Dependency Parsing and Projection Based on Word-Pair Classification

Wenbin Jiang and **Qun Liu**

Key Laboratory of Intelligent Information Processing
Institute of Computing Technology
Chinese Academy of Sciences
P.O. Box 2704, Beijing 100190, China
{jiangwenbin, liuqun}@ict.ac.cn

Abstract

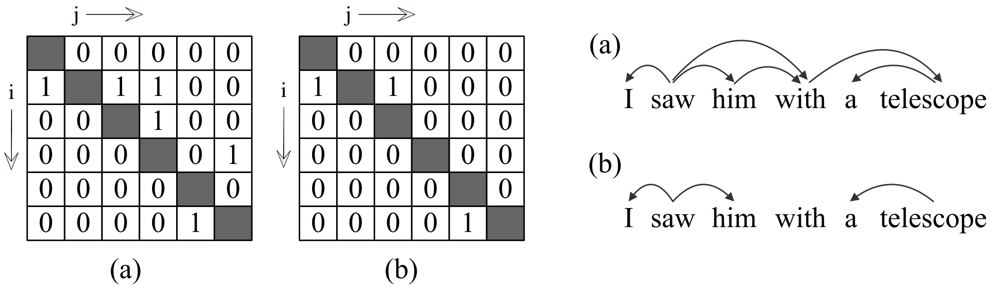
In this paper we describe an intuitionistic method for dependency parsing, where a classifier is used to determine whether a pair of words forms a dependency edge. And we also propose an effective strategy for dependency projection, where the dependency relationships of the word pairs in the source language are projected to the word pairs of the target language, leading to a set of classification instances rather than a complete tree. Experiments show that, the classifier trained on the projected classification instances significantly outperforms previous projected dependency parsers. More importantly, when this classifier is integrated into a maximum spanning tree (MST) dependency parser, obvious improvement is obtained over the MST baseline.

1 Introduction

Supervised dependency parsing achieves the state-of-the-art in recent years (McDonald et al., 2005a; McDonald and Pereira, 2006; Nivre et al., 2006). Since it is costly and difficult to build human-annotated treebanks, a lot of works have also been devoted to the utilization of unannotated text. For example, the unsupervised dependency parsing (Klein and Manning, 2004) which is totally based on unannotated data, and the semisupervised dependency parsing (Koo et al., 2008) which is based on both annotated and unannotated data. Considering the higher complexity and lower performance in unsupervised parsing, and the need of reliable priori knowledge in semisupervised parsing, it is a promising strategy to project the dependency structures from a resource-rich language to a resource-scarce one across a bilingual corpus (Hwa et al., 2002; Hwa et al., 2005; Ganchev et al., 2009; Smith and Eisner, 2009; Jiang et al., 2009).

For dependency projection, the relationship between words in the parsed sentences can be simply projected across the word alignment to words in the unparsed sentences, according to the DCA assumption (Hwa et al., 2005). Such a projection procedure suffers much from the word alignment errors and syntactic isomerism between languages, which usually lead to relationship projection conflict and incomplete projected dependency structures. To tackle this problem, Hwa et al. (2005) use some filtering rules to reduce noise, and some hand-designed rules to handle language heterogeneity. Smith and Eisner (2009) perform dependency projection and annotation adaptation with quasi-synchronous grammar features. Jiang and Liu (2009) resort to a dynamic programming procedure to search for a completed projected tree. However, these strategies are all confined to the same category that dependency projection must produce completed projected trees. Because of the free translation, the syntactic isomerism between languages and word alignment errors, it would be strained to completely project the dependency structure from one language to another.

We propose an effective method for dependency projection, which does not have to produce complete projected trees. Given a word-aligned bilingual corpus with source language sentences parsed, the dependency relationships of the word pairs in the source language are projected to the word pairs of the target language. A dependency relationship is a boolean value that represents whether this word pair forms a dependency edge. Thus a set of classification instances are obtained. Meanwhile, we propose an intuitionistic model for dependency parsing, which uses a classifier to determine whether a pair of words form a dependency edge. The classifier can then be trained on the projected classification instance set, so as to build a projected dependency parser without the need of complete projected trees.



▨	0	0	0	0	0
1	▨	1	1	0	0
0	0	▨	1	0	0
0	0	0	▨	0	1
0	0	0	0	▨	0
0	0	0	0	1	▨

(a)

▨	0	0	0	0	0
1	▨	1	0	0	0
0	0	▨	0	0	0
0	0	0	▨	0	0
0	0	0	0	▨	0
0	0	0	0	1	▨

(b)

Figure 1: Illegal (a) and incomplete (b) dependency tree produced by the simple-collection method.

Experimental results show that, the classifier trained on the projected classification instances significantly outperforms the projected dependency parsers in previous works. The classifier trained on the Chinese projected classification instances achieves a precision of 58.59% on the CTB standard test set. More importantly, when this classifier is integrated into a 2nd-ordered maximum spanning tree (MST) dependency parser (McDonald and Pereira, 2006) in a weighted average manner, significant improvement is obtained over the MST baselines. For the 2nd-order MST parser trained on Penn Chinese Treebank (CTB) 5.0, the classifier give an precision increment of 0.5 points. Especially for the parser trained on the smaller CTB 1.0, more than 1 points precision increment is obtained.

In the rest of this paper, we first describe the word-pair classification model for dependency parsing (section 2) and the generation method of projected classification instances (section 3). Then we describe an application of the projected parser: boosting a state-of-the-art 2nd-ordered MST parser (section 4). After the comparisons with previous works on dependency parsing and projection, we finally five the experimental results.

2 Word-Pair Classification Model

2.1 Model Definition

Following (McDonald et al., 2005a), $\mathbf{x}$ is used to denote the sentence to be parsed, and x_i to denote the i-th word in the sentence. $\mathbf{y}$ denotes the dependency tree for sentence $\mathbf{x}$, and $(i, j) \in \mathbf{y}$ represents a dependency edge from word x_i to word x_j, where x_i is the parent of x_j.

The task of the word-pair classification model is to determine whether any candidate word pair, x_i and x_j s.t. $1 \leq i, j \leq |\mathbf{x}|$ and $i \neq j$, forms a dependency edge. The classification result $\mathcal{C}(i, j)$ can be a boolean value:

$$\mathcal{C}(i, j) = p \qquad p \in \{0, 1\} \tag{1}$$

as produced by a support vector machine (SVM) classifier (Vapnik, 1998). $p = 1$ indicates that the classifier supports the candidate edge (i, j), and $p = 0$ the contrary. $\mathcal{C}(i, j)$ can also be a real-valued probability:

$$\mathcal{C}(i, j) = p \qquad 0 \leq p \leq 1 \tag{2}$$

as produced by an maximum entropy (ME) classifier (Berger et al., 1996). p is a probability which indicates the degree the classifier support the candidate edge (i, j). Ideally, given the classification results for all candidate word pairs, the dependency parse tree can be composed of the candidate edges with higher score (1 for the boolean-valued classifier, and large p for the real-valued classifier). However, more robust strategies should be investigated since the ambiguity of the language syntax and the classification errors usually lead to illegal or incomplete parsing result, as shown in Figure 1.

Follow the edge based factorization method (Eisner, 1996), we factorize the score of a dependency tree $\mathbf{s}(\mathbf{x}, \mathbf{y})$ into its dependency edges, and design a dynamic programming algorithm to search for the candidate parse with maximum score. This strategy alleviate the classification errors to some degree and ensure a valid, complete dependency parsing tree. If a boolean-valued classifier is used, the search algorithm can be formalized as:

$$\begin{aligned} \tilde{\mathbf{y}} &= \operatorname*{argmax}_{\mathbf{y}} \mathbf{s}(\mathbf{x}, \mathbf{y}) \\ &= \operatorname*{argmax}_{\mathbf{y}} \sum_{(i,j) \in \mathbf{y}} \mathcal{C}(i, j) \end{aligned} \tag{3}$$

And if a probability-valued classifier is used instead, we replace the accumulation with cumula-

Type	Features		
Unigram	$word_i \circ pos_i$	$word_i$	pos_i
	$word_j \circ pos_j$	$word_j$	pos_j
Bigram	$word_i \circ pos_i \circ word_j \circ pos_j$	$pos_i \circ word_j \circ pos_j$	$word_i \circ word_j \circ pos_j$
	$word_i \circ pos_i \circ pos_j$	$word_i \circ pos_i \circ word_j$	$word_i \circ word_j$
	$pos_i \circ pos_j$	$word_i \circ pos_j$	$pos_i \circ word_j$
Surrounding	$pos_i \circ pos_{i+1} \circ pos_{j-1} \circ pos_j$	$pos_{i-1} \circ pos_i \circ pos_{j-1} \circ pos_j$	$pos_i \circ pos_{i+1} \circ pos_j \circ pos_{j+1}$
	$pos_{i-1} \circ pos_i \circ pos_j \circ pos_{j+1}$	$pos_{i-1} \circ pos_i \circ pos_{j-1}$	$pos_{i-1} \circ pos_i \circ pos_{j+1}$
	$pos_i \circ pos_{i+1} \circ pos_{j-1}$	$pos_i \circ pos_{i+1} \circ pos_{j+1}$	$pos_{i-1} \circ pos_{j-1} \circ pos_j$
	$pos_{i-1} \circ pos_j \circ pos_{j+1}$	$pos_{i+1} \circ pos_{j-1} \circ pos_j$	$pos_{i+1} \circ pos_j \circ pos_{j+1}$
	$pos_i \circ pos_{j-1} \circ pos_j$	$pos_i \circ pos_j \circ pos_{j+1}$	$pos_{i-1} \circ pos_i \circ pos_j$
	$pos_i \circ pos_{i+1} \circ pos_j$		

Table 1: Feature templates for the word-pair classification model.

tive product:

$$\begin{aligned} \tilde{\mathbf{y}} &= \underset{\mathbf{y}}{\operatorname{argmax}}\, \mathbf{s}(\mathbf{x}, \mathbf{y}) \\ &= \underset{\mathbf{y}}{\operatorname{argmax}} \prod_{(i,j) \in \mathbf{y}} \mathcal{C}(i, j) \end{aligned} \tag{4}$$

Where y is searched from the set of well-formed dependency trees.

In our work we choose a real-valued ME classifier. Here we give the calculation of dependency probability $\mathcal{C}(i, j)$. We use $\mathbf{w}$ to denote the parameter vector of the ME model, and $\mathbf{f}(i, j, r)$ to denote the feature vector for the *assumption* that the word pair i and j has a dependency relationship r. The symbol r indicates the supposed classification result, where $r = +$ means we suppose it as a dependency edge and $r = -$ means the contrary. A feature $\mathbf{f}_k(i, j, r) \in \mathbf{f}(i, j, r)$ equals 1 if it is activated by the assumption and equals 0 otherwise. The dependency probability can then be defined as:

$$\begin{aligned} \mathcal{C}(i, j) &= \frac{exp(\mathbf{w} \cdot \mathbf{f}(i, j, +))}{\sum_r exp(\mathbf{w} \cdot \mathbf{f}(i, j, r))} \\ &= \frac{exp(\sum_k \mathbf{w}_k \times \mathbf{f}_k(i, j, +))}{\sum_r exp(\sum_k \mathbf{w}_k \times \mathbf{f}_k(i, j, r))} \end{aligned} \tag{5}$$

2.2 Features for Classification

The feature templates for the classifier are similar to those of 1st-ordered MST model (McDonald et al., 2005a). [1] Each feature is composed of some words and POS tags surrounded word i and/or word j, as well as an optional distance representations between this two words. Table shows the feature templates we use.

Previous graph-based dependency models usually use the index distance of word i and word j to enrich the features with word distance information. However, in order to utilize some syntax information between the pair of words, we adopt the syntactic distance representation of (Collins, 1996), named *Collins distance* for convenience. A Collins distance comprises the answers of 6 questions:

- Does word i precede or follow word j?
- Are word i and word j adjacent?
- Is there a verb between word i and word j?
- Are there 0, 1, 2 or more than 2 commas between word i and word j?
- Is there a comma immediately following the first of word i and word j?
- Is there a comma immediately preceding the second of word i and word j?

Besides the original features generated according to the templates in Table 1, the enhanced features with Collins distance as postfixes are also used in training and decoding of the word-pair classifier.

2.3 Parsing Algorithm

We adopt logarithmic dependency probabilities in decoding, therefore the cumulative product of probabilities in formula 6 can be replaced by accumulation of logarithmic probabilities:

$$\begin{aligned} \tilde{\mathbf{y}} &= \underset{\mathbf{y}}{\operatorname{argmax}}\, \mathbf{s}(\mathbf{x}, \mathbf{y}) \\ &= \underset{\mathbf{y}}{\operatorname{argmax}} \prod_{(i,j) \in \mathbf{y}} \mathcal{C}(i, j) \\ &= \underset{\mathbf{y}}{\operatorname{argmax}} \sum_{(i,j) \in \mathbf{y}} log(\mathcal{C}(i, j)) \end{aligned} \tag{6}$$

Thus, the decoding algorithm for 1st-ordered MST model, such as the Chu-Liu-Edmonds algorithm

[1] We exclude the *in between* features of McDonald et al. (2005a) since preliminary experiments show that these features bring no improvement to the word-pair classification model.

Algorithm 1 Dependency Parsing Algorithm.

```
1:  Input: sentence $\mathbf{x}$ to be parsed
2:  for $\langle i, j\rangle \subseteq \langle 1, |\mathbf{x}|\rangle$ in topological order do
3:      $\mathbf{buf} \leftarrow \emptyset$
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 DERIV$(l, r)$ into $\mathbf{buf}$
7:              insert DERIV$(r, l)$ into $\mathbf{buf}$
8:      $\mathbf{V}[i, j] \leftarrow$ top $K$ derivations of $\mathbf{buf}$
9:  Output: the best derivation of $\mathbf{V}[1, |\mathbf{x}|]$
10: function DERIV($p$, $c$)
11:     $d \leftarrow p \cup c \cup \{(p \cdot root, c \cdot root)\}$   ▷ new derivation
12:     $d \cdot evl \leftarrow$ EVAL$(d)$                ▷ evaluation function
13:     return $d$
```

used in McDonald et al. (2005b), is also applicable here. In this work, however, we still adopt the more general, bottom-up dynamic programming algorithm Algorithm 1 in order to facilitate the possible expansions. Here, $\mathbf{V}[i, j]$ contains the candidate parsing segments of the span $[i, j]$, and the function EVAL(d) accumulates the scores of all the edges in dependency segment d. 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).

3 Projected Classification Instance

After the introduction of the word-pair classification model, we now describe the extraction of projected dependency instances. In order to alleviate the effect of word alignment errors, we base the projection on the alignment matrix, a compact representation of multiple GIZA++ (Och and Ney, 2000) results, rather than a single word alignment in previous dependency projection works. Figure 2 shows an example.

Suppose a bilingual sentence pair, composed of a source sentence $\mathbf{e}$ and its target translation $\mathbf{f}$. $\mathbf{y_e}$ is the parse tree of the source sentence. $\mathbf{A}$ is the alignment matrix between them, and each element $\mathbf{A}_{i,j}$ denotes the degree of the alignment between word $\mathbf{e}_i$ and word $\mathbf{f}_j$. We define a boolean-valued function $\delta(\mathbf{y}, i, j, r)$ to investigate the dependency relationship of word i and word j in parse tree $\mathbf{y}$:

$$\delta(\mathbf{y}, i, j, r) = \begin{cases} 1 & \begin{array}{l} (i, j) \in \mathbf{y} \textbf{ and } r = + \\ \quad\textbf{or} \\ (i, j) \notin \mathbf{y} \textbf{ and } r = - \end{array} \\ & \\ 0 & \textbf{otherwise} \end{cases} \tag{7}$$

Then the score that word i and word j in the target sentence $\mathbf{y}$ forms a projected dependency edge,

	I	saw	him	with	a	telescope
wo	0.90	0.00	0.15	0.00	0.05	0.00
yong	0.05	0.00	0.00	0.95	0.05	0.00
wangyuanjing	0.00	0.10	0.00	0.05	0.10	0.95
kanjian	0.00	0.85	0.00	0.15	0.10	0.00
le	0.00	0.30	0.00	0.00	0.00	0.05
ta	0.05	0.00	0.95	0.00	0.00	0.00

Figure 2: The word alignment matrix between a Chinese sentence and its English translation. Note that probabilities need not to be normalized across rows or columns.

$\mathbf{s}_+(i, j)$, can be defined as:

$$\mathbf{s}_+(i, j) = \sum_{i', j'} \mathbf{A}_{i,i'} \times \mathbf{A}_{j,j'} \times \delta(\mathbf{y_e}, i', j', +) \tag{8}$$

The score that they do not form a projected dependency edge can be defined similarly:

$$\mathbf{s}_-(i, j) = \sum_{i', j'} \mathbf{A}_{i,i'} \times \mathbf{A}_{j,j'} \times \delta(\mathbf{y_e}, i', j', -) \tag{9}$$

Note that for simplicity, the condition factors $\mathbf{y_e}$ and $\mathbf{A}$ are omitted from these two formulas. We finally define the probability of the supposed projected dependency edge as:

$$\mathcal{C}_p(i, j) = \frac{exp(\mathbf{s}_+(i, j))}{exp(\mathbf{s}_+(i, j)) + exp(\mathbf{s}_-(i, j))} \tag{10}$$

The probability $\mathcal{C}_p(i, j)$ is a real value between 0 and 1. Obviously, $\mathcal{C}_p(i, j) = 0.5$ indicates the most ambiguous case, where we can not distinguish between positive and negative at all. On the other hand, there are as many as $2|\mathbf{f}|(|\mathbf{f}|-1)$ candidate projected dependency instances for the target sentence $\mathbf{f}$. Therefore, we need choose a threshold b for $\mathcal{C}_p(i, j)$ to filter out the ambiguous instances: the instances with $\mathcal{C}_p(i, j) > b$ are selected as the positive, and the instances with $\mathcal{C}_p(i, j) < 1 - b$ are selected as the negative.

4 Boosting an MST Parser

The classifier can be used to boost a existing parser trained on human-annotated trees. We first establish a unified framework for the enhanced parser. For a sentence to be parsed, $\mathbf{x}$, the enhanced parser selects the best parse $\tilde{\mathbf{y}}$ according to both the baseline model $\mathbb{B}$ and the projected classifier $\mathbb{C}$.

$$\tilde{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}}[\mathbf{s}_{\mathbb{B}}(\mathbf{x}, \mathbf{y}) + \lambda \mathbf{s}_{\mathbb{C}}(\mathbf{x}, \mathbf{y})] \tag{11}$$

Here, $\mathbf{s}_{\mathbb{B}}$ and $\mathbf{s}_{\mathbb{C}}$ denote the evaluation functions of the baseline model and the projected classifier, respectively. The parameter λ is the relative weight of the projected classifier against the baseline model.

There are several strategies to integrate the two evaluation functions. For example, they can be integrated deeply at each decoding step (Carreras et al., 2008; Zhang and Clark, 2008; Huang, 2008), or can be integrated shallowly in a reranking manner (Collins, 2000; Charniak and Johnson, 2005). As described previously, the score of a dependency tree given by a word-pair classifier can be factored into each candidate dependency edge in this tree. Therefore, the projected classifier can be integrated with a baseline model deeply at each dependency edge, if the evaluation score given by the baseline model can also be factored into dependency edges.

We choose the 2nd-ordered MST model (McDonald and Pereira, 2006) as the baseline. Especially, the effect of the Collins distance in the baseline model is also investigated. The relative weight λ is adjusted to maximize the performance on the development set, using an algorithm similar to minimum error-rate training (Och, 2003).

5 Related Works

5.1 Dependency Parsing

Both the graph-based (McDonald et al., 2005a; McDonald and Pereira, 2006; Carreras et al., 2006) and the transition-based (Yamada and Matsumoto, 2003; Nivre et al., 2006) parsing algorithms are related to our word-pair classification model.

Similar to the graph-based method, our model is factored on dependency edges, and its decoding procedure also aims to find a maximum spanning tree in a fully connected directed graph. From this point, our model can be classified into the graph-based category. On the training method, however, our model obviously differs from other graph-based models, that we only need a set of word-pair dependency instances rather than a regular dependency treebank. Therefore, our model is more suitable for the partially bracketed or noisy training corpus.

The most apparent similarity between our model and the transition-based category is that they all need a classifier to perform classification conditioned on a certain configuration. However, they differ from each other in the classification results. The classifier in our model predicates a dependency probability for each pair of words, while the classifier in a transition-based model gives a possible next transition operation such as *shift* or *reduce*. Another difference lies in the factorization strategy. For our method, the evaluation score of a candidate parse is factorized into each dependency edge, while for the transition-based models, the score is factorized into each transition operation.

Thanks to the reminding of the third reviewer of our paper, we find that the pairwise classification schema has also been used in Japanese dependency parsing (Uchimoto et al., 1999; Kudo and Matsumoto, 2000). However, our work shows more advantage in feature engineering, model training and decoding algorithm.

5.2 Dependency Projection

Many works try to learn parsing knowledge from bilingual corpora. Lü et al. (2002) aims to obtain Chinese bracketing knowledge via ITG (Wu, 1997) alignment. Hwa et al. (2005) and Ganchev et al. (2009) induce dependency grammar via projection from aligned bilingual corpora, and use some thresholds to filter out noise and some hand-written rules to handle heterogeneity. Smith and Eisner (2009) perform dependency projection and annotation adaptation with Quasi-Synchronous Grammar features. Jiang and Liu (2009) refer to alignment matrix and a dynamic programming search algorithm to obtain better projected dependency trees.

All previous works for dependency projection (Hwa et al., 2005; Ganchev et al., 2009; Smith and Eisner, 2009; Jiang and Liu, 2009) need complete projected trees to train the projected parsers. Because of the free translation, the word alignment errors, and the heterogeneity between two languages, it is reluctant and less effective to project the dependency tree completely to the target language sentence. On the contrary, our dependency projection strategy prefer to extract a set of dependency instances, which coincides our model's demand for training corpus. An obvious advantage of this strategy is that, we can select an appropriate filtering threshold to obtain dependency instances of good quality.

In addition, our word-pair classification model can be integrated deeply into a state-of-the-art MST dependency model. Since both of them are

Corpus	Train	Dev	Test
WSJ (section)	2-21	22	23
CTB 5.0 (chapter)	others	301-325	271-300

Table 2: The corpus partition for WSJ and CTB 5.0.

factorized into dependency edges, the integration can be conducted at each dependency edge, by weightedly averaging their evaluation scores for this dependency edge. This strategy makes better use of the projected parser while with faster decoding, compared with the cascaded approach of Jiang and Liu (2009).

6 Experiments

In this section, we first validate the word-pair classification model by experimenting on human-annotated treebanks. Then we investigate the effectiveness of the dependency projection by evaluating the projected classifiers trained on the projected classification instances. Finally, we report the performance of the integrated dependency parser which integrates the projected classifier and the 2nd-ordered MST dependency parser. We evaluate the parsing accuracy by the precision of lexical heads, which is the percentage of the words that have found their correct parents.

6.1 Word-Pair Classification Model

We experiment on two popular treebanks, the Wall Street Journal (WSJ) portion of the Penn English Treebank (Marcus et al., 1993), and the Penn Chinese Treebank (CTB) 5.0 (Xue et al., 2005). The constituent trees in the two treebanks are transformed to dependency trees according to the head-finding rules of Yamada and Matsumoto (2003). For English, we use the automatically-assigned POS tags produced by an implementation of the POS tagger of Collins (2002). While for Chinese, we just use the gold-standard POS tags following the tradition. Each treebank is splitted into three partitions, for training, development and testing, respectively, as shown in Table 2.

For a dependency tree with n words, only $n-1$ positive dependency instances can be extracted. They account for only a small proportion of all the dependency instances. As we know, it is important to balance the proportions of the positive and the negative instances for a batched-trained classifier. We define a new parameter r to denote the ratio of the negative instances relative to the positive ones.

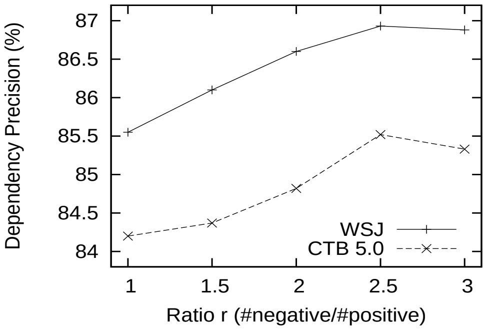


Figure 3: Performance curves of the word-pair classification model on the development sets of WSJ and CTB 5.0, with respect to a series of ratio r.

Corpus	System	P %
WSJ	Yamada and Matsumoto (2003)	90.3
	Nivre and Scholz (2004)	87.3
	1st-ordered MST	90.7
	2nd-ordered MST	91.5
	our model	86.8
CTB 5.0	1st-ordered MST	86.53
	2nd-ordered MST	87.15
	our model	82.06

Table 3: Performance of the word-pair classification model on WSJ and CTB 5.0, compared with the current state-of-the-art models.

For example, $r = 2$ means we reserve negative instances two times as many as the positive ones.

The MaxEnt toolkit by Zhang [2] is adopted to train the ME classifier on extracted instances. We set the gaussian prior as 1.0 and the iteration limit as 100, leaving other parameters as default values. We first investigate the impact of the ratio r on the performance of the classifier. Curves in Figure 3 show the performance of the English and Chinese parsers, each of which is trained on an instance set corresponding to a certain r. We find that for both English and Chinese, maximum performance is achieved at about $r = 2.5$. [3] The English and Chinese classifiers trained on the instance sets with $r = 2.5$ are used in the final evaluation phase. Table 3 shows the performances on the test sets of WSJ and CTB 5.0.

We also compare them with previous works on the same test sets. On both English and Chinese, the word-pair classification model falls behind of the state-of-the-art. We think that it is probably

[2] http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

[3] We did not investigate more fine-grained ratios, since the performance curves show no dramatic fluctuation along with the alteration of r.

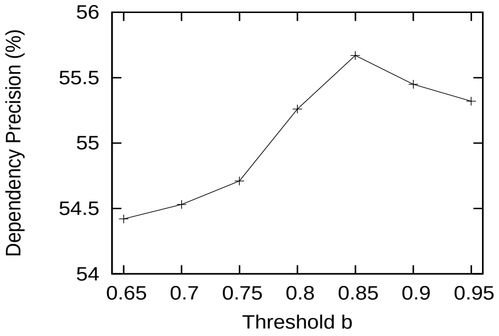


Figure 4: The performance curve of the word-pair classification model on the development set of CTB 5.0, with respect to a series of threshold b.

due to the local optimization of the training procedure. Given complete trees as training data, it is easy for previous models to utilize structural, global and linguistical information in order to obtain more powerful parameters. The main advantage of our model is that it doesn't need complete trees to tune its parameters. Therefore, if trained on instances extracted from human-annotated treebanks, the word-pair classification model would not demonstrate its advantage over existed state-of-the-art dependency parsing methods.

6.2 Dependency Projection

In this work we focus on the dependency projection from English to Chinese. We use the FBIS Chinese-English bitext as the bilingual corpus for dependency projection. It contains 239K sentence pairs with about 6.9M/8.9M words in Chinese/English. Both English and Chinese sentences are tagged by the implementations of the POS tagger of Collins (2002), which trained on WSJ and CTB 5.0 respectively. The English sentences are then parsed by an implementation of 2nd-ordered MST model of McDonald and Pereira (2006), which is trained on dependency trees extracted from WSJ. The alignment matrixes for sentence pairs are generated according to (Liu et al., 2009).

Similar to the ratio r, the threshold b need also be assigned an appropriate value to achieve a better performance. Larger thresholds result in better but less classification instances, the lower coverage of the instances would hurt the performance of the classifier. On the other hand, smaller thresholds lead to worse but more instances, and too much noisy instances will bring down the classifier's discriminating power.

We extract a series of classification instance sets with different thresholds. Then, on each instance set we train a classifier and test it on the development set of CTB 5.0. Figure 4 presents the experimental results. The curve shows that the maximum performance is achieved at the threshold of about 0.85. The classifier corresponding to this threshold is evaluated on the test set of CTB 5.0, and the test set of CTB 2.0 determined by Hwa et al. (2005). Table 4 shows the performance of the projected classifier, as well as the performance of previous works on the corresponding test sets. The projected classifier significantly outperforms previous works on both test sets, which demonstrates that the word-pair classification model, although falling behind of the state-of-the-art on human-annotated treebanks, performs well in projected dependency parsing. We give the credit to its good collaboration with the word-pair classification instance extraction for dependency projection.

Corpus	System	P %
CTB 2.0	Hwa et al. (2005)	53.9
	our model	**56.9**
CTB 5.0	Jiang and Liu (2009)	53.28
	our model	**58.59**

Table 4: The performance of the projected classifier on the test sets of CTB 2.0 and CTB 5.0, compared with the performance of previous works on the corresponding test sets.

Corpus	Baseline P%	Integrated P%
CTB 1.0	82.23	83.70
CTB 5.0	87.15	87.65

Table 5: Performance improvement brought by the projected classifier to the baseline 2nd-ordered MST parsers trained on CTB 1.0 and CTB 5.0, respectively.

6.3 Integrated Dependency Parser

We integrate the word-pair classification model into the state-of-the-art 2nd-ordered MST model. First, we implement a chart-based dynamic programming parser for the 2nd-ordered MST model, and develop a training procedure based on the perceptron algorithm with averaged parameters (Collins, 2002). On the WSJ corpus, this parser achieves the same performance as that of McDonald and Pereira (2006). Then, at each derivation step of this 2nd-ordered MST parser, we weightedly add the evaluation score given by the projected classifier to the original MST evaluation score. Such a weighted summation of two evaluation scores provides better evaluation for candidate parses. The weight parameter λ is tuned by a minimum error-rate training algorithm (Och, 2003).

Given a 2nd-ordered MST parser trained on CTB 5.0 as the baseline, the projected classifier brings an accuracy improvement of about 0.5 points. For the baseline trained on the smaller CTB 1.0, whose training set is chapters 1-270 of CTB 5.0, the accuracy improvement is much significant, about 1.5 points over the baseline. It indicates that, the smaller the human-annotated treebank we have, the more significant improvement we can achieve by integrating the projecting classifier. This provides a promising strategy for boosting the parsing performance of resource-scarce languages. Table 5 summarizes the experimental results.

7 Conclusion and Future Works

In this paper, we first describe an intuitionistic method for dependency parsing, which resorts to a classifier to determine whether a word pair forms a dependency edge, and then propose an effective strategy for dependency projection, which produces a set of projected classification instances rather than complete projected trees. Although this parsing method falls behind of previous models, it can collaborate well with the word-pair classification instance extraction strategy for dependency projection, and achieves the state-of-the-art in projected dependency parsing. In addition, when integrated into a 2nd-ordered MST parser, the projected parser brings significant improvement to the baseline, especially for the baseline trained on smaller treebanks. This provides a new strategy for resource-scarce languages to train high-precision dependency parsers. However, considering its lower performance on human-annotated treebanks, the dependency parsing method itself still need a lot of investigations, especially on the training method of the classifier.

Acknowledgement

This project was supported by National Natural Science Foundation of China, Contract 60736014, and 863 State Key Project No. 2006AA010108. We are grateful to the anonymous reviewers for their thorough reviewing and valuable suggestions. We show special thanks to Dr. Rebecca Hwa for generous help of sharing the experimental data. We also thank Dr. Yang Liu for sharing the codes of alignment matrix generation, and Dr. Liang Huang for helpful discussions.

References

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*.

Xavier Carreras, Mihai Surdeanu, and Lluis Marquez. 2006. Projective dependency parsing with perceptron. In *Proceedings of the CoNLL*.

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

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

Michael Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of ACL*.

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

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

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING*, pages 340–345.

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

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

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

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

Rebecca Hwa, 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.

Wenbin Jiang and Qun Liu. 2009. Automatic adaptation of annotation standards for dependency parsing using projected treebank as source corpus. In *Proceedings of IWPT*.

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 the 47th ACL*.

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

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the ACL*.

Taku Kudo and Yuji Matsumoto. 2000. Japanese dependency structure analysis based on support vector machines. In *Proceedings of the EMNLP*.

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

Yajuan Lü, Sheng Li, Tiejun Zhao, and Muyun Yang. 2002. Learning chinese bracketing knowledge based on a bilingual language model. In *Proceedings of the COLING*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. In *Computational Linguistics*.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*, pages 81–88.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91–98.

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

J. Nivre and M. Scholz. 2004. Deterministic dependency parsing of english text. In *Proceedings of the COLING*.

Joakim Nivre, Johan Hall, Jens Nilsson, Gulsen Eryigit, and Svetoslav Marinov. 2006. Labeled pseudoprojective dependency parsing with support vector machines. In *Proceedings of CoNLL*, pages 221–225.

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

Franz Joseph Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the ACL*, pages 160–167.

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

Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 1999. Japanese dependency structure analysis based on maximum entropy models. In *Proceedings of the EACL*.

Vladimir N. Vapnik. 1998. Statistical learning theory. In *A Wiley-Interscience Publication*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*.

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

Yue Zhang and Stephen Clark. 2008. Joint word segmentation and pos tagging using a single perceptron. In *Proceedings of the ACL*.