Effective Constituent Projection across Languages

Wenbin Jiang and **Yajuan Lü** and **Yang Liu** and **Qun Liu**

Key Laboratory of Intelligent Information Processing
Institute of Computing Technology
Chinese Academy of Sciences
{jiangwenbin, lvyajuan, yliu, liuqun}@ict.ac.cn

Abstract

We describe an effective constituent projection strategy, where constituent projection is performed on the basis of dependency projection. Especially, a novel measurement is proposed to evaluate the candidate projected constituents for a target language sentence, and a PCFG-style parsing procedure is then used to search for the most probable projected constituent tree. Experiments show that, the parser trained on the projected treebank can significantly boost a state-of-the-art supervised parser. When integrated into a tree-based machine translation system, the projected parser leads to translation performance comparable with using a supervised parser trained on thousands of annotated trees.

1 Introduction

In recent years, supervised constituent parsing has been well studied and achieves the state-of-the-art for many resource-rich languages (Collins, 1999; Charniak, 2000; Petrov et al., 2006). Because of the cost and difficulty in treebank construction, researchers have also investigated the utilization of unannotated text, including the unsupervised parsing which totally uses unannotated data (Klein and Manning, 2002; Klein and Manning, 2004; Bod, 2006; Seginer, 2007), and the semi-supervised parsing which uses both annotated and unannotated data (Sarkar, 2001; Steedman et al., 2003; McClosky et al., 2006).

Because of the higher complexity and lower performance of unsupervised methods, as well as the need of reliable priori knowledge in semi-supervised methods, it seems promising to project the syntax structures from a resource-rich language to a resource-scarce one across a bilingual corpus. Lots of researches have so far been devoted to dependency projection (Hwa et al., 2002; Hwa et al., 2005; Ganchev et al., 2009; Smith and Eisner, 2009). While for constituent projection there is few progress. This is due to the fact that the constituent syntax describes the language structure in a more detailed way, and the degree of isomorphism between constituent structures appears much lower.

In this paper we propose for constituent projection a stepwise but totally automatic strategy, which performs constituent projection on the basis of dependency projection, and then use a constraint EM optimization algorithm to optimized the initially projected trees. Given a word-aligned bilingual corpus with source sentences parsed, we first project the dependency structures of these constituent trees to the target sentences using a dynamic programming algorithm, then we generate a set of candidate constituents for each target sentence and design a novel evaluation function to calculate the probability of each candidate constituent, finally, we develop a PCFG-style parsing procedure to search for the most probable projected constituent tree in the evaluated candidate constituent set. In addition, we design a constraint EM optimization procedure to decrease the noise in the initially projected constituent treebank.

Experimental results validate the effectiveness of our approach. On the Chinese-English FBIS corpus, we project the English parses produced by the Charniak parser across to the Chinese sen-

Coling 2010: Poster Volume, pages 516–524,
Beijing, August 2010

tences. A berkeley parser trained on this projected treebank can effectively boost the supervised parsers trained on bunches of CTB trees. Especially, the supervised parser trained on the smaller CTB 1.0 benefits a significant F-measure increment of more than 1 point from the projected parser. When using the projected parser in a tree-based translation model (Liu et al., 2006), we achieve translation performance comparable with using a state-of-the-art supervised parser trained on thousands of CTB trees. This surprising result gives us an inspiration that better translation would be achieved by combining both projected parsing and supervised parsing into a hybrid parsing schema.

2 Stepwise Constituent Projection

We first introduce the dynamic programming procedure for dependency projection, then describe the PCFG-style algorithm for constituent projection which is conducted on projected dependent structures, and finally show the constraint EM procedure for constituent optimization.

2.1 Dependency Projection

For dependency projection we adopt a dynamic programming algorithm, which searches the most probable projected target dependency structure according to the source dependency structure and the word alignment.

In order to mitigate the effect of word alignment errors, multiple GIZA++ (Och and Ney, 2000) results are combined into a compact representation called alignment matrix. Given a source sentence with m words, represented as $E_{1:m}$, and a target sentence with n words, represented as $F_{1:n}$, their word alignment matrix A is an $m \times n$ matrix, where each element $A_{i,j}$ denotes the probability of the source word E_i aligned to the target word F_j.

Using $P(D_F|D_E, A)$ to denote the probability of the projected target dependency structure D_F conditioned on the source dependency structure D_E and the alignment matrix A, the projection algorithm aims to find

$$\tilde{D_F} = \operatorname*{argmax}_{D_F} P(D_F|D_E, A) \qquad (1)$$

Algorithm 1 Dependency projection.

```
1:  Input: $F$, and $P_e$ for all word pairs in $F$
2:  for $\langle i, j \rangle \subseteq \langle 1, |F| \rangle$ in topological order do
3:      buf $\leftarrow \emptyset$
4:      for $k \leftarrow i..j-1$ do                          ▷ all partitions
5:          for $l \in \mathbf{V}[i,k]$ and $r \in \mathbf{V}[k+1,j]$ do
6:              insert $\textsc{Deriv}(l, r, P_e)$ into buf
7:              insert $\textsc{Deriv}(r, l, P_e)$ into buf
8:      $\mathbf{V}[i,j] \leftarrow$ top $K$ derivations of buf
9:  Output: the best derivation of $\mathbf{V}[1, |F|]$
10: function $\textsc{Deriv}(p, c, P_e)$
11:     $d \leftarrow p \cup c \cup \{p \cdot root \curvearrowright c \cdot root\}$ ▷ new derivation
12:     $d \cdot evl \leftarrow \textsc{Eval}(d, P_e)$          ▷ evaluation function
13:     return $d$
```

$P(D_F|D_E, A)$ can be factorized into each dependency edge $x \curvearrowright y$ in D_F

$$P(D_F|D_E, A) = \prod_{x \curvearrowright y \in D_F} P_e(x \curvearrowright y|D_E, A)$$

P_e can then be obtained by simple accumulation across all possible situations of correspondence

$$\begin{aligned} & P_e(x \curvearrowright y|D_E, A) \\ & \quad = \sum_{1 \le x', y' \le |E|} A_{x,x'} \times A_{y,y'} \times \delta(x', y'|D_E) \end{aligned}$$

where $\delta(x', y'|D_E)$ is a 0-1 function that equals 1 only if the dependent relation $x' \curvearrowright y'$ holds in D_E.

The search procedure needed by the argmax operation in equation 1 can be effectively solved by the Chu-Liu-Edmonds algorithm used in (McDonald et al., 2005). In this work, however, we adopt a more general and simple dynamic programming algorithm as shown in Algorithm 1, in order to facilitate the possible expansions. In practice, the cube-pruning strategy (Huang and Chiang, 2005) is used to speed up the enumeration of derivations (loops started by line 4 and 5).

2.2 Constituent Projection

The PCFG-style parsing procedure searches for the most probable projected constituent tree in a shrunken search space determined by the projected dependency structure and the target constituent tree. The shrunken search space can be built as following. First, we generates the candidate constituents of the source tree and the candidate spans of the target sentence, so as to enumerate the candidate constituents of the target sentence. Then we compute the consistent degree for each pair of candidate constituent and span, and further estimate the probability of each candidate constituent for the target sentence.

2.2.1 Candidate Constituents and Spans

For the candidate constituents of the source tree, using only the original constituents imposes a strong hypothesis of isomorphism on the constituent projection between two languages, since it requires that each couple of constituent and span must be strictly matched. While for the candidate spans of the target sentences, using all subsequences makes the search procedure suffer from more perplexity. Therefore, we expand the candidate constituent set and restrict the candidate span set:

- Candidate Constituent: Suppose a production in the source constituent tree, denoted as $p \rightarrow c_1 c_2 .. c_h .. c_{|p|}$, and c_h is the head child of the parent p. Each constituent, p or c, is a triple $\langle lb, rb, nt \rangle$, where nt denotes its non-terminal, while lb and rb represent its left- and right bounds of the sub-sequence that the constituent covers. The candidate constituent set of this production consists the head of the production itself, and a set of incomplete constituents,

  $$\begin{aligned} \{\langle l, r, p \cdot nt* \rangle | c_1 \cdot lb \leq l \leq c_h \cdot lb \wedge \\ c_h \cdot rb \leq r \leq c_{|p|} \cdot rb \wedge \\ (l < c_h \cdot lb \vee r > c_h \cdot rb)\} \end{aligned}$$

  where the symbol $*$ indicates an incomplete non-terminal. The candidate constituent set of the entire source tree is the unification of the sets extracted from all productions of the tree.

- Candidate Span: A candidate span of the target sentence is a tuple $\langle lb, rb \rangle$, where lb and rb indicate the same as in a constituent. We define the candidate span set as the spans of all *regular dependent segment*s in the corresponding projected dependency structure. A regular dependency segment is a dependent segment that every modifier of the root is a complete dependency structure. Suppose a dependency structure rooted at word p, denoted as $c_{lL} .. c_{l2} c_{l1} \curvearrowleft p \curvearrowright c_{r1} c_{r2} .. c_{rR}$, it has L $(L \geq 0)$ modifiers on its left and R $(R \geq 0)$ modifiers on its right, each of them is a smaller complete dependency structure. Then the word p itself is a regular dependency segment without any modifier, and

  $$\begin{aligned} \{c_{li} .. c_{l1} \curvearrowleft p \curvearrowright c_{r1} .. c_{rj} | 0 \leq i \leq L \wedge \\ 0 \leq j \leq R \wedge \\ (i > 0 \vee j > 0)\} \end{aligned}$$

  is a set of regular dependency structures with at least one modifier. The regular dependency segments of the entire projected dependency structure can simply be accumulated across all dependency nodes.

2.2.2 Span-to-Constituent Correspondence

After determining the candidate constituent set of the source tree, denoted as Φ_E, and the candidate span set of the target sentence, denoted as Ψ_F, we then calculate the consistent degree for each pair of candidate constituent and candidate span.

Given a candidate constituent $\phi \in \Phi_E$ and a candidate span $\psi \in \Psi_F$, their consistent degree $\mathcal{C}(\psi, \phi | A)$ is the probability that they are aligned to each other according to A.

We display the derivations from bottom to up. First, we define the alignment probability from a word i in the span ψ to the constituent ϕ as

$$P(i \mapsto \phi | A) = \frac{\sum_{\phi \cdot lb \leq j \leq \phi \cdot rb} A_{i,j}}{\sum_j A_{i,j}}$$

Then we define the alignment probability from the span ψ to the constituent ϕ as

$$P(\psi \mapsto \phi | A) = \prod_{\psi \cdot lb \leq i \leq \psi \cdot rb} P(i \mapsto \phi | A)$$

Note that we use i to denote both a word and its index for simplicity without causing confusion. Finally, we define $\mathcal{C}(\phi, \psi | A)$ as

$$\mathcal{C}(\psi, \phi | A) = P(\psi \mapsto \phi | A) \times P(\phi \mapsto \psi | A^T) \quad (2)$$

Where $P(\phi \mapsto \psi | A^T)$ denotes the alignment probability from the constituent ϕ to the span ψ, it can be calculated in the same manner.

2.2.3 Constituent Projection Algorithm

The purpose of constituent projection is to find the most probable projected constituent tree for the target sentence conditioned on the source constituent tree and the word alignment

$$\tilde{T_F} = \operatorname*{argmax}_{T_F \subseteq \Phi_F} P(T_F|T_E, A) \tag{3}$$

Here, we use Φ_F to denote the set of candidate constituents of the target sentence

$$\begin{aligned}\Phi_F &= \Psi_F \otimes NT(\Phi_E) \\ &= \{\phi_F|\psi(\phi_F) \in \Psi_F \wedge nt(\phi_F) \in NT(\Phi_E)\}\end{aligned}$$

where $\psi(\cdot)$ and $nt(\cdot)$ represent the span and the non-terminal of a constituent respectively, and $NT(\cdot)$ represents the set of non-terminals extracted from a constituent set. Note that T_F is a subset of Φ_F if we treat a tree as a set of constituents.

The probability of the projected tree T_F can be factorized into the probabilities of the projected constituents that composes the tree

$$P(T_F|T_E, A) = \prod_{\phi_F \in T_F} P_\phi(\phi_F|T_E, A)$$

while the probability of the projected source constituent can be defined as a statistics of span-to-constituent- and constituent-to-constituent consistent degrees

$$P_\phi(\phi_F|T_E, A) = \frac{\sum_{\phi_E \in \Phi_E} \mathcal{C}(\phi_F, \phi_E|A)}{\sum_{\phi_E \in \Phi_E} \mathcal{C}(\psi(\phi_F), \phi_E|A)}$$

where $\mathcal{C}(\phi_F, \phi_E|A)$ in the numerator denotes the consistent degree for each pair of constituents, which can be calculated based on that of span and constituent described in Formula 2

$$\mathcal{C}(\phi_F, \phi_E) = \begin{cases} 0 & \textbf{if } \phi_F \cdot nt \neq \phi_E \cdot nt \\ \mathcal{C}(\psi(\phi_F), \phi_E) & \textbf{else} \end{cases}$$

Algorithm 2 shows the pseudocode for constituent projection. A PCFG-style parsing procedure searches for the best projected constituent tree in the constrained space determined by Ψ_F. Note that the projected trees are binarized, and can be easily recovered according to the asterisks at the tails of non-terminals.

Algorithm 2 Constituent projection.

```
Input: $\Psi_F$, $\Phi_F$, and $P_\phi$ for all spans in $\Psi_F$
for $\langle i, j \rangle \in \Psi$ in topological order do
    buf ← $\emptyset$
    for $p \in \Phi_F$ s.t. $\psi(p) = \langle i, j \rangle$ do
        for $k \leftarrow i..j-1$ do                    ▷ all partitions
            for $l \in \mathbf{V}[i, k]$ and $r \in \mathbf{V}[k+1, j]$ do
                insert DERIV($l, r, p, P_\phi$) into buf
    $\mathbf{V}[i, j]$ ← top $K$ derivations of buf
Output: the best derivation of $\mathbf{V}[1, |F|]$
function DERIV($l$, $r$, $p$, $P_\phi$)
    $d \leftarrow l \cup r \cup \{p\}$                  ▷ new derivation
    $d \cdot evl \leftarrow$ EVAL($d, P_\phi$)          ▷ evaluation function
    return $d$
```

2.3 EM Optimization

Since the constituent projection is conducted on each sentence pair separately, the projected treebank is apt to suffer from more noise caused by free translation and word alignment error. It can be expected that an EM iteration over the whole projected treebank will lead to trees with higher consistence.

We adopt the inside-outside algorithm to improve the quality of the initially projected treebank. Different from previous works, all expectation and maximization operations for a single tree are performed in a constrained space determined by the candidate span set of the projected target dependency structure. That is to say, all the summation operations, both for calculating α/β values and for re-estimating the rule probabilities, only consider the spans in the candidate span set. This means that the projected dependency structures are supposed believable, and the noise is mainly introduced in the following constituent projection procedure.

Here we give an overall description of the treebank optimization procedure. First, an initial PCFG grammar G_F^0 is estimated from the original projected treebank. Then several iterations of α/β calculation and rule probability re-estimation are performed. For example in the i-the iteration, α/β values are calculated based on the current grammar G_F^{i-1}, afterwards the optimized grammar G_F^i is obtained based on these α/β values. The iterative procedure terminates when the likelihood of whole treebank increases slowly. Finally, with the optimized grammar, a constrained PCFG parsing procedure is conducted on each of the initial projected trees, so as to obtain an optimized treebank.

3 Applications of Constituent Projection

The most direct contribution of constituent projection is pushing an initial step for the statistical constituent parsing of resource-scarce languages. It also has some meaningful applications even for the resource-rich languages. For instances, the projected treebank, due to its large scale and high coverage, can used to boost an traditional supervised-trained parser. And, the parser trained on the projected treebank can adopted to conduct tree-to-string machine translation, since it give parsing results with larger isomorphism with the target language than a supervised-trained parser dose.

3.1 Boost an Traditional Parser

We first establish a unified framework for the enhanced parser where a projected parser is adopted to guide the parsing procedure of the baseline parser.

For a given target sentence S, the enhanced parser selected the best parse $\tilde{T}$ among the set of candidates $\Omega(S)$ according to two evaluation functions, given by the baseline parser $\mathbb{B}$ and the projected guide parser $\mathbb{G}$, respectively.

$$\tilde{T} = \operatorname*{argmax}_{T \in \Omega(S)} P(T|\mathbb{B}) \times P(T|\mathbb{G})^{\lambda} \tag{4}$$

These two evaluation functions can be integrated deeply into the decoding procedure (Carreras et al., 2008; Zhang and Clark, 2008; Huang, 2008), or can be integrated at a shallow level in a reranking manner (Collins, 2000; Charniak and Johnson, 2005). For simplicity and generability, we adopt the reranking strategy. In k-best reranking, $\Omega(S)$ is simply a set of candidate parses, denoted as $\{T_1, T_2, ..., T_k\}$, and we use the single parse of the guide parser, $T_{\mathbb{G}}$, to re-evaluate these candidates. Formula 4 can be redefined as

$$\tilde{T}(T_{\mathbb{G}}) = \operatorname*{argmax}_{T \in \Omega(S)} \mathbf{w} \cdot \mathbf{f}(T, T_{\mathbb{G}}) \tag{5}$$

Here, $\mathbf{f}(T, T_{\mathbb{G}})$ and $\mathbf{w}$ represent a high dimensional feature representation and a corresponding weight vector, respectively. The first feature $f_1(T, T_{\mathbb{G}}) = logP(T|\mathbb{B})$ is the log probability of the baseline parser, while the remaining features are integer-valued guide features, and each of them represents the guider parser's predication result for a particular configuration in candidate parse T, so as to utilize the projected parser's knowledge to guide the parsing procedure of the traditional parser.

In our work a guide feature is composed of two parts, the non-terminal of a certain constituent ϕ in the candidate parse T,[1] and the non-terminal at the corresponding span $\psi(\phi)$ in the projected parse $T_{\mathbb{G}}$. Note that in the projected parse this span does not necessarily correspond to a constituent. In such situations, we simply use the non-terminal of the constituent that just be able to cover this span, and attach a asterisk at the tail of this non-terminal. Here is an example of the guide features

$$f_{100}(T, T_{\mathbb{G}}) = VP \in T \ \circ PP* \ \in T_{\mathbb{G}}$$

It represents that a VP in the candidate parse corresponds to a segment of a PP in the projected parse. The quantity of its weight w_{100} indicates how probably a span can be predicated as VP if the span corresponds to a partial PP in the projected parse.

We adopt the perceptron algorithm to train the reranker. To reduce overfitting and produce a more stable weight vector, we also use a refinement strategy called averaged parameters (Collins, 2002).

3.2 Using in Machine Translation

Researchers have achieved promising improvements in tree-based machine translation (Liu et al., 2006; Huang et al., 2006). Such models use a parsed tree as input and converts it into a target tree or string. Given a source language sentence, first we use a traditional source language parser to parse the sentence to obtain the syntax tree T, and then use the translation decoder to search for the best derivation $\tilde{d}$, where a derivation d is a sequence of transformations that converts the source tree into the target language string

$$\tilde{d} = \operatorname*{argmax}_{d \in D} P(d|T) \tag{6}$$

[1] Using non-terminals as features brings no improvement in the reranking experiments, so as to examine the impact of the projected parser.

Here D is the candidate set of d, and it is determined by the source tree T and the transformation rules.

Since the tree-based models are based on the synchronous transformational grammars, they suffer much from the isomerism between the source syntax and the target sentence structure. Considering that the parsed tree produced by a projected parser may have larger isomorphism with the target language, it would be a promising idea to adopt the projected parser to parse the input sentence for the subsequent translation decoding procedure.

4 Experiments

In this section, we first invalidate the effect of constituent projection by evaluating a parser trained on the projected treebank. Then we investigate two applications of the projected parser: boosting an traditional supervised-trained parser, and integration in a tree-based machine translation system. Following the previous works, we depict the parsing performance by F-score on sentences with no more than 40 words, and evaluate the translation quality by the case-sensitive BLEU-4 metric (Papineni et al., 2002) with 4 references.

4.1 Constituent Projection

We perform constituent projection from English to Chinese on the FBIS corpus, which contains 239K sentence pairs with about 6.9M/8.9M words in Chinese/English. The English sentences are parsed by the Charniak Parser and the dependency structures are extracted from these parses according to the head-finding rules of (Yamada and Matsumoto, 2003). The word alignment matrixes are obtained by combining the 10-best results of GIZA++ according to (Liu et al., 2009).

We first project the dependency structures from English to Chinese according to section 2.1, and then project the constituent structures according to section 2.2. We define an assessment criteria to evaluate the confidence of the final projected constituent tree

$$c = \sqrt[n]{P(D_F|D_E, A) \times P(T_F|T_E, A)}$$

where n is the word count of a Chinese sentence in our experiments. A series of projected Chinese treebanks with different scales are obtained by specifying different c as the filtering threshold. The state-of-the-art Berkeley Parser is adopted to train on these treebanks because of its high performance and independence of head word information.

Thres c	**#Resrv**	**Cons-F_1**	**Span-F_1**
0.5	12.6K	23.9	32.7
0.4	17.8K	23.9	33.4
0.3	27.2K	25.4	35.7
0.2	45.1K	26.6	38.0
0.1	87.0K	27.8	40.4

Table 1: Performances of the projected parsers on the CTB test set. #Resrv denotes the amount of reserved trees within threshold c. Cons-F_1 is the traditional F-measure, while Span-F_1 is the F-measure without consideration of non-terminals.

Table 1 shows the performances of these projected parsers on the standard CTB test set, which is composed of sentences in chapters 271-300. We find that along with the decrease of the filtering threshold c, more projected trees are reserved and the performance of the projected parser constantly increases. We also find that the traditional F-value, Cons-F_1, is obviously lower than the one without considering non-terminals, Span-F_1. This indicates that the constituent projection procedure introduces more noise because of the higher complexity of constituent correspondence. In all the rest experiments, however, we simply use the projected treebank filtered by threshold $c = 0.1$ and do not try any smaller thresholds, since it already takes more than one weak to train the Berkeley Parser on the 87 thousands trees resulted by this threshold.

The constrained EM optimization procedure described in section 2.3 is used to alleviate the noise in the projected treebank, which may be caused by free translation, word alignment errors, and projection on each single sentence pair. Figure 1 shows the log-likelihood on the projected treebank after each EM iteration. It is obvious that the log-likelihood increases very slowly after 10 iterations. We terminate the EM procedure after 40 iterations.

Finally we train the Berkeley Parser on the optimized projected treebank, and test its perfor-

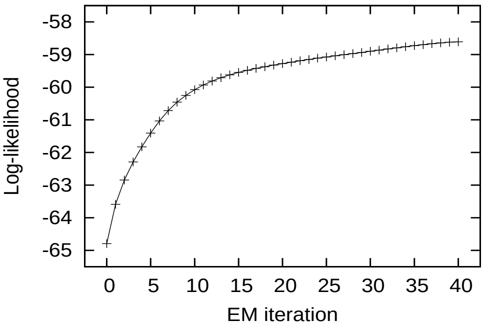

Figure 1: Log-likelihood of the 87K-projected treebank after each EM interation.

Train Set	Cons-F_1	Span-F_1
Original 87K	27.8	40.4
Optimized 87K	22.8	40.2

Table 2: Performance of the parser trained on the optimized projected treebank, compared with that of the original projected parser.

Train Set	Baseline	Bst-Ini	Bst-Opt
CTB 1.0	75.6	76.4	76.9
CTB 5.0	85.2	85.5	85.7

Table 3: Performance improvement brought by the projected parser to the baseline parsers trained on CTB 1.0 and CTB 5.0, respectively. Bst-Ini/Bst-Opt: boosted by the parser trained on the initial/optimized projected treebank.

mance on the standard CTB test set. Table 2 shows the performance of the parser trained on the optimized projected treebank. Unexpectedly, we find that the constituent F_1-value of the parser trained on the optimized treebank drops sharply from the baseline, although the span F_1-value remains nearly the same. We assume that the EM procedure gives the original projected treebank more consistency between each single tree while the revised treebank deviates from the CTB annotation standard, but it needs to be validated by the following experiments.

4.2 Boost an Traditional Parser

The projected parser is used to help the reranking of the k-best parses produced by another state-of-the-art parser, which is called the baseline parser for convenience. In our experiments we choose the revised Chinese parser (Xiong et al., 2005)

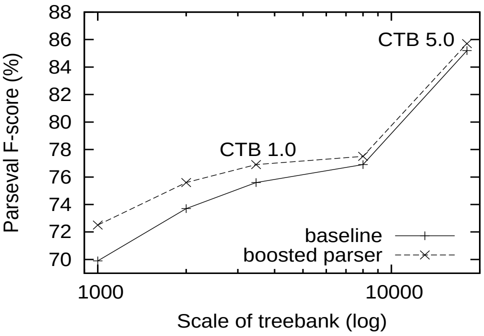

Figure 2: Boosting performance of the projected parser on a series of baseline parsers that are trained on treebanks of different scales.

based on Collins model 2 (Collins, 1999) as the baseline parser.[2]

The baseline parser is respectively trained on CTB 1.0 and CTB 5.0. For both corpora we follow the traditional corpus splitting: chapters 271-300 for testing, chapters 301-325 for development, and else for training. Experimental results are shown in Table 3. We find that both projected parsers bring significant improvement to the baseline parsers. Especially the later, although performs worse on CTB standard test set, gives a larger improvement than the former. This to some degree confirms the previous assumption. However, more investigation must be conducted in the future.

We also observe that for the baseline parser trained on the much larger CTB 5.0, the boosting performance of the projected parser is relatively lower. To further investigate the regularity that the boosting performance changes according to the scale of training treebank of the baseline parser, we train a series of baseline parsers with different amounts of trees, then use the projected parser trained on the optimized treebank to enhance these baseline parsers. Figure 2 shows the experimental results. From the curves we can see that the smaller the training corpus of the baseline parser, the more significant improvement can be obtained. This is a good news for the resource-scarce languages that have no large treebanks.

[2] The Berkeley Parser fails to give k-best parses for some sentences when trained on small treebanks, and these sentences have to be deleted in the k-best reranking experiments.

4.3 Using in Machine Translation

We investigate the effect of the projected parser in the tree-based translation model on Chinese-to-English translation. A series of contrast translation systems are built, each of which uses a supervised Chinese parser (Xiong et al., 2005) trained on a particular amount of CTB trees.

We use the FBIS Chinese-English bitext as the training corpus, the 2002 NIST MT Evaluation test set as our development set, and the 2005 NIST MT Evaluation test set as our test set. We first extract the tree-to-string translation rules from the training corpus by the algorithm of (Liu et al., 2006), and train a 4-gram language model on the Xinhua portion of GIGAWORD corpus with Kneser-Ney smoothing using the SRI Language Modeling Toolkit (Stolcke and Andreas, 2002). Then we use the standard minimum error-rate training (Och, 2003) to tune the feature weights to maximize the system’ s BLEU score.

Figure 3 shows the experimental results. We find that the translation system using the projected parser achieves the performance comparable with the one using the supervised parser trained on CTB 1.0. Considering that the F-score of the projected parser is only 22.8%, which is far below of the 75.6% F-score of the supervised parser trained on CTB 1.0, we can give more confidence to the assumption that the projected parser is apt to describe the syntax structure of the counterpart language. This surprising result also gives us an inspiration that better translation would be achieved by combining projected parsing and supervised parsing into hybrid parsing schema.

5 Conclusion

This paper describes an effective strategy for constituent projection, where dependency projection and constituent projection are consequently conducted to obtain the initial projected treebank, and an constraint EM procedure is then performed to optimized the projected trees. The projected parser, trained on the projected treebank, significantly boosts an existed state-of-the-art supervised-trained parser, especially trained on a smaller treebank. When using the projected parser in tree-based translation, we achieve the translation performance comparable with using a supervised parser trained on thousands of human-annotated trees.

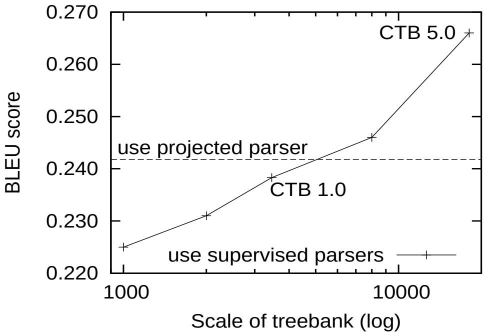

Figure 3: Performances of the translation systems, which use the projected parser and a series of supervised parsers trained CTB trees.

As far as we know, this is the first time that the experimental results are systematically reported about the constituent projection and its applications. However, many future works need to do. For example, more energy needs to be devoted to the treebank optimization, and hybrid parsing schema that integrates the strengths of both supervised-trained parser and projected parser would be valuable to be investigated for better translation.

Acknowledgments

The authors were supported by 863 State Key Project No. 2006AA010108, National Natural Science Foundation of China Contract 60873167, Microsoft Research Asia Natural Language Processing Theme Program grant (2009-2010), and National Natural Science Foundation of China Contract 90920004. We are grateful to the anonymous reviewers for their thorough reviewing and valuable suggestions.

References

Bod, Rens. 2006. An all-subtrees approach to unsupervised parsing. In *Proceedings of the COLING-ACL*.

Carreras, Xavier, Michael Collins, and Terry Koo. 2008. Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of the CoNLL*.

Charniak, Eugene and Mark Johnson. 2005. Coarse-to-fine-grained n-best parsing and discriminative reranking. In *Proceedings of the ACL*.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of the NAACL*.

Collins, Michael. 1999. Head-driven statistical models for natural language parsing. In *Ph.D. Thesis*.

Collins, Michael. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the ICML*, pages 175–182.

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

Ganchev, Kuzman, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proceedings of the 47th ACL*.

Huang, Liang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the IWPT*, pages 53–64.

Huang, Liang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of the AMTA*.

Huang, Liang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the ACL*.

Hwa, Rebecca, Philip Resnik, Amy Weinberg, and Okan Kolak. 2002. Evaluating translational correspondence using annotation projection. In *Proceedings of the ACL*.

Hwa, Rebecca, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. In *Natural Language Engineering*, volume 11, pages 311–325.

Klein, Dan and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of the ACL*.

Klein, Dan and Christopher D. Manning. 2004. Corpusbased induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the ACL*.

Liu, Yang, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the ACL*.

Liu, Yang, Tian Xia, Xinyan Xiao, and Qun Liu. 2009. Weighted alignment matrices for statistical machine translation. In *Proceedings of the EMNLP*.

McClosky, David, Eugene Charniak, and Mark Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of the ACL*.

McDonald, Ryan, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP*.

Och, Franz J. and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the ACL*.

Och, Franz Joseph. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Papineni, Kishore, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the ACL*.

Petrov, Slav, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the ACL*.

Sarkar, Anoop. 2001. Applying co-training methods to statistical parsing. In *Proceedings of NAACL*.

Seginer, Yoav. 2007. Fast unsupervised incremental parsing. In *Proceedings of the ACL*.

Smith, David and Jason Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of EMNLP*.

Steedman, Mark, Miles Osborne, Anoop Sarkar, Stephen Clark, Rebecca Hwa, Julia Hockenmaier, Paul Ruhlen, Steven Baker, and Jeremiah Crim. 2003. Bootstrapping statistical parsers from small datasets. In *Proceedings of the EACL*.

Stolcke and Andreas. 2002. Srilm - an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 311–318.

Xiong, Deyi, Shuanglong Li, Qun Liu, and Shouxun Lin. 2005. Parsing the penn chinese treebank with semantic knowledge. In *Proceedings of IJCNLP 2005*, pages 70–81.

Yamada, H and Y Matsumoto. 2003. Statistical dependency analysis using support vector machines. In *Proceedings of IWPT*.

Zhang, Yue and Stephen Clark. 2008. A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proceedings of EMNLP*.