Chinese Dependency Parsing with Large Scale Automatically Constructed Case Structures

Kun Yu
Graduate School of Informatics,
Kyoto University, Japan
kunyu@nlp.kuee.kyoto-u.ac.jp

Daisuke Kawahara
National Institute of Information and Communications Technology, Japan
dk@nict.go.jp

Sadao Kurohashi
Graduate School of Informatics,
Kyoto University, Japan
kuro@i.kyoto-u.ac.jp

Abstract

This paper proposes an approach using large scale case structures, which are automatically constructed from both a small tagged corpus and a large raw corpus, to improve Chinese dependency parsing. The case structure proposed in this paper has two characteristics: (1) it relaxes the predicate of a case structure to be all types of words which behaves as a head; (2) it is not categorized by semantic roles but marked by the neighboring modifiers attached to a head. Experimental results based on Penn Chinese Treebank show the proposed approach achieved 87.26% on unlabeled attachment score, which significantly outperformed the baseline parser without using case structures.

1 Introduction

Case structures (i.e. predicate-argument structures) represent what arguments can be attached to a predicate, which are very useful to recognize the meaning of natural language text. Researchers have applied case structures to Japanese syntactic analysis and improved parsing accuracy successfully (Kawahara and Kurohashi, 2006(a); Abekawa and Okumura, 2006). However, few works focused on using case structures in Chinese parsing. Wu (2003) proposed an approach to learn the relations between verbs and nouns and applied these relations to a Chinese parser. Han et al. (2004) presented a method to acquire the sub-categorization of Chinese verbs and used them in a PCFG parser.

Normally, case structures are categorized by semantic roles for verbs. For example, Kawahara and Kurohashi (2006(b)) constructed Japanese case structures which were marked by post positions. Wu (2003) classified the Chinese verb-noun relations as ‘verb-object’ and ‘modifier-head’. In this paper, we propose a new type of Chinese case structure, which is different from those presented in previous work (Wu, 2003; Han et al., 2004; Kawahara and Kurohashi, 2006(a); Abekawa and Okumura, 2006) in two aspects:

(1) It relaxes the predicate of a case structure to be all types of words which behaves as a head;

(2) It is not categorized by semantic roles but marked by the neighboring modifiers attached to a head. The sibling modification information remembers the parsing history of a head node, which is useful to correct the parsing error such as a verb 看 (see) is modified by two nouns 电影 (film) and 简介 (introduction) as objects (see Figure 1).

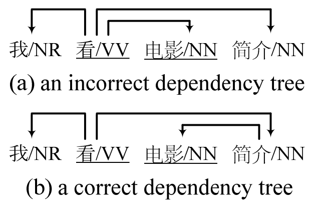
我/NR
看/VV
电影/NN
简介/NN
(a) an incorrect dependency tree
我/NR
看/VV
电影/NN
简介/NN
(b) a correct dependency tree

Figure 1. Dependency trees of an example sentence (I see the introduction of a film).

We automatically construct large scale case structures from both a small tagged corpus and a large raw corpus. Then, we apply the large scale case structures to a Chinese dependency parser to improve parsing accuracy.

The Chinese dependency parser using case structures is evaluated by Penn Chinese Treebank 5.1 (Xue et al., 2002). Results show that the automatically constructed case structures helped increase parsing accuracy by 2.13% significantly.

The rest of this paper is organized as follows: Section 2 describes the proposed Chinese case structure and the construction method in detail; Section 3 describes a Chinese dependency parser using constructed case structures; Section 4 lists the experimental results with a discussion in section 5; Related work is introduced in Section 6; Finally, Section 7 gives a brief conclusion and the direction of future work.

2 Chinese Case Structure and its Construction

2.1 A New Type of Chinese Case Structure

We propose a new type of Chinese case structure in this paper, which is represented as the combination of a case pattern and a case element (see Figure 2). Case element remembers the bi-lexical dependency relation between all types of head-modifier pairs, which is also recognized in previous work (Wu, 2003; Han et al., 2004; Kawahara and Kurohashi, 2006(a); Abekawa and Okumura, 2006). Case pattern keeps the pos-tag sequence of all the modifiers attached to a head to remember the parsing history of a head node.

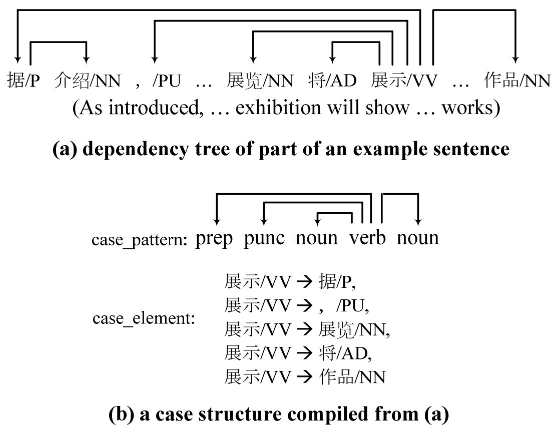
据/P 介绍/NN ，/PU … 展览/NN 将/AD 展示/VV … 作品/NN
(As introduced, … exhibition will show … works)
(a) dependency tree of part of an example sentence
case_pattern: prep punc noun verb noun
case_element:
展示/VV → 据/P,
展示/VV → ，/PU,
展示/VV → 展览/NN,
展示/VV → 将/AD,
展示/VV → 作品/NN
(b) a case structure compiled from (a)

Figure 2. An example of constructed case structure.

2.2 Construction Corpus

We use 9,684 sentences from Penn Chinese Treebank 5.1 as the tagged corpus, and 7,338,028 sentences written in simplified Chinese from Chinese Gigaword (Graff et al., 2005) as the raw corpus for Chinese case structure construction.

Before constructing case structures from the raw corpus, we need to get the syntactic analysis of it. First, we do word segmentation and pos-tagging for the sentences in Chinese Gigaword by a Chinese morphological analyzer (Nakagawa and Uchimoto, 2007). Then a Chinese deterministic syntactic analyzer (Yu et al., 2007) is used to parse the whole corpus.

To guarantee the accuracy of constructed case structures, we only use the sentences with less than *k* words from Chinese Gigaword. It is based on the assumption that parsing short sentences is more accurate than parsing long sentences. The performance of the deterministic parser used for analyzing Chinese Gigaword (see Figure 3) shows smaller *k* ensures better parsing quality but suffers from lower sentence coverage. Referring to Figure 2, we set *k* as 30.

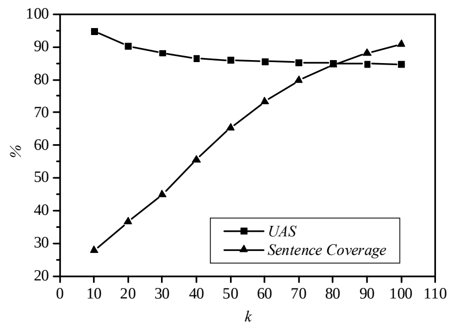

Figure 3. Performance of the deterministic parser with different *k* on 1,800 sentences[2].

2.3 Case Pattern Construction

A case pattern consists of a sequence of pos-tags indicating the order of all the modifiers attached to a head (see Figure 1), which can be represented as following.

$$cp_i = <[pos_m, pos_{m-1}, ..., pos_1]_l, [pos_1, ..., pos_{n-1}, pos_n]_r>$$

Here, $[pos_m, pos_{m-1}, ..., pos_1]_l$ means the pos-tag sequence of the modifiers attached to a head from the left side, and $[pos_1, ..., pos_{n-1}, pos_n]_r$ means the pos-tag sequence of the modifiers attached to a head from the right side.

We use the 33 pos-tags defined in Penn Chinese Treebank (Xue et al., 2002) to describe a case pattern, and make following modifications:

- group common noun, proper noun and pronoun together and mark them as ‘*noun*’;
- group predicative adjective and all the other verbs together and mark them as ‘*verb*’;
- only regard comma, pause, colon and semi-colon as punctuations and mark them as ‘*punc*’, and neglect other punctuations.

[2] *UAS* means unlabeled attachment score (Buchholz and Marsi, 2006). The sentences used for this evaluation are from Penn Chinese Treebank with gold word segmentation and pos-tag.

- group cardinal number and ordinal number together and mark them as '*num*';
- keep the original definition for other pos-tags but label them by new tags, such as labeling 'P' as 'prep' and labeling 'AD' as 'adv'.

The task of case pattern construction is to extract cp_i for each head from both the tagged corpus and the raw corpus. As we will introduce later, the Chinese dependency parser using case structures applies CKY algorithm for decoding. Thus the following substrings of cp_i are also extracted for each head as horizontal Markovization during case pattern construction.

$$< [pos_k, pos_{k-1}, ..., pos_1]_l, [pos_1, ..., pos_{j-1}, pos_j]_r > \quad \forall\, k \in [1, m], j \in [1, n]$$

$$< [pos_k, pos_{k-1}, ..., pos_1]_l >, \forall\, k \in [1, m]$$

$$< [pos_1, ..., pos_{j-1}, pos_j]_r >, \forall\, j \in [1, n]$$

2.4 Case Element Construction

As introduced in Section 2.1, a case element keeps the lexical preference between a head and its modifier. Therefore, the task of case element construction is to extract head-modifier pairs from both the tagged corpus and the raw corpus.

Although only the sentences with less than k (k=30) words from Chinese Gigaword are used as raw corpus to guarantee the accuracy, there still exist some dependency relations with low accuracy in these short sentences because of the non-perfect parsing quality. Therefore, we apply a head-modifier (HM) classifier to the parsed sentences from Chinese Gigaword to further extract head-modifier pairs with high quality. This HM classifier is based on SVM classification. Table 1 lists the features used in this classifier.

Feature	Description
Pos_{head}/ Pos_{mod}	Pos-tag pair of head and modifier
Distance	Distance between head and modifier
HasComma	If there exists comma between head and modifier, set as 1; otherwise as 0
HasColon	If there exists colon between head and modifier, set as 1; otherwise as 0
HasSemi	If there exists semi-colon between head and modifier, set as 1; otherwise as 0

Table 1. Features for HM classifier.

The HM classifier is trained on 3500 sentences from Penn Chinese Treebank with gold-standard word segmentation and pos-tag. All the sentences are parsed by the same Chinese deterministic parser used for Chinese Gigaword analysis. The correct dependency relations created by the parser are looked as positive examples and the left dependency relations are used as negative examples. TinySVM[3] is selected as the SVM toolkit. A polynomial kernel is used and degree is set as 2. Tested on 346 sentences, which are from Penn Chinese Treebank and parsed by the same deterministic parser with gold standard word segmentation and pos-tag, this HM classifier achieved 96.77% on precision and 46.35% on recall.

3 A Chinese Dependency Parser Using Case Structures

3.1 Parsing Model

We develop a lexicalized Chinese dependency parser to use constructed case structures. This parser gives a probability $P(T|S)$ to each possible dependency tree T of an input sentence $S=w_1,w_2,\ldots,w_n$ (w_i is a node representing a word with its pos-tag), and outputs the dependency tree T^* that maximizes $P(T|S)$ (see equation 1). CKY algorithm is used to decode the dependency tree from bottom to up.

$$T^* = \arg\max_T P(T \mid S) \tag{1}$$

To use case structures, $P(T|S)$ is divided into two parts (see equation 2): the probability of a sentence S generating a root node w_{ROOT}, and the product of the probabilities of a node w_i generating a case structure CS_i.

$$P(T \mid S) = P(w_{ROOT} \mid S) \times \prod_{i=1}^{m} P(CS_i \mid w_i) \tag{2}$$

As introduced in Section 2, a case structure CS_i is composed of a case pattern cp_i and a case element cm_i. Thus

$$\begin{aligned} P(CS_i \mid w_i) &= P(cp_i, cm_i \mid w_i) \\ &= P(cp_i \mid w_i) \times P(cm_i \mid w_i, cp_i) \end{aligned} \tag{3}$$

A case element cm_i consists of a set of dependencies $\{D_j\}$, in which each D_j is a tuple $<w_j, dis_j, comma_j>$. Here w_j means a modifier node, dis_j means the distance between w_j and its head, and $comma_j$ means the number of commas between w_j and its head. Assuming any D_j and D_k are independent of each other when they belong to the same case element, $P(cm_i|w_i,cp_i)$ can be written as

$$\begin{aligned} P(cm_i \mid w_i, cp_i) &= \prod_j P(D_j \mid w_i, cp_i) \\ &= \prod_j P(w_j, dis_j, comma_j \mid w_i, cp_i) \end{aligned} \tag{4}$$

[3] http://chasen.org/~taku/software/TinySVM/

Finally, $P(w_j, dis_j, comma_j \mid w_i, cp_i)$ is divided as

$$P(w_j, dis_j, comma_j \mid w_i, cp_i) \\ = P(w_j \mid w_i, cp_i) \times P(dis_j, comma_j \mid w_i, w_j, cp_i) \tag{5}$$

Maximum likelihood estimation is used to estimate $P(w_{ROOT}|S)$ on training data set with the smoothing method used in (Collins, 1996). The estimation of $P(cp_i|w_i)$, $P(w_j|w_i,cp_i)$, and $P(dis_j, comma_j|w_i,w_j,cp_i)$ will be introduced in the following subsections.

3.2 Estimating $P(cp_i|w_i)$ by Case Patterns

Three steps are used to estimate $P(cp_i|w_i)$ by maximum likelihood estimation using the constructed case patterns:

- Estimate $P(cp_i|w_i)$ only by the case patterns from the tagged corpus and represent it as $\hat{P}_{tagged}(cp_i \mid w_i)$;
- Estimate $P(cp_i|w_i)$ only by the case patterns from the raw corpus and represent it as $\hat{P}_{raw}(cp_i \mid w_i)$;
- Estimate $P(cp_i|w_i)$ by equation 6, in which $\lambda_{pattern}$ is calculated by equation 7 to set proper ratio for the probabilities estimated by the case patterns from different corpora.

$$\hat{P}(cp_i \mid w_i) \\ = \lambda_{pattern} \times \hat{P}_{tagged}(cp_i \mid w_i) + (1-\lambda_{pattern}) \times \hat{P}_{raw}(cp_i \mid w_i) \tag{6}$$

$$\lambda_{pattern} = \frac{\delta_{tagged} + \eta_{tagged} + \delta_{raw} + \eta_{raw}}{\delta_{tagged} + \eta_{tagged} + \delta_{raw} + \eta_{raw} + 1} \tag{7}$$

In equation 7, δ_{tagged} and δ_{raw} mean the occurrence of a lexicalized node w_i=<lex_i, pos_i> generating cp_i in the tagged or raw corpus, η_{tagged} and η_{raw} mean the occurrence of a back-off node w_i=<pos_i> generating cp_i in the tagged or raw corpus. To overcome the data sparseness problem, we not only apply the smoothing method used in (Collins, 1996) for a lexicalized head to back off it to its part-of-speech, but also assign a very small value to $P(cp_i|w_i)$ when there is no cp_i modifying w_i in the constructed case patterns.

3.3 Estimating $P(w_j|w_i,cp_i)$ and $P(dis_j, comma_j|w_i,w_j,cp_i)$ by Case Elements

To estimate $P(w_j|w_i,cp_i)$ and $P(dis_j, comma_j|w_i,w_j,cp_i)$ by maximum likelihood estimation, we also use three steps:

- Estimate the two probabilities only by the case elements from the tagged corpus and represent them as $\hat{P}_{tagged}(w_j \mid w_i, cp_i)$ and $\hat{P}_{tagged}(dis_j, comma_j \mid w_i, w_j, cp_i)$;
- Estimate the two probabilities only by the case elements from the raw corpus, and represent them as $\hat{P}_{raw}(w_j \mid w_i, cp_i)$ and $\hat{P}_{raw}(dis_j, comma_j \mid w_i, w_j, cp_i)$;
- Estimate $P(w_j|w_i,cp_i)$ and $P(dis_j, comma_j|w_i,w_j,cp_i)$ by equation 8 and equation 9.

$$\hat{P}(w_j \mid w_i, cp_i) \\ = \lambda_{element} \times \hat{P}_{tagged}(w_j \mid w_i, cp_i) + \\ (1-\lambda_{element}) \times \hat{P}_{raw}(w_j \mid w_i, cp_i) \tag{8}$$

$$\hat{P}(dis_j, comma_j \mid w_i, w_j, cp_i) \\ = \lambda_{element} \times \hat{P}_{tagged}(dis_j, comma_j \mid w_i, w_j, cp_i) + \\ (1-\lambda_{element}) \times \hat{P}_{raw}(dis_j, comma_j \mid w_i, w_j, cp_i) \tag{9}$$

The smoothing method used in (Collins, 1996) is applied during estimation.

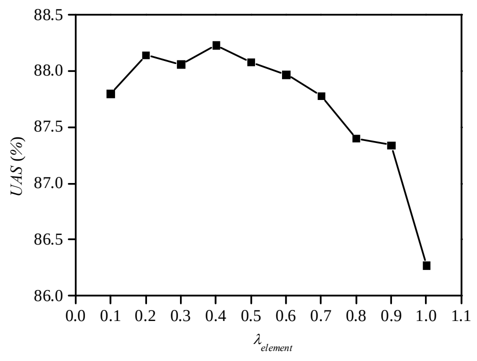

Figure 4. Parsing accuracy with different $\lambda_{element}$ on development data set.

In order to set proper ratio for the probabilities estimated by the case elements from different corpora, we use a parameter $\lambda_{element}$ in equation 8 and 9. The appropriate setting ($\lambda_{element}$ =0.4) is learned by a development data set (see Figure 4).

4 Evaluation Results

4.1 Experimental Setting

We use Penn Chinese Treebank 5.1 as data set to evaluate the proposed approach. 9,684 sentences from Section 001-270 and 400-931, which are also used for constructing case structures, are used as training data. 346 sentences from Section 271-300 are used as testing data. 334 sentences from Section 301-325 are used as development data. Penn2Malt[4] is used to transfer the phrase structure of Penn Chinese Treebank to dependency structure. Gold-standard word segmentation and pos-tag are applied in all the experiments.

[4] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html

Unlabeled attachment score (*UAS*) (Buchholz and Marsi, 2006) is used as evaluation metric. Because of the difficulty of assigning correct head to Chinese punctuation, we calculate *UAS* only on the dependency relations in which the modifier is not punctuation.

4.2 Results

Three parsers were evaluated in this experiment:

- '*baseline*': a parser not using case structures, where $P(T|S)$ is calculated by equation 11 and $P(w_j|w_i)$ and $P(dis_j, comma_j \,|w_i,w_j)$ are estimated by training data set only.

$$
\begin{aligned}
&P(T \mid S) \\
&= P(w_{ROOT} \mid S) \times \prod_{i,j \in [1,n], i \neq j} P(w_j, dis_j, comma_j \mid w_i) \\
&= P(w_{ROOT} \mid S) \times \prod_{i,j \in [1,n], i \neq j} (P(w_j \mid w_i) \times P(dis_j, comma_j \mid w_i, w_j))
\end{aligned} \tag{11}
$$

- '*w/ case elem*': a parser only using case element, which also calculates $P(T|S)$ by equation 11 but estimates $P(w_j|w_i)$ and $P(dis_j, comma_j \,|w_i,w_j)$ by constructed case elements.
- '*proposed*': the parser introduced in Section 3, which uses both case elements and case patterns.

The evaluation results on testing data set (see Table 2) shows the *proposed* parser achieved 87.26% on *UAS*, which was 2.13% higher than that of the *baseline* parser. This improvement is regarded as statistically significant (McNemar's test: p<0.0005). Besides, Table 2 shows only using case elements increased parsing accuracy by 1.30%. It means both case elements and case patterns gave help to parsing accuracy, and case elements contributed more in the proposed approach.

Parsing model	*baseline*	*w/ case elem*	*proposed*
UAS (%)	85.13	86.43 (+1.30)	**87.26 (+2.13)**

Table 2. Parsing accuracy of different parsing models.

Figure 5 and Figure 6 show the dependency trees of two example sentences created by both the *baseline* parser and the *proposed* parser. In Figure 5, *the baseline* parser incorrectly assigned 签订/NN (signing) as the head of 合作/NN (co-operation). However, after using the case element 项目/NN (project) $\rightarrow$ 合作/NN, the correct head of 合作/NN was found by the *proposed* parser. Figure 6 shows the *baseline* parser recognized 开幕/VV (opening) as the head of 据/P (as) incorrectly. But in the *proposed* parser, the probability of 开幕/VV generating the case pattern '[*prep*, *punc*, *prep*]$_l$' was much lower than that of 开幕/VV generating the case pattern '[*prep*]$_l$'. Therefore, the proposed parser rejected the incorrect dependency that 据/P modified 开幕/VV and got the correct head of 据/P as 展示/VV (show) successfully.

5 Discussion

5.1 Influence of the Number of Case Structures on Parsing Accuracy

During case structure construction, we only used the sentences with less than k (k=30) words from Chinese Gigaword as the raw corpus. Enlarging k will introduce more sentences from Chinese Gigaword and increase the number of case structures. Table 4 lists the number of case structures and parsing accuracy of the *proposed* parser on testing data set with different k[5]. It shows enlarging the number of case structures is a possible way to increase parsing accuracy. But simply setting larger k did not help parsing, because it decreased the parsing accuracy of Chinese Gigaword and consequently decreased the accuracy of constructed case structures. Using good parse selection (Reichart and Rappoport, 2007; Yates et al., 2006) on the syntactic analysis of Chinese Gigaword is a probable way to construct more case structures without decreasing their accuracy. We will consider about it in the future.

k	10	20	30	40
# of Case Element (M)	0.66	1.14	1.81	2.75
# of Case Pattern (M)	0.57	1.55	3.91	8.48
UAS (%)	85.16	86.42	87.26	87.07

Table 4. Case structure number and parsing accuracy with different k.

5.2 Influence of the Case Structure Construction Corpus on Parsing Accuracy

We also evaluated the *proposed* parser on testing data set using case structures constructed from different corpora.

Results (see Table 5) show that parsing accuracy was improved greatly only when using case structures constructed from both the two corpora. The case structures constructed from either of a

[5] Considering about the time expense of case structure construction, we only did test for $k \leq 40$.

single corpus only gave a little help to parsing. It is because among all the case structures used during testing (see Table 6), 19.57% case elements were constructed from the tagged corpus only and 54.18% case patterns were constructed from the raw corpus only. The incorrect head-modifier pairs extracted from Chinese Gigaword is a possible reason for the fact that some case elements only existing in the tagged corpus. Enhancing good parse selection on Chinese Gigaword could improve the quality of extracted head-modifier pairs and solve this problem. In addition, the strict definition of case pattern is a probable reason that makes more than half of the case patterns only exist in the raw corpus and 18.18% case patterns exist in neither of the two corpora. We will modify the representation of case pattern to make it more flexible to the number of modifiers to resolve this issue in the future.

Corpus	Tagged	Raw	Tagged+Raw
UAS (%)	85.25	85.90	87.26

Table 5. Parsing accuracy with case structures constructed from different corpora.

Corpus	Tagged	Raw	Tagged+Raw	None
% of case element	19.57	8.95	68.07	3.41
% of case pattern	0.03	54.18	27.61	18.18

Table 6. Ratio of case structures constructed from different corpora.

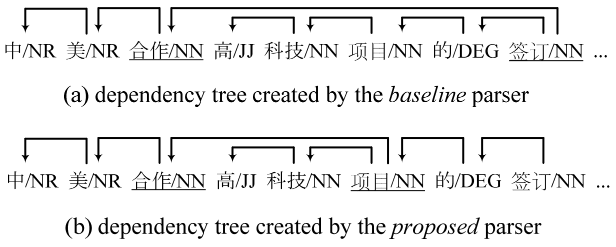
中/NR 美/NR 合作/NN 高/JJ 科技/NN 项目/NN 的/DEG 签订/NN ...
(a) dependency tree created by the *baseline* parser
中/NR 美/NR 合作/NN 高/JJ 科技/NN 项目/NN 的/DEG 签订/NN ...
(b) dependency tree created by the *proposed* parser

Figure 5. Dependency trees of an example sentence (The signing of China-US cooperation high tech project …).

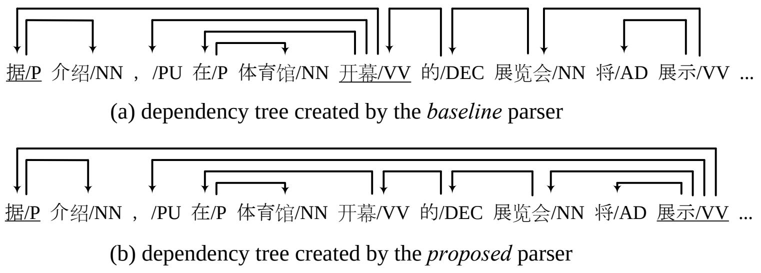
据/P 介绍/NN ， /PU 在/P 体育馆/NN 开幕/VV 的/DEC 展览会/NN 将/AD 展示/VV ...
(a) dependency tree created by the *baseline* parser
据/P 介绍/NN ， /PU 在/P 体育馆/NN 开幕/VV 的/DEC 展览会/NN 将/AD 展示/VV ...
(b) dependency tree created by the *proposed* parser

Figure 6. Dependency trees of an example sentence (As introduced, the exhibition opening in the stadium will show…).

5.3 Parsing Performance with Real Pos-tag

Gold standard word segmentation and pos-tag are applied in previous experiments. However, parsing accuracy will be affected by the incorrect word segmentation and pos-tag in the real applications. Currently, the best performance of Chinese word segmentation has achieved 99.20% on F-score, but the best accuracy of Chinese pos-tagging was 96.89% (Jin and Chen, 2008). Therefore, we think pos-tagging is more crucial for applying parser in real task compared with word segmentation. Considering about this, we evaluated the parsing models introduced in Section 4 with real pos-tag in this experiment.

Parsing model	*baseline*	*proposed*
UAS (%)	80.91	**82.90 (+1.99)**

Table 7. Parsing accuracy of different parsing models with real pos-tag.

An HMM-based pos-tagger is used to get pos-tag for testing sentences with gold word segmentation. The pos-tagger was trained on the same training data set described in Section 4.1 and achieved 93.70% F-score on testing data set. Results (see Table 7) show that even if with real pos-tags, the *proposed* parser still outperformed the *baseline* parser significantly. However, the results in Table 7 indicate that incorrect pos-tag affected the parsing accuracy of the *proposed* parser greatly. Some researchers integrated pos-tagging into parsing and kept n-best pos-tags to reduce the effect of pos-tagging errors on parsing accuracy (Cao et al., 2007). We will also consider about this in our future work.

6 Related Work

To our current knowledge, there were few works about using case structures in Chinese parsing, except for the work of Wu (2003) and Han et al. (2004). Compared with them, our proposed approach presents a new type of case structures for all kinds of head-modifier pairs, which not only recognizes bi-lexical dependency but also remembers the parsing history of a head node.

Parsing history has been used to improve parsing accuracy by many researchers (Yamada and Matsumoto, 2003; McDonald and Pereira, 2006). Yamada and Matsumoto (2003) showed that keeping a small amount of parsing history was useful to improve parsing performance in a shift-reduce parser. McDonald and Pereira (2006) expanded their first-order spanning tree model to be second-order by factoring the score of the tree into the sum of adjacent edge pair scores. In our proposed approach, the case patterns remember the neighboring modifiers for a head node like McDonald and Pereira's work. But it keeps all the parsing histories of a head, which is different from only keeping adjacent two modifiers in (McDonald and Pereira, 2006). Besides, to use the parsing histories in CKY decoding, our approach applies horizontal Markovization during case pattern construction. In general, the success of using case patterns in Chinese parsing in his paper proves again that keeping parsing history is crucial to improve parsing performance, no matter in which way and to which parsing model it is applied.

There were also some works that handled lexical preference for Chinese parsing in other ways. For example, Cheng et al. (2006) and Hall et al. (2007) applied shift-reduce deterministic parsing to Chinese. Sagae and Tsujii (2007) generalized the standard deterministic framework to probabilistic parsing by using a best-first search strategy. In these works, lexical preferences were introduced as features for predicting parsing action. Besides, Bikel and Chiang (2000) applied two lexicalized parsing models developed for English to Penn Chinese Treebank. Wang et al. (2005) proposed a completely lexicalized bottom-up generative parsing model to parse Chinese, in which a word-similarity-based smoothing was introduced to replace part-of-speech smoothing.

7 Conclusion and Future Work

This paper proposes an approach to use large scale case structures, which are automatically constructed from both a small tagged corpus and the syntactic analysis of a large raw corpus, to improve Chinese dependency parsing. The proposed case structures not only recognize the lexical preference between all types of head-modifier pairs, but also keep the parsing history of a head word. Experimental results show the proposed approach improved parsing accuracy significantly. Besides, although we only apply the proposed approach to Chinese dependency parsing currently, the same idea could be adapted to other languages easily because it doesn't use any language specific knowledge.

There are several future works under consideration, such as modifying the representation of case patterns to make it more robust, enhancing good parse selection on the analysis of raw corpus, and integrating pos-tagging into parsing model.

References

T.Abekawa and M.Okumura. 2006. Japanese Dependency Parsing Using Co-occurrence Information and a Combination of Case Elements. In *Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics 2006*. pp. 833-840.

D.Bikel. 2004. Intricacies of Collins' Parsing Model. *Computational Linguistics*, 30(4): 479-511.

D.Bikel and D.Chiang. 2000. Two Statistical Parsing Models Applied to the Chinese Treebank. In *Proceedings of the 2nd Chinese Language Processing Workshop*. pp. 1-6.

S.Buchholz and E.Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning*.

H.Cao et al.. 2007. Empirical Study on Parsing Chinese Based on Collins' Model. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguisitcs*. pp. 113-119.

Y.Cheng, M.Asahara and Y.Matsumoto. 2006. Multilingual Dependency Parsing at NAIST. In *Proceedings of the 10th Conference on Computational Natural Language Learning*. pp. 191-195.

M.Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*. pp. 184-191.

M.Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing. *Ph.D Thesis*. University of Pennsylvania.

D.Graff et al.. 2005. Chinese Gigaword Second Edition. *Linguistic Data Consortium*, Philadelphia.

J.Hall et al. 2007. Single Malt or Blended? A Study in Multilingual Parser Optimization. In *Proceedings of the shared task at the Conference on Computational Natural Language Learning 2007*. pp. 933-939.

X.Han et al.. 2004. Subcategorization Acquisition and Evaluation for Chinese Verbs. In *Proceedings of the 20th International Conference on Computational Linguistics*.

G.Jin and X.Chen. 2008. The Fourth International Chinese Language Processing Bakeoff: Chinese Word Segmentation, Named Entity Recognition and Chinese Pos Tagging. In *Proceedings of the 6th SIGHAN Workshop on Chinese Language Processing*.

D.Kawahara and S.Kurohashi. 2006 (a). A Fully-lexicalized Probabilistic Model for Japanese Syntactic and Case frame Analysis. In *Proceedings of the Human Language Technology conference - North American chapter of the Association for Computational Linguistics annual meeting 2006*. pp. 176-183.

D.Kawahara and S.Kurohashi. 2006 (b). Case Frame Compilation from the Web Using High-performance Computing. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*.

T.Kudo and Y.Matsumoto. 2002. Japanese Dependency Analysis Using Cascaded Chunking. In *Proceedings of the Conference on Natural Language Learning*. pp. 29-35.

R.McDonald and F.Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithm. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*.

R.McDonald and J.Nivre. 2007. Characterizing the Errors of Data-driven Dependency Parsing Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing Conference on Computational Natural Language Learning 2007*.

T.Nakagawa and K.Uchimoto. 2007. A Hybrid Approach to Word Segmentation and POS Tagging. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. pp. 217-220.

R.Reichart and A.Rappoport. 2007. An Ensemble Method for Selection of High Quality Parses. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. pp. 408-415.

K.Sagae and J.Tsujii. 2007. Dependency Parsing and Domain Adaptation with LR Models and Parser Ensembles. In *Proceedings of the shared task at the Conference on Computational Natural Language Learning 2007*. pp. 1044-1050.

Q.Wang, D.Schuurmans, and D.Lin. 2005. Strictly Lexical Dependency Parsing. In *Proceedings of the 9th International Workshop on Parsing Technologies*. pp. 152-159.

A.Wu. 2003. Learning Verb-Noun Relations to Improve Parsing. In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*. pp. 119-124.

N.Xue, F.Chiou and M.Palmer. 2002. Building a Large-Scale Annotated Chinese Corpus. In *Proceedings of the 18th International Conference on Computational Linguistics*.

N.Xue and M.Palmer. 2003. Annotating the Propositions in the Penn Chinese Treebank. In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*.

N.Xue and M.Palmer. 2005. Automatic Semantic Rule Labeling for Chinese Verbs. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*.

H.Yamada and Y.Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of the 7th International Workshop on Parsing Technologies*.

A.Yates, S.Schoenmackers, and O.Etzioni. 2006. Detecting Parser Errors Using Web-based Semantic Filters. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. pp. 27-34.

J.You and K.Chen. 2004. Automatic Semantic Role Assignment for a Tree Structure. In *Proceedings of the 3rd SIGHAN Workshop on Chinese Language Processing*.

K.Yu, S.Kurohashi, and H.Liu. 2007. A Three-step Deterministic Parser for Chinese Dependency Parsing. In *Proceedings of the Human Language Technologies: the Annual Conference of the North American Chapter of the Association for Computational Linguistics 2007*. pp. 201-204.