edu.cn/.

The rest of this paper is organized as follows. In Section 2, we first describe a parser based method for DS to PS conversion, and then we discuss possible strategies to refine conversion results, and finally we adopt the corpus weighting technique for parsing on homogeneous treebanks. Section 3 provides experimental results of grammar formalism conversion on a dependency treebank produced from the Penn Treebank. In Section 4, we evaluate our two-step solution on two existing heterogeneous Chinese treebanks. Section 5 reviews related work and Section 6 concludes this work.

2 Our Two-Step Solution

2.1 Grammar Formalism Conversion

Previous DS to PS conversion methods built a converted tree by iteratively attaching nodes and edges to the tree with the help of conversion rules and heuristic rules, based on current head-dependent pair from a source dependency tree and the structure of the built tree (Collins et al., 1999; Covington, 1994; Xia and Palmer, 2001; Xia et al., 2008). Some observations can be made on these methods: (1) for each head-dependent pair, only one locally optimal conversion was kept during tree-building process, at the risk of pruning globally optimal conversions, (2) heuristic rules are required to deal with the problem that one head-dependent pair might have multiple conversion candidates, and these heuristic rules are usually hand-crafted to reflect the structural preference in their target grammars. To overcome these limitations, we propose to employ a parser to generate N-best parses as conversion candidates and then use the structural information of source trees to select the best parse as a converted tree.

We formulate our conversion method as follows.

Let C_{DS} be a source treebank annotated with DS and C_{PS} be a target treebank annotated with PS. Our goal is to convert the grammar formalism of C_{DS} to that of C_{PS}.

We first train a constituency parser on C_{PS}

```
**Input**: $C_{PS}$, $C_{DS}$, $Q$, and a constituency parser      **Output**: Converted trees $C_{PS}^{DS}$
1. **Initialize**:
  — Set $C_{PS}^{DS,0}$ as null, $DevScore$=0, $q$=0;
  — Split $C_{PS}$ into training set $C_{PS,train}$ and development set $C_{PS,dev}$;
  — Train the parser on $C_{PS,train}$ and denote it by $P_{q-1}$;
2. **Repeat**:
  — Use $P_{q-1}$ to generate N-best PS parses for each sentence in $C_{DS}$, and convert PS to DS for each parse;
  — **For** each sentence in $C_{DS}$ **Do**
      $\diamond$ $\hat{t}$=$argmax_t Score(x_{i,t})$, and select the $\hat{t}$-th parse as a converted tree for this sentence;
  — Let $C_{PS}^{DS,q}$ represent these converted trees, and let $C_{train}$=$C_{PS,train} \bigcup C_{PS}^{DS,q}$;
  — Train the parser on $C_{train}$, and denote the updated parser by $P_q$;
  — Let $DevScore_q$ be the f-score of $P_q$ on $C_{PS,dev}$;
  — **If** $DevScore_q > DevScore$ **Then** $DevScore$=$DevScore_q$, and $C_{PS}^{DS}$=$C_{PS}^{DS,q}$;
  — **Else break**;
  — q++;
 **Until** $q > Q$
```

Table 1: Our algorithm for DS to PS conversion.

(90% trees in C_{PS} as training set $C_{PS,train}$, and other trees as development set $C_{PS,dev}$) and then let the parser generate N-best parses for each sentence in C_{DS}.

Let n be the number of sentences (or trees) in C_{DS} and n_i be the number of N-best parses generated by the parser for the i-th ($1 \leq i \leq n$) sentence in C_{DS}. Let $x_{i,t}$ be the t-th ($1 \leq t \leq n_i$) parse for the i-th sentence. Let y_i be the tree of the i-th ($1 \leq i \leq n$) sentence in C_{DS}.

To evaluate the quality of $x_{i,t}$ as a conversion candidate for y_i, we convert $x_{i,t}$ to a dependency tree (denoted as $x_{i,t}^{DS}$) and then use unlabeled dependency f-score to measure the similarity between $x_{i,t}^{DS}$ and y_i. Let $Score(x_{i,t})$ denote the unlabeled dependency f-score of $x_{i,t}^{DS}$ against y_i. Then we determine the converted tree for y_i by maximizing $Score(x_{i,t})$ over the N-best parses.

The conversion from PS to DS works as follows:

Step 1. Use a head percolation table to find the head of each constituent in $x_{i,t}$.

Step 2. Make the head of each non-head child depend on the head of the head child for each constituent.

Unlabeled dependency f-score is a harmonic mean of unlabeled dependency precision and unlabeled dependency recall. Precision measures how many head-dependent word pairs found in $x_{i,t}^{DS}$ are correct and recall is the percentage of head-dependent word pairs defined in the gold-standard tree that are found in $x_{i,t}^{DS}$. Here we do not take dependency tags into consideration for evaluation since they cannot be obtained without more sophisticated rules.

To improve the quality of N-best parses, we attempt to use the converted trees as additional training data to retrain the parser. The procedure of tree conversion and parser retraining can be run iteratively until a termination condition is satisfied. Here we use the parser's f-score on $C_{PS,dev}$ as a termination criterion. If the update of training data hurts the performance on $C_{PS,dev}$, then we stop the iteration.

Table 1 shows this DS to PS conversion algorithm. Q is an upper limit of the number of loops, and $Q \geq 0$.

2.2 Target Grammar Parsing

Through grammar formalism conversion, we have successfully turned the problem of using heterogeneous treebanks for parsing into the problem of parsing on homogeneous treebanks. Before using converted source treebank for parsing, we present two strategies to refine conversion results.

Instance Pruning For some sentences in C_{DS}, the parser might fail to generate high quality N-best parses, resulting in inferior converted trees. To clean the converted treebank, we can remove the converted trees with low unlabeled dependency f-scores (defined in Section 2.1) before using the converted treebank for parser training

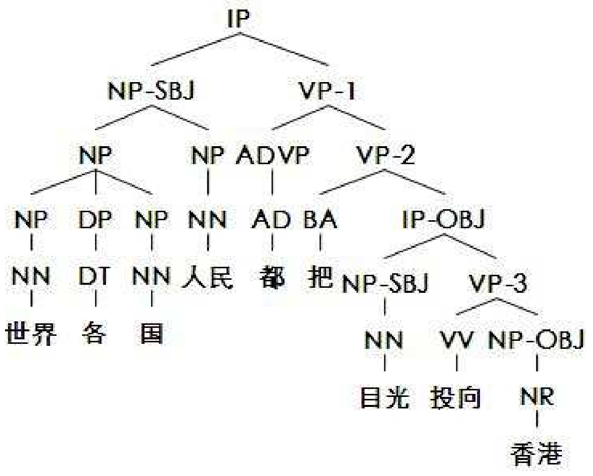

Figure 1: A parse tree in CTB for a sentence of “世界<world> 各<every> 国<country> 人民<people> 都<all> 把<with> 目光<eyes> 投向<cast> 香港<Hong Kong>” with “People from all over the world are casting their eyes on Hong Kong” as its English translation.

because these trees are “misleading” training instances. The number of removed trees will be determined by cross validation on development set.

Score Interpolation Unlabeled dependency f-scores used in Section 2.1 measure the quality of converted trees from the perspective of the source grammar only. In extreme cases, the top best parses in the N-best list are good conversion candidates but we might select a parse ranked quite low in the N-best list since there might be conflicts of syntactic structure definition between the source grammar and the target grammar.

Figure 1 shows an example for illustration of a conflict between the grammar of CDT and that of CTB. According to Chinese head percolation tables used in the PS to DS conversion tool “Penn2Malt”[3] and Charniak's parser[4], the head of VP-2 is the word “把” (a preposition, with “BA” as its POS tag in CTB), and the head of IP-OBJ is 投向”. Therefore the word “投向” depends on the word “把”. But according to the annotation scheme in CDT (Liu et al., 2006), the word “把” is a dependent of the word “投向”. The conflicts between the two grammars may lead to the problem that the selected parses based on the information of the source grammar might not be preferred from the perspective of the target grammar.

Therefore we modified the selection metric in Section 2.1 by interpolating two scores, the probability of a conversion candidate from the parser and its unlabeled dependency f-score, shown as follows:

$$\widehat{Score}(x_{i,t}) = \lambda \times Prob(x_{i,t}) + (1-\lambda) \times Score(x_{i,t}). \quad (1)$$

The intuition behind this equation is that converted trees should be preferred from the perspective of both the source grammar and the target grammar. Here $0 \leq \lambda \leq 1$. $Prob(x_{i,t})$ is a probability produced by the parser for $x_{i,t}$ ($0 \leq Prob(x_{i,t}) \leq 1$). The value of λ will be tuned by cross validation on development set.

After grammar formalism conversion, the problem now we face has been limited to how to build parsing models on multiple homogeneous treebank. A possible solution is to simply concatenate the two treebanks as training data. However this method may lead to a problem that if the size of C_{PS} is significantly less than that of converted C_{DS}, converted C_{DS} may weaken the effect C_{PS} might have. One possible solution is to reduce the weight of examples from converted C_{DS} in parser training. Corpus weighting is exactly such an approach, with the weight tuned on development set, that will be used for parsing on homogeneous treebanks in this paper.

3 Experiments of Grammar Formalism Conversion

3.1 Evaluation on WSJ section 22

Xia et al. (2008) used WSJ section 19 from the Penn Treebank to extract DS to PS conversion rules and then produced dependency trees from WSJ section 22 for evaluation of their DS to PS conversion algorithm. They showed that their conversion algorithm outperformed existing methods on the WSJ data. For comparison with their work, we conducted experiments in the same setting as theirs: using WSJ section 19 (1844 sentences) as C_{PS}, producing dependency trees from WSJ section 22 (1700 sentences) as C_{DS}[5], and using labeled bracketing f-scores from the tool “EVALB” on WSJ section 22 for performance evaluation.

[3]Available at http://w3.msi.vxu.se/∼nivre/.

[4]Available at http://www.cs.brown.edu/∼ec/.

[5]We used the tool “Penn2Malt” to produce dependency structures from the Penn Treebank, which was also used for PS to DS conversion in our conversion algorithm.

Models	DevScore (%)	All the sentences LR (%)	LP (%)	F (%)
The best result of Xia et al. (2008)	-	90.7	88.1	89.4
Q-0-method	86.8	92.2	92.8	92.5
Q-10-method	88.0	93.4	94.1	93.8

Table 2: Comparison with the work of Xia et al. (2008) on WSJ section 22.

Models	DevScore (%)	All the sentences LR (%)	LP (%)	F (%)
Q-0-method	91.0	91.6	92.5	92.1
Q-10-method	91.6	93.1	94.1	93.6

Table 3: Results of our algorithm on WSJ section 2$\sim$18 and 20$\sim$22.

We employed Charniak's maximum entropy inspired parser (Charniak, 2000) to generate N-best (N=200) parses. Xia et al. (2008) used POS tag information, dependency structures and dependency tags in test set for conversion. Similarly, we used POS tag information in the test set to restrict search space of the parser for generation of better N-best parses.

We evaluated two variants of our DS to PS conversion algorithm:

Q-0-method: We set the value of Q as 0 for a baseline method.

Q-10-method: We set the value of Q as 10 to see whether it is helpful for conversion to retrain the parser on converted trees.

Table 2 shows the results of our conversion algorithm on WSJ section 22. In the experiment of Q-10-method, DevScore reached the highest value of 88.0% when q was 1. Then we used $C_{PS}^{DS,1}$ as the conversion result. Finally Q-10-method achieved an f-score of 93.8% on WSJ section 22, an absolute 4.4% improvement (42% error reduction) over the best result of Xia et al. (2008). Moreover, Q-10-method outperformed Q-0-method on the same test set. These results indicate that it is feasible to use a parser for DS to PS conversion and the conversion benefits from the use of converted trees for parser retraining.

3.2 Evaluation on WSJ section 2$\sim$18 and 20$\sim$22

In this experiment we evaluated our conversion algorithm on a larger test set, WSJ section 2$\sim$18 and 20$\sim$22 (totally 39688 sentences). Here we also used WSJ section 19 as C_{PS}. Other settings for this experiment are as same as that in Section 3.1, except that here we used a larger test set.

Training data	All the sentences LR (%)	LP (%)	F (%)
$1 \times CTB + CDT^{PS}$	84.7	85.1	84.9
$2 \times CTB + CDT^{PS}$	85.1	85.6	85.3
$5 \times CTB + CDT^{PS}$	85.0	85.5	85.3
$10 \times CTB + CDT^{PS}$	85.3	85.8	85.6
$20 \times CTB + CDT^{PS}$	85.1	85.3	85.2
$50 \times CTB + CDT^{PS}$	84.9	85.3	85.1

Table 4: Results of the generative parser on the development set, when trained with various weighting of CTB training set and CDT^{PS}.

Table 3 provides the f-scores of our method with Q equal to 0 or 10 on WSJ section 2$\sim$18 and 20$\sim$22.

With Q-10-method, DevScore reached the highest value of 91.6% when q was 1. Finally Q-10-method achieved an f-score of 93.6% on WSJ section 2$\sim$18 and 20$\sim$22, better than that of Q-0-method and comparable with that of Q-10-method in Section 3.1. It confirms our previous finding that the conversion benefits from the use of converted trees for parser retraining.

4 Experiments of Parsing

We investigated our two-step solution on two existing treebanks, CDT and CTB, and we used CDT as the source treebank and CTB as the target treebank.

CDT consists of 60k Chinese sentences, annotated with POS tag information and dependency structure information (including 28 POS tags, and 24 dependency tags) (Liu et al., 2006). We did not use POS tag information as inputs to the parser in our conversion method due to the difficulty of conversion from CDT POS tags to CTB POS tags.

We used a standard split of CTB for performance evaluation, articles 1-270 and 400-1151 as training set, articles 301-325 as development set, and articles 271-300 as test set.

We used Charniak's maximum entropy inspired parser and their reranker (Charniak and Johnson, 2005) for target grammar parsing, called a generative parser (GP) and a reranking parser (RP) respectively. We reported ParseVal measures from the EVALB tool.

		All the sentences		
Models	Training data	LR (%)	LP (%)	F (%)
GP	CTB	79.9	82.2	81.0
RP	CTB	82.0	84.6	83.3
GP	$10 \times CTB + CDT^{PS}$	80.4	82.7	81.5
RP	$10 \times CTB + CDT^{PS}$	82.8	84.7	83.8

Table 5: Results of the generative parser (GP) and the reranking parser (RP) on the test set, when trained on only CTB training set or an optimal combination of CTB training set and CDT^{PS}.

4.1 Results of a Baseline Method to Use CDT

We used our conversion algorithm[6] to convert the grammar formalism of CDT to that of CTB. Let CDT^{PS} denote the converted CDT by our method. The average unlabeled dependency f-score of trees in CDT^{PS} was 74.4%, and their average index in 200-best list was 48.

We tried the corpus weighting method when combining CDT^{PS} with CTB training set (abbreviated as CTB for simplicity) as training data, by gradually increasing the weight (including 1, 2, 5, 10, 20, 50) of CTB to optimize parsing performance on the development set. Table 4 presents the results of the generative parser with various weights of CTB on the development set. Considering the performance on the development set, we decided to give CTB a relative weight of 10.

Finally we evaluated two parsing models, the generative parser and the reranking parser, on the test set, with results shown in Table 5. When trained on CTB only, the generative parser and the reranking parser achieved f-scores of 81.0% and 83.3%. The use of CDT^{PS} as additional training data increased f-scores of the two models to 81.5% and 83.8%.

4.2 Results of Two Strategies for a Better Use of CDT

4.2.1 Instance Pruning

We used unlabeled dependency f-score of each converted tree as the criterion to rank trees in CDT^{PS} and then kept only the top M trees with high f-scores as training data for parsing, resulting in a corpus CDT$^{PS}_{M}$. M varied from $100\%\times$|CDT^{PS}| to $10\%\times$|CDT^{PS}| with $10\%\times$|CDT^{PS}| as the interval. |CDT^{PS}|

[6]The setting for our conversion algorithm in this experiment was as same as that in Section 3.1. In addition, we used CTB training set as $C_{PS,train}$, and CTB development set as $C_{PS,dev}$.

		All the sentences		
Models	Training data	LR (%)	LP (%)	F (%)
GP	$CTB + CDT^{PS}_{\lambda}$	81.4	82.8	82.1
RP	$CTB + CDT^{PS}_{\lambda}$	83.0	85.4	84.2

Table 6: Results of the generative parser and the reranking parser on the test set, when trained on an optimal combination of CTB training set and converted CDT.

is the number of trees in CDT^{PS}. Then we tuned the value of M by optimizing the parser's performance on the development set with 10$\times$CTB+CDT$^{PS}_{M}$ as training data. Finally the optimal value of M was $100\%\times$|CDT|. It indicates that even removing very few converted trees hurts the parsing performance. A possible reason is that most of non-perfect parses can provide useful syntactic structure information for building parsing models.

4.2.2 Score Interpolation

We used $\widehat{Score}(x_{i,t})$[7] to replace $Score(x_{i,t})$ in our conversion algorithm and then ran the updated algorithm on CDT. Let CDT$^{PS}_{\lambda}$ denote the converted CDT by this updated conversion algorithm. The values of λ (varying from 0.0 to 1.0 with 0.1 as the interval) and the CTB weight (including 1, 2, 5, 10, 20, 50) were simultaneously tuned on the development set[8]. Finally we decided that the optimal value of λ was 0.4 and the optimal weight of CTB was 1, which brought the best performance on the development set (an f-score of 86.1%). In comparison with the results in Section 4.1, the average index of converted trees in 200-best list increased to 2, and their average unlabeled dependency f-score dropped to 65.4%. It indicates that structures of converted trees become more consistent with the target grammar, as indicated by the increase of average index of converted trees, further away from the source grammar.

Table 6 provides f-scores of the generative parser and the reranker on the test set, when trained on CTB and CDT$^{PS}_{\lambda}$. We see that the performance of the reranking parser increased to

[7]Before calculating $\widehat{Score}(x_{i,t})$, we normalized the values of $Prob(x_{i,t})$ for each N-best list by (1) $Prob(x_{i,t})$=$Prob(x_{i,t})$-Min($Prob(x_{i,*})$), (2)$Prob(x_{i,t})$=$Prob(x_{i,t})$/Max($Prob(x_{i,*})$), resulting in that their maximum value was 1 and their minimum value was 0.

[8]Due to space constraint, we do not show f-scores of the parser with different values of λ and the CTB weight.

		All the sentences		
Models	Training data	LR (%)	LP (%)	F (%)
Self-trained GP	$10 \times T$+$10 \times D$+P	83.0	84.5	83.7
Updated RP	CTB+CDT_{λ}^{PS}	84.3	86.1	85.2

Table 7: Results of the self-trained generative parser and updated reranking parser on the test set. 10×T+10×D+P stands for 10×CTB+10×CDT$_{\lambda}^{PS}$+PDC.

84.2% f-score, better than the result of the reranking parser with CTB and CDT^{PS} as training data (shown in Table 5). It indicates that the use of probability information from the parser for tree conversion helps target grammar parsing.

4.3 Using Unlabeled Data for Parsing

Recent studies on parsing indicate that the use of unlabeled data by self-training can help parsing on the WSJ data, even when labeled data is relatively large (McClosky et al., 2006a; Reichart and Rappoport, 2007). It motivates us to employ self-training technique for Chinese parsing. We used the POS tagged People Daily corpus[9] (Jan. 1998∼Jun. 1998, and Jan. 2000∼Dec. 2000) (PDC) as unlabeled data for parsing. First we removed the sentences with less than 3 words or more than 40 words from PDC to ease parsing, resulting in 820k sentences. Then we ran the reranking parser in Section 4.2.2 on PDC and used the parses on PDC as additional training data for the generative parser. Here we tried the corpus weighting technique for an optimal combination of CTB, CDT$_{\lambda}^{PS}$ and parsed PDC, and chose the relative weight of both CTB and CDT$_{\lambda}^{PS}$ as 10 by cross validation on the development set. Finally we retrained the generative parser on CTB, CDT$_{\lambda}^{PS}$ and parsed PDC. Furthermore, we used this self-trained generative parser as a base parser to retrain the reranker on CTB and CDT$_{\lambda}^{PS}$.

Table 7 shows the performance of self-trained generative parser and updated reranker on the test set, with CTB and CDT$_{\lambda}^{PS}$ as labeled data. We see that the use of unlabeled data by self-training further increased the reranking parser's performance from 84.2% to 85.2%. Our results on Chinese data confirm previous findings on English data shown in (McClosky et al., 2006a; Reichart and Rappoport, 2007).

[9]Available at http://icl.pku.edu.cn/.

4.4 Comparison with Previous Studies for Chinese Parsing

Table 8 and 9 present the results of previous studies on CTB. All the works in Table 8 used CTB articles 1-270 as labeled data. In Table 9, Petrov and Klein (2007) trained their model on CTB articles 1-270 and 400-1151, and Burkett and Klein (2008) used the same CTB articles and parse trees of their English translation (from the English Chinese Translation Treebank) as training data. Comparing our result in Table 6 with that of Petrov and Klein (2007), we see that CDT$_{\lambda}^{PS}$ helps parsing on CTB, which brought 0.9% f-score improvement. Moreover, the use of unlabeled data further boosted the parsing performance to 85.2%, an absolute 1.0% improvement over the previous best result presented in Burkett and Klein (2008).

5 Related Work

Recently there have been some studies addressing how to use treebanks with same grammar formalism for domain adaptation of parsers. Roark and Bachiani (2003) presented count merging and model interpolation techniques for domain adaptation of parsers. They showed that their system with count merging achieved a higher performance when in-domain data was weighted more heavily than out-of-domain data. McClosky et al. (2006b) used self-training and corpus weighting to adapt their parser trained on WSJ corpus to Brown corpus. Their results indicated that both unlabeled in-domain data and labeled out-of-domain data can help domain adaptation. In comparison with these works, we conduct our study in a different setting where we work with multiple heterogeneous treebanks.

Grammar formalism conversion makes it possible to reuse existing source treebanks for the study of target grammar parsing. Wang et al. (1994) employed a parser to help conversion of a treebank from a simple phrase structure to a more informative phrase structure and then used this converted treebank to train their parser. Collins et al. (1999) performed statistical constituency parsing of Czech on a treebank that was converted from the Prague Dependency Treebank under the guidance of conversion rules and heuristic rules, e.g., one level of projection for any category, minimal projection for any dependents, and fixed position of attachment. Xia and Palmer (2001) adopted better heuristic rules to build converted trees, which

	≤ 40 words			All the sentences		
	LR	LP	F	LR	LP	F
Models	(%)	(%)	(%)	(%)	(%)	(%)
Bikel & Chiang (2000)	76.8	77.8	77.3	-	-	-
Chiang & Bikel (2002)	78.8	81.1	79.9	-	-	-
Levy & Manning (2003)	79.2	78.4	78.8	-	-	-
Bikel's thesis (2004)	78.0	81.2	79.6	-	-	-
Xiong et. al. (2005)	78.7	80.1	79.4	-	-	-
Chen et. al. (2005)	81.0	81.7	81.2	76.3	79.2	77.7
Wang et. al. (2006)	79.2	81.1	80.1	76.2	78.0	77.1

Table 8: Results of previous studies on CTB with CTB articles 1-270 as labeled data.

	≤ 40 words			All the sentences		
	LR	LP	F	LR	LP	F
Models	(%)	(%)	(%)	(%)	(%)	(%)
Petrov & Klein (2007)	85.7	86.9	86.3	81.9	84.8	83.3
Burkett & Klein (2008)	-	-	-	-	-	84.2

Table 9: Results of previous studies on CTB with more labeled data.

reflected the structural preference in their target grammar. For acquisition of better conversion rules, Xia et al. (2008) proposed to automatically extract conversion rules from a target treebank. Moreover, they presented two strategies to solve the problem that there might be multiple conversion rules matching the same input dependency tree pattern: (1) choosing the most frequent rules, (2) preferring rules that add fewer number of nodes and attach the subtree lower.

In comparison with the works of Wang et al. (1994) and Collins et al. (1999), we went further by combining the converted treebank with the existing target treebank for parsing. In comparison with previous conversion methods (Collins et al., 1999; Covington, 1994; Xia and Palmer, 2001; Xia et al., 2008) in which for each head-dependent pair, only one locally optimal conversion was kept during tree-building process, we employed a parser to generate globally optimal syntactic structures, eliminating heuristic rules for conversion. In addition, we used converted trees to retrain the parser for better conversion candidates, while Wang et al. (1994) did not exploit the use of converted trees for parser retraining.

6 Conclusion

We have proposed a two-step solution to deal with the issue of using heterogeneous treebanks for parsing. First we present a parser based method to convert grammar formalisms of the treebanks to the same one, without applying predefined heuristic rules, thus turning the original problem into the problem of parsing on homogeneous treebanks. Then we present two strategies, instance pruning and score interpolation, to refine conversion results. Finally we adopt the corpus weighting technique to combine the converted source treebank with the existing target treebank for parser training.

The study on the WSJ data shows the benefits of our parser based approach for grammar formalism conversion. Moreover, experimental results on the Penn Chinese Treebank indicate that a converted dependency treebank helps constituency parsing, and it is better to exploit probability information produced by the parser through score interpolation than to prune low quality trees for the use of the converted treebank.

Future work includes further investigation of our conversion method for other pairs of grammar formalisms, e.g., from the grammar formalism of the Penn Treebank to more deep linguistic formalism like CCG, HPSG, or LFG.

References

Anne Abeille, Lionel Clement and Francois Toussenel. 2000. Building a Treebank for French. *In Proceedings of LREC 2000*, pages 87-94.

Daniel Bikel and David Chiang. 2000. Two Statistical Parsing Models Applied to the Chinese Treebank. *In Proceedings of the Second SIGHAN workshop*, pages 1-6.

Daniel Bikel. 2004. On the Parameter Space of Generative Lexicalized Statistical Parsing Models. *Ph.D. thesis*, University of Pennsylvania.

Alena Bohmova, Jan Hajic, Eva Hajicova and Barbora Vidova-Hladka. 2003. The Prague Dependency Treebank: A Three-Level Annotation Scenario. *Treebanks:*

Building and Using Annotated Corpora. Kluwer Academic Publishers, pages 103-127.

Thorsten Brants, Wojciech Skut and Hans Uszkoreit. 1999. Syntactic Annotation of a German Newspaper Corpus. *In Proceedings of the ATALA Treebank Workshop*, pages 69-76.

David Burkett and Dan Klein. 2008. Two Languages are Better than One (for Syntactic Parsing). *In Proceedings of EMNLP 2008*, pages 877-886.

Eugene Charniak. 2000. A Maximum Entropy Inspired Parser. *In Proceedings of NAACL 2000*, pages 132-139.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-Fine N-Best Parsing and MaxEnt Discriminative Reranking. *In Proceedings of ACL 2005*, pages 173-180.

Ying Chen, Hongling Sun and Dan Jurafsky. 2005. A Corrigendum to Sun and Jurafsky (2004) Shallow Semantic Parsing of Chinese. *University of Colorado at Boulder CSLR Tech Report TR-CSLR-2005-01*.

David Chiang and Daniel M. Bikel. 2002. Recovering Latent Information in Treebanks. *In Proceedings of COLING 2002*, pages 1-7.

Micheal Collins, Lance Ramshaw, Jan Hajic and Christoph Tillmann. 1999. A Statistical Parser for Czech. *In Proceedings of ACL 1999*, pages 505-512.

Micheal Covington. 1994. GB Theory as Dependency Grammar. *Research Report AI-1992-03*.

Martin Forst. 2003. Treebank Conversion - Establishing a Testsuite for a Broad-Coverage LFG from the TIGER Treebank. *In Proceedings of LINC at EACL 2003*, pages 25-32.

Chunghye Han, Narae Han, Eonsuk Ko and Martha Palmer. 2002. Development and Evaluation of a Korean Treebank and its Application to NLP. *In Proceedings of LREC 2002*, pages 1635-1642.

Sadao Kurohashi and Makato Nagao. 1998. Building a Japanese Parsed Corpus While Improving the Parsing System. *In Proceedings of LREC 1998*, pages 719-724.

Roger Levy and Christopher Manning. 2003. Is It Harder to Parse Chinese, or the Chinese Treebank? *In Proceedings of ACL 2003*, pages 439-446.

Ting Liu, Jinshan Ma and Sheng Li. 2006. Building a Dependency Treebank for Improving Chinese Parser. *Journal of Chinese Language and Computing*, 16(4):207-224.

Mitchell P. Marcus, Beatrice Santorini and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313-330.

David McClosky, Eugene Charniak and Mark Johnson. 2006a. Effective Self-Training for Parsing. *In Proceedings of NAACL 2006*, pages 152-159.

David McClosky, Eugene Charniak and Mark Johnson. 2006b. Reranking and Self-Training for Parser Adaptation. *In Proceedings of COLING/ACL 2006*, pages 337-344.

Antonio Moreno, Susana Lopez, Fernando Sanchez and Ralph Grishman. 2003. Developing a Syntactic Annotation Scheme and Tools for a Spanish Treebank. *Treebanks: Building and Using Annotated Corpora. Kluwer Academic Publishers*, pages 149-163.

Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. *In Proceedings of HLT/NAACL 2007*, pages 404-411.

Roi Reichart and Ari Rappoport. 2007. Self-Training for Enhancement and Domain Adaptation of Statistical Parsers Trained on Small Datasets. *In Proceedings of ACL 2007*, pages 616-623.

Brian Roark and Michiel Bacchiani. 2003. Supervised and Unsupervised PCFG Adaptation to Novel Domains. *In Proceedings of HLT/NAACL 2003*, pages 126-133.

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 Teruko Mitamura. 2006. A Fast, Accurate Deterministic Parser for Chinese. *In Proceedings of COLING/ACL 2006*, pages 425-432.

Stephen Watkinson and Suresh Manandhar. 2001. Translating Treebank Annotation for Evaluation. *In Proceedings of ACL Workshop on Evaluation Methodologies for Language and Dialogue Systems*, pages 1-8.

Fei Xia and Martha Palmer. 2001. Converting Dependency Structures to Phrase Structures. *In Proceedings of HLT 2001*, pages 1-5.

Fei Xia, Rajesh Bhatt, Owen Rambow, Martha Palmer and Dipti Misra. Sharma. 2008. Towards a Multi-Representational Treebank. *In Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories*, pages 159-170.

Deyi Xiong, Shuanglong Li, Qun Liu, Shouxun Lin and Yueliang Qian. 2005. Parsing the Penn Chinese Treebank with Semantic Knowledge. *In Proceedings of IJCNLP 2005*, pages 70-81.

Nianwen Xue, Fei Xia, Fu-Dong Chiou and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207-238.