An Effective Framework for Chinese Syntactic Parsing

Xing Li, Chengqing Zong

Abstract—This paper presents an effective framework for Chinese syntactic parsing, which includes two parts. The first one is a parsing framework, which is based on an improved bottom-up chart parsing algorithm, and integrates the idea of the beam search strategy of N best algorithm and heuristic function of A* algorithm for pruning, then get multiple parsing trees. The second is a novel evaluation model, which integrates contextual and partial lexical information into traditional PCFG model and defines a new score function. Using this model, the tree with the highest score is found out as the best parsing tree. Finally, the contrasting experiment results are given.

Keywords—syntactic parsing, PCFG, pruning, evaluation model.

I. INTRODUCTION

Chinese syntactic parsing is one of the significant components in natural language processing. The simplest and safest way is to do exhaustive bottom-up parsing, and then selects the totally correct parsing tree from all the possible trees. However, because of the ambiguous characteristics of natural language, it is a prevalent phenomenon that one sentence has thousands of possible parsing trees. An exhaustive parsing is often impractical or impossible [1].

In recent years, with the development of large scale of Chinese tree-bank, the Probabilistic Context Free Grammar (PCFG) has become the mostly commonly used method to disambiguate trees. Researchers have presented many algorithms with PCFG to get the best parsing tree, such as best-first algorithm and A* search strategies [2], etc. Most of the algorithms aim to obtain the unique tree with the highest PCFG probability (Viterbi tree). However, as to Chinese, the Viterbi parsing tree is not totally correct in many situations.

There is an example in Fig. 1.

Above phenomenon is caused by the unreasonable assumption of PCFG. For Chinese, the two basic principles of PCFG – the independence assumption and context-freeness assumption – are questionable. So, the additional lexical and contextual information are necessary, which will be discussed in section two.

Above all, although the result of pure PCFG is not satisfying, yet it can be used to do initial disambiguation in parsing and get multiple trees. Then a context-sensitive model is used to reevaluate all the trees and choose the best one.

II. OUR APPROACH

A. The overall description

The original input for the system is the text that has been processed by the word segmentation and part-of-speech (POS) tagging system, such as $T = w_1 / t_1, w_2 / t_2, \ldots, w_n / t_n$. Through processing of parsing and disambiguation module, finally the best result tree is got, as shown in Fig. 1. This tree can best describe the syntactic structure of the original sentence.

The process of this system can be shown as Fig. 2:

The preprocessing module mainly takes use of the indication information of Chinese punctuations to decompose long compound sentences into several single sentences, and simplify the parsing of some sentences that include some special punctuation. For example, the “《” and “》” always indicate that
there must be a name of a book between them, so it can be parsed as a noun phrase (NP). Therefore, a syntax-pattern can be defined as follows:

\[
NP \rightarrow X X, X \text{ stands for any input text strings.}
\]

Similar laws are discovered, so a syntax-pattern set is constructed. This part is not the emphasis of our discussion in this paper, and we will give particular description in other paper. The later two modules will be explained detailedly in next two sections.

**B. The parsing and pruning module**

This module includes two parts: parsing algorithm and pruning strategy.

1) *The Improved Chart Parsing Algorithm*

The basic parsing algorithm in our system is an improved bottom-up chart-parsing algorithm, which is presented by Bai Shuo et al. [5]. The algorithm makes reference to GLR algorithm, by introducing a kind of effectual “look ahead” function at low cost to avoid creating lots of useless arcs. Therefore, an inverted role list and a starting rules list are constructed first based on the input PCFG rules. Please refer to [5]-[7] for details.

In their approach, edges that can’t lead to the root node S (successful parsing) are filtered out and the searching space and time can be reduced greatly. However, all rules are given the same priority, which can be modified in our approach with PCFG. In our approach, some modifications are made to the lists. The two lists are sorted according to the probabilities of corresponding rules, which ensure more possible parsing path with higher priority. With the proposed modifications, the searching can be heuristic in later parsing process.

2) *Pruning strategy*

Our approach refers to the beam search strategy of N best algorithm with some variations. N best algorithm keeps only a portion of the edges with higher figure-of-merits (FOMs) compared to other edges in the same cell. But our approach reserves all edges with different POS tags in the same cell. To edges with the same POS tags in the same cell (they have ambiguous sub-trees), no more than N best ambiguous sub-trees can be reserved and others are pruned. The heuristic function of A* algorithm and two thresholds are defined here.

In A* algorithm, the edges are ordered based on a heuristic function, which is the sum of their known internal costs of construction and a conservative estimate of their costs of completion [2]. Formally, for edge \( e \), \( cost(e) \) denotes the valuation of \( e \), \( \alpha(e) \) denotes the internal cost of construction and \( \beta(e) \) denotes the estimate of cost of completion.

\[
\text{cost}(e) = \alpha(e) + \beta(e)
\]

The grammar projection estimate is used because the context information is helpful to select the most possible parsing path before creating corresponding edges. Only the parent context information is considered here.

For edge \( e \) shown in Fig. 3, \( \alpha(e) \) and \( \beta(e) \) are defined as:

\[
\alpha(e) = \text{Insidecost}(X) = \max\{P(X_1), P(X_2), \ldots, P(X_j)\}
\]

\[
\beta(e) = \text{Outsidecost}(X) = \text{Insidecost}(Y) + \text{Insidecost}(Z) + \log P(A \rightarrow X Y Z)
\]

Where \( \alpha(e) \) is defined as the logarithm probability of the best inside parse of edge \( e \) (its Viterbi inside score), and \( \beta(e) \) is defined as the outside cost of edge \( e \) in given context.

For example, a simple PCFG \( G \) and an input sentence \( S \) are shown in Table 1.

**TABLE 1**

<table>
<thead>
<tr>
<th>Rules</th>
<th>Probability (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S \rightarrow d_1 )</td>
<td>0.467386</td>
</tr>
<tr>
<td>( S \rightarrow z_3 )</td>
<td>0.224409</td>
</tr>
<tr>
<td>( d_1 \rightarrow v \rightarrow a )</td>
<td>0.000561</td>
</tr>
<tr>
<td>( d_1 \rightarrow a \rightarrow a )</td>
<td>0.005911</td>
</tr>
<tr>
<td>( v \rightarrow v \rightarrow a )</td>
<td>0.055620</td>
</tr>
<tr>
<td>( a \rightarrow a \rightarrow v )</td>
<td>0.000884</td>
</tr>
<tr>
<td>( z_3 \rightarrow d_1 \rightarrow v )</td>
<td>0.342163</td>
</tr>
</tbody>
</table>

Fig. 4 shows a chart graph in the parsing process.

The graph shown in Fig. 4 can be transformed into the tree structure as Fig. 5:
similar. Given next edge \(a[2,3]\), \(\text{cost}(e_j)\) and \(\text{cost}(e_j)\) can be computed as follows:
\[
\alpha(e_j) = \text{InsideCost}(vp[0,2])
\]
\[
= \text{InsideCost}(v[0,1]) + \text{InsideCost}(n[1,2]) + \log P(vp > v n)
\]
\[
= \log(1) + \log(1) + \log(0.055620) = -1.254769
\]
\[
\beta(e_j) = \text{OutsideCost}(vp[0,2])
\]
\[
= \text{InsideCost}(a[2,3]) + \log P(dj > vp a)
\]
\[
= \log(1) + \log(0.004959) = -3.105395
\]
\[
\text{cost}(e_j) = \alpha(e_j) + \beta(e_j) = -4.561064
\]
\[
\text{cost}(e_j) = \alpha(e_j) + \beta(e_j) = -5.218428
\]

For \(\text{cost}(e_j) > \text{cost}(e_j),\) the branch node \(dj[0,3]\) has two ambiguous sub-trees, and the sub-tree including edge \(e_i\) is Viterbi sub-tree. The classical A* algorithm only reserves edge \(e_i\) in this situation. However, here the sub-tree including edge \(e_i,\) instead of \(e_i,\) is the correct sub-tree. Our approach reserves these two sub-trees.

When there are many ambiguous sub-trees, a proper threshold should be defined to combine with heuristic function to prune. The evaluations of all such edges are computed and sorted, then edge \(e_i\) can be cut if it meets condition: \(\text{cost}(\max\{e_1, e_2, ..., e_n\}) + \text{cost}(e_j) > \text{PROB},\) (It means that \(\text{cost}(\max\{e_1, e_2, ..., e_n\}) > \text{PROB} \times \text{cost}(e_j)\).)

\(\text{PROB}\) is defined as a dynamic threshold. It decreases by an invariable value with the increase of the level of the branch node in the tree. \(\text{PROB}\) of branch node \(X\) is defined as follows (the data in (4) are given experimentally):
\[
\text{PROB} = \begin{cases}
6.0 - \text{level}(X)/2, & \text{level}(X) <= 8 \\
2.0 & \text{level}(X) > 8
\end{cases}
\]

Another threshold \(N\) is defined to restrict the maximum number of ambiguous sub-trees for every branch node. Number \(N\) is defined as 7, which is given experimentally.

C. The evaluation module

Traditional PCFG is based on two basic assumptions:
1) Context-freeness assumption, which means that the score of a parsing tree is decided only by its sub-trees and the grammar rules used currently.
2) Independence assumption, which means that the sub-trees of one parsing tree are independent to each other and all the sub-trees are independent to the grammar rules.

Based on the two assumptions, the score function is defined as follows: For a tree \(T\) which is produced by such a grammar rule \(e: r : T \rightarrow u \_ r \_ u \_ u \_ u,\) the score function of \(T\) is:
\[
S(T) = P(r) \times \prod_{i=1}^{*} S(u_i)
\]

As we have said before, the above two assumptions are questionable. The analysis and corresponding resolution are discussed detailedly in next two sections.

1) Construction of Co-occurrence matrix

Assumption 1) is wrong because the context-sensitiveness of Chinese is obvious. There are some Chinese function words that have important indication to sentence structure.

For example, in Chinese, structural auxiliary “的” is the mark of attribute [8]. That’s to say that the probability that there is a noun phrase (NP) behind “的” is much higher than any other phrases such as verb phrase (VP) etc. Considering the phrase “我们的研究计划” (“研究计划” can be a NP or a VP, but because of the anterior “的”, obviously it is a NP. However, in PCFG, without considering the indication of “的”, the “研究计划” “will possibly be parsed as a VP, because the probability that “研究” as a verb is higher than that of a noun.

In order to solve this problem, a terminal (POS tag of word) co-occurrence matrix and a non-terminal (phrase or syntax tag) co-occurrence matrix are defined to integrate the context sensitive information into PCFG. In the tree structure, the terminals constitute leaf nodes (e.g. \(v\) and \(n\) in Fig. 5), the non-terminals constitute branch nodes (e.g. \(dj\) and \(vp\) in Fig. 5).

Formally, they are defined as follows:
The letter \(t\) denotes the terminal.
The letter \(P\) denotes the non-terminal.
The backward co-occurrence probability \(B(P,t)\) denotes the probability that there is a \(t\) immediately after \(P\).

The forward co-occurrence probability \(F(P,t)\) denotes the probability that there is a \(t\) immediately after \(P\). So,
\[
B(P,t) = P(t/P)/P, F(P,t) = P(P/t)/P
\]

(6)

Two special marks named \(b\) and \(e\) are added into the model, which denote the beginning and end marks of a sentence separately. So,
\[
B(P,b) = P(b/P)/P, F(P,e) = P(P/e)/P
\]

(7)

Obviously, to all \(P\), such unitary condition should be satisfied:
\[
\sum_{t} B(P,t') = \sum_{t} F(P,t') = 1
\]

(8)

Similarly, the definitions of terminal \(t\) are as follows:
\[
B(t,t') = P(t'/t)/t, F(t,t') = P(t'/t)
\]
\[
B(t,b) = P(b/t)/t, F(t,e) = P(e/t)
\]
\[
\sum_{t} B(t,t') = \sum_{t} F(t,t') = 1
\]

(9)

For all \(t\) and \(P\), their B and F values are computed, then the terminal and non-terminal co-occurrence matrices are constructed. When computing the scores of parsing trees, the two matrices are used in later formula to contribute to the evaluation of sentences. Thus, the context co-occurrence information is added into our model.

2) New evaluation formula

Assumption 2) is questionable because sub-trees can’t
always be independent to each other. So using formula (5) to
compute the score of trees is not reasonable. In addition, the
product of all the probabilities of sub-trees is lower than that of
any sub-tree. A fairly low probability sub-tree will greatly
reduce the product. In other words, the result only reflects the
low probability events. So, some modifications are made to the
score function of (5).

The most difference is that the evaluation score of the
parsing tree is the geometric mean of all probabilities but not the
product of them.

Formally, for the terminal node \( t \), the terminal before \( t \) is
\( t' \), the one after \( t \) is \( t'' \), score of \( t \) is defined as follows:

\[
S(t) = \sqrt{B(t,t') + F(t,t'')}
\]

(10)

For the non-terminal node \( P \), which has been deduced by
such a rule \( r: P \rightarrow u_1 u_2 \ldots u_n \), the terminal before \( P \) is \( t' \), the
one after \( P \) is \( t'' \). So, score of \( P \) is defined as follows:

\[
S(P) = \sqrt{P(r) + \sum_{i=1}^{n} S(u_i) + B(P,t') + F(P,t'')}
\]

(11)

III. EXPERIMENT RESULTS

Our experiments are performed based on TCT973 Chinese
tree-bank (http://www.chineseldc.org/index.html), which is
built from true Chinese text corpus after 1990th with manifold
styles.

Firstly, 8600 sentences were chosen randomly from the
tree-bank as train set, and other 500 sentences as test set. The
2706 PCFG rules used in our system are extracted from the
train set, and the corresponding probabilities are computed by
using Maximum Likelihood Estimate (MLE) method. The two
c0-occurrence matrixes are also constructed based on the train
set and computed by using MLE method. In addition, some
Chinese function words which have special functions are
tagged with special marks, such as structural auxiliary “的” ,
“地”，“得” and tense auxiliary “了” , “着” , “过” etc.

To test that our model has an advantage over PCFG, we do
two groups of contrasting experiment.

The first experiment parses the 500 sentences of test set. The
lengths of these sentences are between 5 and 40 words with the
average length 22.8 words. The standard PARSEVAL
measures [9] are used to evaluate the two models, the results are
shown in Table 2:

<table>
<thead>
<tr>
<th>Model</th>
<th>LR (%  )</th>
<th>LR (S)</th>
<th>CUBS</th>
<th>CRICS (S)</th>
<th>2CRICS (S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCFG</td>
<td>74.0</td>
<td>73.1</td>
<td>1.86</td>
<td>33.5</td>
<td>67.6</td>
</tr>
<tr>
<td>Our model</td>
<td>76.2</td>
<td>76.9</td>
<td>1.80</td>
<td>36.9</td>
<td>69.3</td>
</tr>
</tbody>
</table>

Above contrasting results show that our model is superior to
traditional PCFG model.

The second experiment selects all the sentences whose
lengths are between 5 and 20 from the 500 sentences of test set.
This is still an opening test. The results are shown in Table 3:

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of sentences</th>
<th>Including correct sentences</th>
<th>Inclusion ratio</th>
<th>Number of correct sentences</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCFG</td>
<td>152</td>
<td>167</td>
<td>87%</td>
<td>116</td>
<td>69.5%</td>
</tr>
<tr>
<td>Our model</td>
<td>133</td>
<td>147</td>
<td>84%</td>
<td>133</td>
<td>79.8%</td>
</tr>
</tbody>
</table>

In table 3, the “including correct sentences” are the numbers
of sentences which include the correct tree in the multiple result
trees, and the “inclusion ratio” is the percentage of them. The
percentages of 87 prove that our parsing algorithm including
pruning is effective. The accuracy shows the disambiguation
ability of the evaluation model. It can be seen that disambiguation ability of our model is superior to PCFG.

But the results in table 3 are based on the sentences with middling or short lengths. When dealing with long ones, the
results are not so good. Which may be caused by following reasons.
Firstly, by analyzing long sentences, especially those
compound ones, we found that even if most of the single
sentences in the compound sentences are correct, only one
wrong single ones can make the whole sentences wrong. So the
inclusion ratio is low. Besides, with the increase of sentences
length, there will be much more ambiguous result trees, and to
disambiguate is much more difficult. In a word, Chinese
syntactic parsing is still a difficult task.

IV. CONCLUSION

In this paper, we present an efficient framework for Chinese
syntactic parsing. Two main modules are included – a parsing
and pruning module and a disambiguation module. The latter is
a disambiguation model, which integrates contextual and
partial lexical information into traditional PCFG model.
Contrasting experiments show that the disambiguation ability is
improved.

REFERENCES

probabilistic HPSG parsing: integrating semantic and syntactic preference
to guide the parsing”. Proceedings of IJCNLP-04 (Companion Volume),


recognizer”, ACM Transactions on Programming Languages and Systems,

[9] E.Charniak, “context-free grammar and word statistics”. In Proc of