Multi-stage Chinese Dependency Parsing Based on Dependency Direction

Wenjing Lang **Qiaoli Zhou** **Guiping Zhang** **Dongfeng Cai**
Knowledge Engineering Research Center
Shenyang Aerospace University
110136, Shenyang, China
{ langlang_2671@126.com, qiaoli_z@yahoo.com.cn,
zgp@ge-soft.com,caidf@vip.163.com }

Abstract

This paper presents a novel method for multi-stage dependency parsing based on dependency direction. In the method, dependency parsing processes are divided into multiple sub-stages, and each stage is in a sequential pattern, which makes it easier to take applicable solutions for different issues in dependency parsing. Meanwhile, dependency parsing in the previous stage provides a clearer context for next stage. Furthermore, due to the dependency direction, the proposed method has lower search complexity than that of classic graph-based methods. Experimental results show that compared with common methods, the proposed method in this paper offers comparable accuracy and higher efficiency.

1 Introduction

Dependency parsing has been recognized as a basic technology in natural language processing, and has recently gained a wide interest. The advantage of the dependency analysis is that it provides a simple description of the syntactic relations in a sentence that could be easily understood and converted to semantic dependency description. Dependency parsing has seen a surge of interest lately for applications such as relation extraction (Culotta and Sorensen, 2004), machine translation (Ding and Palmer, 2005), ontology construction (Snow et al., 2005), and semantic role labeling (Surdeanu et al., 2008). The primary reasons for using dependency structures instead of more informative lexicalized phrase structures is that they are more efficient to learn and parse while still encoding much of the predicate-argument information needed in applications.

1.1 Dependency Grammar Theory

Dependency Grammar (DG) is a class of syntactic theories developed by Lucien Tesnière. While exploring the possibility of using DG as the base component of a transformational-generative formalism, Robinson (1970) gives four axioms for the well-formedness of dependency structures:

- One and only one element is independent;
- All others depend directly on some element;
- No elements depends directly on more than one other;
- If A depends directly on B and some element C intervenes between them (in linear order of string), then C depends directly on A or on B or some other intervening element.

These are also the foundation of the dependency grammar for use in computational linguistics. Sentence elements represent the nodes, and dependency between two nodes forms a dependency arc. The fourth axiom shows that if we put the words in their linear order, preceded by the root, the edges can be drawn above the words without crossings, which is also called Projective Dependency Grammar. Otherwise, when a node has multiple parent nodes or arcs contain crossings, it will inevitably undermine the first axiom and the fourth axiom, which is called Non-projective Dependency Grammar. In languages with more

flexible word order than Chinese, such as Czech, Dutch and Turkish, Non-projective Dependency Grammar is more common. Chinese strictly confirms to the Projective Dependency Grammar. In this paper, the characteristics of projective dependency, no crossing and single parent node of Chinese are utilized as limitations for building a dependency tree, so search complexity is reduced, and search efficiency is improved.

1.2 Related Works

Dependency structure analysis aims at getting the dependency structure of an input sentence automatically, and has recently gained a wide interest. A number of studies have been proposed for the analysis. The previous dependency analysis is divided into two approaches. One is graph-based approach and the other is transition-based approach.

In transition-based parsing, we learn a model for scoring transitions from one parser state to the next, conditioned on the parse history, and perform parsing by greedily taking the highest-scoring transition out of every parser state until we have derived a complete dependency graph. Transition-based parser has been proposed as a robust and efficient parser for syntactic parsing of unrestricted natural language text. The approach is represented, for example, by the models of Covington (2001), Yamada and Matsumoto (2003), and Nivre (2004). The main differences among the models focus on the actions and machine learning models (such as SVM or ME) for the score function for transition from one parsing state to the next.

In graph-based parsing, we learn a model for scoring possible dependency graphs for a given sentence, typically by factoring the graphs into their component arcs, and perform parsing by searching for the highest-scoring graph. As shown in the CoNLL 2006 shard tasks on dependency parsing, the performance of transition-based analyzer (Buchholz and Marsi, 2006) is better than the performance of graph-based analyzer in LAS. In following years, the graph-based analyzer has better performance. So the graph-based method hits the mainstream.

Eisner (1996) defines dependency parsing models where each word has a set of possible "senses" and the parser recovers the best joint assignment of syntax and senses. The complexity of the Eisner parser increases by factors of $O(n^3)$ time and $O(n^2)$ space. McDonald et al. (2005a) present a first-order model which assumes the dependency arcs are independent and the score of tree is the accumulation of all the arcs. They also extend the maximum spanning tree (MST) dependency parsing framework of McDonald et al. (2005b) to incorporate higher-order feature representations and allow dependency structures with multiple parents per word. Koo and Collins (2010) present algorithms for higher-order dependency parsing that are "third-order" in the sense that they can evaluate substructures containing three dependencies, and "efficient" in the sense that they require $O(n^4)$ time. The new parsers can utilize both sibling-style and grandchild-style interactions. Chen et al. (2009) propose a parsing model which uses all grandchildren nodes to compose high-order features, constrains the searching space by the beam-search strategy, and finds the approximately optimal dependency tree. In the CoNLL 2009 international evaluation task of multilingual syntactic and semantic dependency parsing, this method ranks first in the joint task, and third in the syntactic parsing task.

Nowadays, many researchers have investigated the use of bilingual constraints for parsing. Chen et al. (2010) propose a dependency parsing method that uses bilingual constrains to improve the accuracy of parsing bilingual texts. In their method, a target-side tree fragment is identified via word alignment and mapping rules that are automatically learned. Then it is verified by checking the subtree list that is collected from large scale automatically parsed data on the target side. Experiments on the translated portion of the Chinese Treebank show that the system outperforms monolingual parsers by 2.93 points for Chinese and 1.64 points for English.

These methods have good performance. However, the overall probability of each candidate tree shall be calculated at a time, so these methods have high search complexity and time complexity. This paper presents a multi-stage Chinese dependency parsing method based on dependency direction. In this paper, we introduce dependency direction, and dependency tree building is divided into multiple sub-stages. Furthermore, the Chinese dependency grammar is used as a limitation for building the dependency tree, so research

complexity is reduced, and search efficiency is improved.

The remainder of this paper is divided as follows: Section 2 gives the definition of the dependency model based on direction and the decoding algorithm, Section 3 describe our new method, Section 4 presents our experimental results, and Section 5 concludes.

2 Dependency Parsing Model

2.1 Model Definition

Follow McDonald et al. (1996), x is used to denote the sentence to be parsed, and x_i to denote the i-th word in the sentence. y denotes the dependency tree for sentence x, and $(i, j) \in y$ 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 dependency model is to determine whether any candidate word pair, x_i and x_j s.t. $1 \le i, j \le |x|$ and $i \neq j$, forms a dependency edge. The result $\delta(Y, ij)$ can be real valued:

$$\delta(Y, ij) = p \quad 0 \le p \le 1 \tag{1}$$

as produced by a conditional random fields (CRFs) model (John Lafferty et al., 2001). Y is the token which indicates (i, j) is a candidate edge, and p is a probability which indicates the degree the model supports the candidate edge (i, j).

We factorize the score of a dependency tree s(x, y) into its dependency edges; therefore, the task can be denoted to find a y' with maximum score in all candidate dependency trees. Where y is the set of candidate dependency trees.

$$y' = \arg\max_{y} s(x, y) = \arg\max_{y} \prod_{(i,j)\in y} \delta(Y, ij) \tag{2}$$

Here we give the calculation of dependency probability $\delta(Y, ij)$. We use λ to denote the weight from the CRF model, and $F(Y, ij)$ to denote the feature assuming that the word pair i and j has a dependency relationship R. R indicates the token results, where $R = Y$ means we suppose it as a dependency edge and $R = N$ means the contrary. A feature $F_k(R, ij) \in F(R, ij)$ equals 1 or 0.

$$\delta(Y, ij) = \frac{\exp(\lambda \times F(Y, ij))}{\sum_{R} \exp(\lambda \times F(Y, ij))} = \frac{\exp(\sum_{k} \lambda_k \times F_k(Y, ij))}{\sum_{R} \exp(\sum_{k} \lambda_k \times F_k(R, ij))} \tag{3}$$

Algorithm 1. Dependency Parsing Algorithm

```
Input: sentence x to be parsed
inti parse(candEdge)
Parse()
Output: the best parsing result

function Parse()
 if candEdge is empty
  return edgeSet
  (i, j) ← MaxEdge(candEdge)
  if Condition(i, j) and (i, j).prob ∈ [L,H] then
   edgeSet ← (i, j)
   DeleteNode(j)
   Parse()
  else
   DeleteEdge(i, j)
   Parse()

function MaxEdge(candEdge)
perform finding an edge with maximum probability in
the candEdge.

function Condition(i, j)
(i, j) confirms to the dependency axiom.

function DeleteNode(j)
delete all the edges which begins with j from the
candEdge.

function DeleteEdge (i, j)
delete (i, j) from the candEdge.
```

2.2 Parsing Algorithm

We design a dynamic programming algorithm shown in Algorithm 1 to search for the candidate parse with maximum score. This strategy alleviates the errors to some degree according to the dependency axiom. In Algorithm 1, candEdge

contains the candidate edges of the sentence. edgeSet is the final result set.

3 Multi-stage Chinese Dependency Parsing Based on Dependency Direction

3.1 Stage Division

This paper presents a method for multi-stage Chinese dependency parsing based on dependency direction for building a dependency tree. The multi-stage means that processes for building a dependency tree are divided into multiple sub-tasks according to different problems in dependency parsing.

Compared with other languages, Chinese has the characteristics of multiple syntactic elements for one word, recursion, flexible word order (Yu Shiwen, 1997), etc., so it is difficult to carry out direct automatic dependency parsing. In this paper, in accordance with the characteristics of Chinese, we divide Chinese dependency parsing tasks into the following stages:

- Dependency Direction Determination
- Forward and Backward partial Dependency Parsing Based on Dependency Direction
- Rule Processing
- Statistical Dependency Parsing Based on Dependency Direction

Each stage is in a sequential pattern, so it is easier to take applicable solutions for different issues in dependency parsing. Meanwhile, dependency parsing in the previous stage will provide a clearer context to control the search complexity in next stage.

3.2 Dependency Direction Determination

At present, Japanese dependency parsing is better than Chinese dependency parsing, which is mainly caused by regular word orders of Japanese, namely the principle that the head is located backwards. While Chinese has flexible word orders, so the head always changes.

In Figure 1, the head "是" of the word "他" is on the right of "他", and the head "是" of the word "教师" is on the left of "教师". Therefore, a dependency direction is given to each word in the sentence in this paper, i.e. backward dependency is defined when the head is on the right of the current word, and forward dependency is defined when the head is on the left of the current word; in particular, the word which is dependent on the root node is defined as backward dependency. Therefore, for a Chinese sentence, we can effectively predict the relative position of the head for each word of the sentence based on the determination of dependency direction for each word, so search complexity is greatly reduced. Furthermore, compared with dependency parsing of a complete sentence, dependency direction determination of a word is also a simple task. Therefore, we take the dependency direction determination as the first stage of multi-stage dependency parsing.

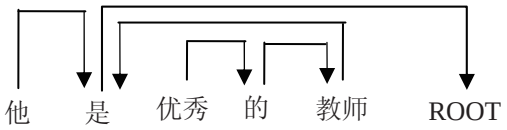

Figure 1. Example of Dependency Parsing

3.3 Forward and Backward partial Dependency Parsing Based on Dependency Direction

After the first stage, next stage is to find the head of each word in the sentence. The simplest method is to find the dependency between the current word and other words in the sentence. If the length of the sentence is n, there will be n*n-1 dependency instances. This method will cause the problems of excessive instances, oversize training files, etc. In this paper, we have predicted the relative position of the head for each word in the first stage, so we only need to combine word pairs in the corresponding dependency direction, i.e. for a word in the backward direction, we only combine the current word with latter words to form dependency, and the word in the forward direction is on the contrary. In this way, the instances of dependency will be decreased by half. Due to the introduction of dependency direction, dependency parsing is divided into forward parsing and backward parsing, so we shall build a forward model and a backward model respectively in this stage. During quantitative research of Chinese, Liu Haitao (2009) presented, “From the perspective of

dependency distance, the syntactic characteristics of Chinese not only lie in only one-third of the modifiers are located after their heads, but also in the much longer dependency distance when the heads are located after their modifiers than that of when the heads are located before their modifiers." In addition, the dependency distance will, to a great extent, affect the performance of dependency parsing results. Therefore, in this paper, we firstly accomplish backward dependency parsing, and then we accomplish forward dependency parsing on the basis of the prior analysis to alleviate the effects of excessive forward dependency distance.

Meanwhile, in order to avoid serious error accumulation, we set thresholds to optimize the dependency direction determining results and forward and backward dependency parsing results. Therefore, the second stage is called as forward and backward partial dependency parsing based on dependency direction.

3.4 Rule Processing

After the second stage, the heads of some words in the sentence are determined. Then, we can process the dependency relationship with clear grammar based on the determined heads. The procedures are as shown in Algorithm 2.

3.5 Statistical Dependency Parsing Based on Dependency Direction

Since we retain the dependency relationship with greater dependency probability after the backward dependency parsing model and the forward dependency parsing model in the second stage, it is not easy to determine the remaining dependency relationship by the backward dependency parsing model and the forward dependency parsing model in this stage. Thus, we redefine dependency parsing and introduce dependency determination based on dependency direction, and finally, we adopt logarithmic dependency probabilities:

$$\begin{aligned} y' &= \arg\max_{y} s(x, y) \\ &= \arg\max_{y}\Big(\sum_{(i,j)\in y} \alpha \log(\delta(Y, ij) + \theta) \\ &+ \sum_{j=1}^{|x|} (1-\alpha)\log(\gamma(j,t) + \theta)\Big) \end{aligned} \tag{4}$$

Algorithm 2. Rule Processing

Input: Forward and backward partial dependency parsing results based on dependency direction

Basic Operations:

① Input a sentence sequence $w_1/p_1/d_1\ w_2/p_2/d_2\ldots\ldots w_n/p_n/d_n$ (w_i is used to denote the i-th word in the sentence, p_i to denote the post-of-speech, and d_i to denote the head location of w_i, i s.t. $-1\le d_i\le n$ and $1\le i\le n$).

② If any $p_i\in$ {NN, NR, NT, PU}, then $d_i = n$. Rule processing ends, or go to ③;

③ If any d_i !=0, $j\in U$ (U denotes the location without dependency results, and j s.t. $1\le j\le n$), and there exists a unique $p_j\in$ {VA, VC, VE, VV, BA, LB}, then $d_j = 0$;

④ If $w_n\in$ {。, ？, ！}, and d_i =0, then $d_n = i$;

⑤ For the sequence $w_i \ldots w_j\ w_{j+1}\ w_{j+2} w_{j+3}\ldots w_k\ w_{k+1}\ w_{k+2}$, ($1\le i, j, k\le n$), if p_r =JJ, ($i\le r\le j$, or $j+3\le r\le k$), $w_{j+1}\in$ {NN, NR, NT,}, w_{j+2} = 、, $w_{k+1}\in$ {NN, NR, NT,}, w_{k+2} = 、, then $d_{j+2}=j+1$, $d_{k+1}=j+2$, $d_{k+2}=j+1$;

⑥ For the sequence $w_i\ w_{i+1}\ldots w_j$,($1\le i, j\le n$), if w_i = 《, w_j =》, then:

a. If d_k= 0, ($i+1\le k\le j-1$), then d_k= -1;
b. If d_i = -1, d_j != -1, then $d_i = d_j$; or, if d_j = -1, d_i != -1, then $d_j = d_i$. Rule processing ends, or go to c;
c. If $i+1\le k\le j-1$, only when there exists a unique k s.t. $d_k\le i-1$ or $j+1\le d_k$, $d_i = d_j = k$;

Basic Procedures: Do the above operations for all the sequences till the end of the file.

Here, $\gamma(j,t)$ represents the probability of x_j in the dependency direction t. According to the dependency direction determining model, α denotes the weight parameter, and θ is the smoothing factor.

4 Experiments

4.1 Preparation

In this paper, the training set is the training corpus of CoNLL2009 Share Task. It contains 22277 sentences with the average length of 27.34 words. The testing set contains 1762 sentences with the average length of 28.16 words.

Criteria for evaluating intermediate result performance of dependency parsing include precision (P), recall (R) and the comprehensive evaluation index F. The final dependency parsing

Type	Features
Unigrams	W_i, P_i, P_{i-3}, P_{i-2}, P_{i-1}, P_{i+1}, P_{i+2}, P_{i+3}, W_{i-3}, W_{i-2}, W_{i-1}, W_{i+1}, W_{i+2} , W_{i+3}
Surrounding	W_{i-2}/W_{i-1},W_{i-1}/W_i, W_i/W_{i+1}, W_{i-1}/W_i, P_i/P_{i+1}, P_{i+1}/P_{i+2}, W_{i-1}/P_{i-1}, W_{i+1}/P_{i+1}, $W_{i-1}/W_i/W_{i+1}$, $P_{i-1}/P_i/P_{i+1}$, $W_{i-2}/W_{i-1}/W_i/W_{i+1}$, $P_{i-2}/P_{i-1}/P_i/P_{i+1}$, $W_{i-3}/W_{i-2}/W_{i-1}/W_i$, $P_{i-3}/P_{i-2}/P_{i-1}/P_i$,$W_i/W_{i+1}/W_{i+2}/W_{i+3}$,$P_i/P_{i+1}/P_{i+2}/P_{i+3}$, $W_{i-4}/W_{i-3}/W_{i-2}/W_{i-1}/W_i$, $P_{i-4}/P_{i-3}/P_{i-2}/P_{i-1}/P_i$, $W_i/W_{i+1}/W_{i+2}/W_{i+3}/W_{i+4}$, $_i/P_{i+1}/P_{i+2}/P_{i+3}/P_{i+4}$

Table1. Feature Template for Dependency Direction Determination Model.

Type	Features
Unigrams	W_i, W_j, P_i, P_{i-2}, P_{i-1}, P_{i+1}, P_{i+2}, P_j, P_{j-2}, P_{j-1}, P_{j+1}, P_{j+2}, Dis=(i-j)
Surrounding	W_i/P_i, W_j/P_j, $W_i/P_i/W_j/P_j$, $W_j/P_j/P_i$, $W_j/P_j/W_i$, $W_i/P_i/P_j$, $W_i/P_i/W_j$, W_i/W_j, P_i/P_j, W_i/P_j, P_i/W_j, $P_i/P_{i+1}/P_{j-1}/P_j$, $P_i/P_{i+1}/P_j/P_{j+1}$, $P_{i-1}/P_i/P_{j-1}/P_j$, $P_{i-1}/P_i/P_j/P_{j+1}$, $P_{i-1}/P_i/P_{j-1}$, $P_{i-1}/P_i/P_{j+1}$, $P_i/P_{i+1}/P_{j-1}$, $P_i/P_{i+1}/P_{j+1}$, $P_{i-1}/P_{j-1}/P_j$, $P_{i-1}/P_j/P_{j+1}$, $P_{i+1}/P_{j-1}/P_j$, $P_{i+1}/P_j/P_{j+1}$, $P_i/P_{i-1}/P_j$, $P_i/P_j/P_{j+1}$, $P_{i-1}/P_i/P_j$, $P_i/P_{i+1}/P_j$, W_i/W_j/Dis, P_i/P_j/Dis

Table 3. Feature Template for Forward and Backward Models.

	N_1	N_2	N_3	P	R	F
Back wards	31010	31041	28895	93.18%	93.09%	93.13%
For wards	18610	18579	16464	88.47%	88.62%	88.54%
Total	49620	-	45359	91.41%	-	-

Table 2. Results of Direction Determination.

	N_1	N_2	N_3	P	R	F
Backward Analysis	8970	49620	8828	98.42%	17.79%	30.13%
Forward Analysis	13025	49620	12841	98.59%	25.88%	41.00%
Rule	14116	49620	13891	98.41%	27.99%	43.59%

Table 4. Results of the Second stage and the Third stage.

precision of dependency arcs is evaluated with unlabeled attachment score (UAS).

4.2 Direction Determination Results

In this paper, the CRF++4.9 toolkit which can obtain the global optimal solution is adopted to identify the dependency direction of a word to avoid decision greed caused by classifiers. The feature template for the direction determination model is shown in table 1, in which W represents the word in the input sequence, P represents POS, and subscript represents the location.

The results of direction determination are listed in Table 2. (N_1 refers to the number of words in results, N_2 to the number in test set, and N_3 to the right number in results.)

Table 2 shows that in Chinese, the number of backward dependency is much more than that of forward dependency, which conforms to what Liu Haitao (2009) found in Chinese quantitative research, i.e. “Chinese is a hybrid language whose heads are located backwards”. Meanwhile, determination results of backward dependency are better than that of forward dependency, which provides a basis for dependency parsing in next stage.

4.3 Forward and Backward Partial Dependency Parsing and Preliminary Rule Processing Based on Dependency Direction

The existing method for determining the dependency relationship between two words is that the dependency edges are translated into a binary classification task. In this paper, based on the advantage of the CRF model of comprehensiveness, we convert the determination of relationship between two words to a sequence identification task to build a forward dependency model and a backward dependency model. In the method, each element for the sentence is taken into consideration to offer an optimal identification result for each identified element, which overcomes the disadvantage of classifiers of non-comprehensiveness. The feature template is listed in Table 3, in which W represents the word in the input sequence, P represents POS, and the subscript represents the location. Dis represents the distance between the words i and j, in positive and passive. In order to avoid error accumulation, we set the thresholds according to the marginal probability provided by the CRF model, wherein we respectively take the results above 0.99 for

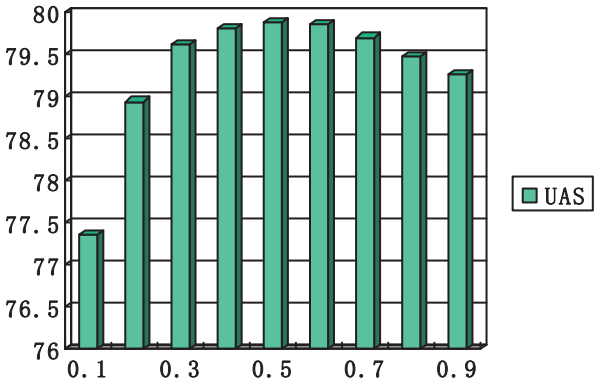

Figure 2. Dependency Performance with Respect to a series of Ratio α .

forward and backward dependency direction determination and the forward and backward dependency parsing results. Algorithm 1 is used for decoding, and experimental results of the second stage and the third stage are as shown in Table 4. The experimental results show that after the second stage, the recall rate is up to 27.99%, i.e. more than a quarter of words in the sentence have found their heads. Moreover, the experimental results have high precision.

4.4 Statistical Dependency Parsing Based on Dependency Direction

The main work in this stage is to use forward and backward dependency models to accomplish dependency parsing of the remaining words in the sentence. The decoding probability is calculated by Algorithm 1. We first investigate the impact of the α on the performance of the parsing. Figure 1 shows the dependency performance, each of which is corresponding to certain α . We find that, maximum performance is achieved at about α = 0.5. Therefore, α = 0.5 is used in the final evaluation phase, and θ is set as 0.0000001.

4.5 Final Result Analysis

Table5 shows the final performance on the test sets of CoNLL2009. We also compare them with previous works on the same test set. Our system falls behind of the Chen system a little. We think that it is probably caused by error accumulation. The main advantage of our model is that compared

System	UAS
Chen(2009)	80.38%
Our model	80.21%

Table 5. Final Performance of Parsing compared with the Current state-of-the-art System.

	N_1	N_2	N_3	P	R	F
Forward Analysis	18552	18579	14180	76.43%	76.32%	76.39%
Backward Analysis	31068	31041	25628	82.49%	82.56%	82.53%

Table 6. Forward and Backward Dependency Parsing Results.

with $O(n^4)$ in Chen(2009) system, the time complexity in this paper is $O(n^2)$, so costs of time for search are reduced. Meanwhile, due to the introduction of the dependency direction, costs of spaces for search are also reduced; the multi-stage method is also favorable for partial dependency parsing. Therefore, our method in this paper is effective.

In order to further analyze the experimental results, we test the final results of forward and backward dependency parsing, as shown in Table 6 which shows that the backward dependency parsing results are better than the forward dependency results. It indicates that the difficulties in Chinese dependency parsing focus on the forward dependency parsing.

5 Conclusion and Future Work

This paper presents a novel multi-stage Chinese dependency parsing method based on dependency direction. In this method, dependency parsing processes are divided into multiple sub-stages, and dependency parsing in the previous stage provides a clearer context for next stage. Furthermore, due to the introduction of the dependency direction and the use of the characteristics of Chinese dependency grammar, the complexities of time and space are effectively controlled on the premise of high dependency parsing precision.

However, the existing dependency parsing results in each stage in this paper are applied to the extent of dependency axioms. Therefore, our future work will focus on the application of the existing dependency parsing results. Meanwhile, we shall

further explore the syntax information containing in the dependency direction to provide more effective assistance for dependency parsing.

References

Culotta, A. and Sorensen, J. 2004. Dependency tree kernels for relation extraction. In *ACL2004*, pp. 423-429.

Ding, Y. and Palmer, M. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *ACL2005*, pp. 541-548.

Snow, R., Jurafsky, D. and Ng, A. Y. 2005. Learning syntactic pattern for automatic hypernym discovery. In *NIPS2005*, pp. 1297-1304.

Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L. and Nivre, J. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL2008*, pp. 159-197.

Robinson, J. J. 1970. Dependency structures and transformational rules. *Language*, 46(2):259-285.

Michael A. Covington. 2001. A fundamental algorithm for dependency parsing. *Proceedings of the 39th Annual ACM Southeast Conference*.

Yamada, H. and Matsumoto, Y. 2003. Statistical dependency analysis with support vector machines. In *IWPT2003*, pp. 195-206.

Joakim Nivre and Mario Scholaz. 2004. Deterministic Dependency Parsing of English Text. *Proceedings of COLING*, pp. 64-70.

Buchholz, S. and Marsi, E. 2006. CoNLL-2006 shared task on multilingual dependency parsing. *In CoNLL2006*, pp. 149-164.

Eisner, J. 1996. Three new probabilistic models for dependency parsing: an exploration. In *COLING-96*, pp. 340-345.

Ryan McDonald, Koby Crammer and Fernando Pereira. 2005. Online Large-Margin Training of Dependency Parsers. *Association for Computational Linguistics (ACL)*.

Ryan McDonald and Fernando Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithms. *European Association for Computational Linguistics (EACL)*.

Koo, T. and Collins, M. 2010. Efficient third-order dependency parsers. In *ACL2010*, pp. 1-11.

Che, W., Li, Z., Li, Y., Guo, Y., Qin, B. and Liu, T. 2009. Multilingual dependency-based syntactic and semantic parsing. In *CoNLL2009*, pp. 49-54.

Chen, W. L., Kazama, J., and Torisawa, K. 2010. Bitext dependency parsing with bilingual subtree constraints. In *ACL2010*, pp. 21-29.

Yu Shiwen. 1997. The application of grammar in language processing. *Journal of the Applied Linguistics.* (4): 81-87.

Y. Cheng, M. Asahara and Y Matsumoto. 2006. Multi-lingual dependency parsing at NAIST. *Proc. Of CoNLL*. New York City, USA: pp. 191-195.

John Lafferty, Andrew McCallum, Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc of ICML*, pp. 282-289.

Liu Haitao. 2009. Dependency Grammar from theory to practice. *Technology Press*, pp. 255.