VIETNAMESE PARSING APPLYING THE PCFG MODEL

Hoang Anh Viet, Dinh Thi Phuong Thu, Huynh Quyet Thang
Hanoi University of Technology

Abstract— This paper presents a Vietnamese syntax parsing method by applying PCFG model and improved CYK algorithm. The PCFG model (Probabilistic Context–Free Grammar) has been widely applied for language parsing problems and given a high effect especially for English. In this paper, we propose a model that is applied the PCFG for Vietnamese syntax parsing and an approach for building a set of linguistic data that called Vietnamese Treebank, this data will be used for establishing PCFG rules. At the same time, we also propose some improvements for building set of Vietnamese syntax rules to enhance the accuracy of parsing. The result of our experiment has been compared and demonstrated the effect of applying the PCFG model for Vietnamese syntax parsing.

Index Terms— the Vietnamese parsing, the CYK algorithm, Probabilistic Context–Free Grammar, Vietnamese grammatical rules.

I. INTRODUCTION

We always have many problems in natural language processing especially for syntax parsing, for examples: the ambiguity of lexicon, the ambiguity of Vietnamese syntax and the lack of necessary Vietnamese linguistic data for Vietnamese text processing.

The ambiguity of lexicon: This is the first problem, or the first phase that need to be performed, that is analysis of the form of word. Vietnamese is monosyllabic language. It is so different from polysyllabic languages that are clearly analyzed list of words that make the input sentence after parsing lexical phase. With polysyllabic languages, every word is parted by space character “…”. But in Vietnamese, the unit between two space characters is syllable, not word. One word can be made by one or many syllables. So, to identify word in Vietnamese, we have to make a combination of syllables, and then compare this combination with an available list of Vietnamese words. We can conclude the combination is right word if this combination exists in Vietnamese vocabulary. In a sentence, we usually have many combinations of close syllables and many right meaning of word. This is unclear input for syntax parsing phase, list of words in the input sentence is not clear. To resolve this problem, we used the result of another research about Vietnamese word segmentation, the accuracy about 94%-95% [12].

The ambiguity of Vietnamese syntax: Syntax parsing is designed to find out the relationship between every elements of the input sentence, and then understand the meaning of whole sentence [1]. It’s very easy to understand the meaning of sentence in case of having only one output parsing. But, in general we have many results for each Vietnamese sentence [1,2,3]. With ambiguous sentences we have more difficult for parsing phase. In fact, we have more ambiguities in spoken language, so it’s very difficult to build a system that can understand meaning of whole sentence in every language forms. So, in this paper, we only make an examination of standard sentence.

The lack of necessary Vietnamese linguistic data: In general, Vietnamese text processing has not been deeply researched, there’re not many results for Vietnamese. One of the biggest problems for Vietnamese is the missing test corpus. Meanwhile, we have Pen Treebank II, that is an English corpus including about 50.000 syntax trees parsing [19], and some other Treebank of some languages. These linguistic data corpora are necessary knowledge resources for building high accuracy model, we can apply statistics models, probability or use extraction technique, comparison, conversion… based on these corpora. But in fact, we don’t have any such knowledge resources for Vietnamese, so we have extremely difficulty in natural language processing especially Vietnamese syntax parsing.

In this paper, we present our researches that take aim at improving the performance of Vietnamese syntax parsing. This paper is organized as follow: section 2 presents the construction of Vietnamese syntax rules task, this set of rules ensures the closeness, grasp the whole of almost of Vietnamese sentences, and easily enlarge. Section 3 presents the ambiguity solution model by using the PCFG model. Finally, in section 4, we present experiment results of using PCFG probabilistic model for Vietnamese syntax parsing, including the comparison of effect between our proposed model and existing Vietnamese syntax parsing models.

II. COMPLETING THE SET OF VIETNAMESE SYNTAX RULES

A. Approach for building the rules

In [5], an approach for constructing the Vietnamese syntax rules set (VSRS) applying for context-free grammar is
presented. The VSRS is built up with rules for Part-of-Speech (POS), rules for phrases, rules for composition of sentence, and rules for sentences. The grammatical opinion used to build this set of rules is based on Nguyễn Tài Cán [1], Nguyễn Chí Hòa [2] and Diệp Quang Ban [3].

The rules for Part-Of-Speech: According to [5], we have reused these kinds of rules consisting of 7 rules for noun, 23 rules for verb, 2 rules for adjective, 5 rules for adjunct, 2 rules for linking verb, and 5 rules for pronoun…

The rules for phrase: According to [1,2,3], there are three phrases: noun phrase, verb phrase and adjective phrase. With regard to structure, these phrases have three parts: pre-affix, post-affix and main word. However, according to [2], each phrase has different affix parts depend on the main word. The building processes for noun phrase, verb phrase and adjective phrase are fairly similar, so we will show the details of the building process for rules of noun phrases and the improvement in comparison with [5].

B. Building the rules for Noun phrase

According to [2], the common structure of noun phrase is:

General Affix/ Quantity Affix/ Classifier concrete Affix/Main word/ Definite Affix/ Demonstrate Affix

The Affixes depend on the Part- Of – Speech of the Main word.

According to [2], there are four types of noun could be the main word of noun phrase. Adding to set of symbols of grammar four symbols, they are: <Abstract noun phrase>, <locative noun phrase>, <countable noun phrase>, <collective noun phrase>, thus, having four rules:

1. < Main word Noun phrase > < Abstract noun phrase >
2. < Main word Noun phrase > < locative noun phrase >
3. < Main word Noun phrase > < countable noun phrase >
4. < Main word Noun phrase > < collective noun phrase >

With collective noun, the pre-affix only be collective-affix[2], thus we have one more rule:

5. < Pre-affix collective noun > < collective-affix >

The post-affix of noun phrase can be the definite-affix or demonstrative-affix, thus we have more rules[2,3]:

6. < Post-affix noun phrase > < definite-affix >
7. < Post-affix noun phrase > < demonstrative-affix >
8. < Post-affix noun phrase > < demonstrative-affix > < definite-affix >
9. < Post-affix noun phrase > < definite-affix > < demonstrative-affix >

Because the collective-affix is composed by the collective-noun, thus we have one more rule:

10. < collective-affix > < collective-noun >

Because the definite-affix can be composed by adjective, verb, noun or one phrase so we have more rules:

11. < definite-affix > < adjective >
12. < definite-affix > < verb >
13. < definite-affix > < noun >
14. < definite-affix > < Verb phrase >
15. < definite-affix > < adjective phrase >
16. < definite-affix > < noun phrase >

17. < definite-affix > < definite-affix > < demonstrative-affix >
We have a new type of rule here, called “repetitive rule”:

18. < definite-affix > < definite-affix > < definite-affix >

This kind of rule will widen the ability of the grammal to cover the whole of Vietnamese sentences, this not only reduce the number of identical rules, but also increase the accuracy of the rules.

Because the demonstrative-affix is composed by demonstrative pronoun so one more rule will be added:

19. < demonstrative-affix > < demonstrative pronoun >

After building rules for ingredients of noun phrase, we build rules for noun phrase with the main word is collective-noun:

20. < Noun phrase > < Pre-affix collective noun > < Main word collective-noun > < Post-affix noun phrase >
21. < Noun phrase > < Pre-affix collective noun > < Main word collective-noun > < linking verb > < Post-affix noun phrase >

Like this, we have accomplished the set of rules for collective noun phrase. Other noun phrases such as verb phrases, adjective phrases are built up with the same way as well. The building process ensures the rules are in free context grammar form \( r = A \rightarrow \emptyset \).

C. The improvement of rules

With noun phrase, we have the same post-affix for all 4 types of the main word: Abstract noun, locative noun, countable noun, and collective noun. So we don’t have to build post-affix for each individual noun phrases like [5], just construct pre-affix which is related to the main word. This will reduce the redundant and repetitive rules.

To go into details of comparison in [5], we present the structure of a phrase as below:

Pre-affix X/ Main word X/ Post-affix X

With noun phrase, we have X is one of 4 types of noun mentioned above. Normally, we have to build a set of rules for Pre-affix and a set of rules for Post-affix that relate to each individual X. But in our approach, the set of rules for Post-affix is the same for 4 types of X. So, we have only one set of rules for Post-affix of noun phrase instead of 4 ones in [5], thus there are so many redundant rules in [5]. Especially with verb phrase, [5] had built 297 rules for this phrase while we have just only 26 rules.

The rules for ingredients and sentences are reused in [5]. Finally, we have 181 Vietnamese syntax rules. The

<table>
<thead>
<tr>
<th>Set of Rules</th>
<th>Quantity</th>
<th>Outspread</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our rules</td>
<td>181</td>
<td>Simple sentence, exclamative sentence, complicate sentence, question sentence…</td>
</tr>
</tbody>
</table>

comparison about the performance between our rules and ones in [5] is described in table 1.
III. VIETNAMESE SYNTAX PARSING WITH PCFG MODEL

According to [7,8,9,10,11], syntax parsing algorithms that are usually used for Context free grammar are Cocke-Younger-Kasami (CYK) algorithm and Earley algorithm. And as the research and comparison results of [5], the CYK algorithm is more effective for Vietnamese, so we continue using and improving this algorithm for Vietnamese syntax parsing. The CYK algorithm is a bottom up algorithm and is important theoretically, used standard rules in Chomsky format. There is a problem need to be solved, that is how to choose correctly the output tree. With each input sentence, normally having many output trees. For instance, with a sentence “Tôi đi học”/”I go to school”, there are two output tree t1, t2 as below:

\[ P(w_{lm}) = \sum_{t} P(w_{lm}, t) \]

Moreover, it is easy to find the probability of a tree in a PCFG model. One just multiplies the probabilities of the rules that built its local subtrees.

To apply PCFG model to resolve the ambiguous problem, we have to build a linguistic data store that training for PCFG. In reality, English has a corpus that consists of thousands of parsed sentences and stored in standard format [8,9]. Thus to build the PCFG for English become very easy. Meanwhile, we still don’t have the similar corpus for Vietnamese, so the first step we need to do is that building a linguistic data for Vietnamese that consists correctly parsed sentences. We propose an approach for solving this linguistic data by building Vietnamese Treebank- A corpus that consists correctly parsed sentences.

B. PCFG applying model for Vietnamese

In our research, we use PCFG model as ambiguity solution phase to choose the correct output tree. The result of this work is only one parsing tree with the highest probabilistic for each input sentence is given.

As below, we propose two models for building and using PCFG model.

1) Building Treebank Model

As mention above, we lack one linguistic data corpus for training process to make PCFG. So, in order to apply PCFG model we have to do one more task that is build up Treebank for Vietnamese – A corpus that consists parsed sentences.
Fig 2. Building Treebank model based on basic syntax parsing Model
In our model, we consider that the accuracy of the syntax parsing phase has achieved acceptable measure (without resolving the ambiguity). The main idea of this model can be interpreted as three steps below:

**Step 1:** Choose a set of standard Vietnamese sentences.

**Step 2:** Use a Vietnamese syntax parsing tool to parse the set in the step 1, this tool is call a basic parsing model because with an input sentence will have many output tree. Normally, this basic model use CYK algorithm without resolving the ambiguity.

**Step 3:** With the set of output trees, we choose the most suitable tree to the input sentence, then save it into Treebank with xml format. This choice based on knowledge of Vietnamese grammar.

According to this model, the quality of the Treebank will depend on the choice the standard sentence corpus and the selection the correct output tree. So, it’s necessary to have linguists joined into this phase. In restriction of our research, we have selected a set of sample sentences in some Vietnamese materials [1,2,3] and our TreeBank will be built up from this collection.

2) **PCFG syntax parsing Model**

This model will be applied after having Vietnamese Treebank Corpus (VTC). At this time, the VTC doesn’t need to have too many samples, just be enough for estimating PCFG parameters.

![Diagram of Syntax parsing model with PCFG](image)

**Fig 3. Syntax parsing model with PCFG**

This model has three steps as description below:

**Step 1:** Estimating probabilistic parameters for PCFG from VTC, this step just be done once time only. The algorithm for estimating probabilistic is carried out as below, however we will add 1 to each probabilistic value. It means each of grammatical rules has a PCFG parameter that has value from 1 to 2.

Example: the rule “*câu → chủ ngữ + ví ngữ*” (sentence → subject + predicative) has a probabilistic that estimated from VTC is 0.862, but its value of PCFG parameter is 1.862.

**Step 2:** With a set of PCFG rules (each rule has a PCFG parameter) and a sentence needs to be parsed, the PCFG parsing model will has only one parsing tree output. We have two strategies for using PCFG to resolve the ambiguity, that is using PCFG and CYK directly or using PCFG parameters at the end of parsing processing to find out the correct result.

**Step 3:** With the correct tree output, user may save it to enlarge the VTC.

There is description for “Estimating probabilistic parameters for PCFG” function:

**Input:** The VTC.

**Output:** A set of PCFG rules that has one parameter for each rule.

Details for this function can be described below:

**Step 1:** For each tree in the Treebank, convert it to a suitable structure to easy count the time of using of each rule.

**Step 2:** Carry out couting how many times one rule is used in the VTC, keep the rule and its number of appearance times.

**Step 3:** After examining all trees of the VTC in step 2, we have a set of rules that has one number of appearance times for each rule. Carry out estimating probability for each rule follow formula below:

\[
\hat{P}(N^i \rightarrow \zeta) = \frac{C(N^i \rightarrow \zeta)}{\sum_{\gamma} C(N^i \rightarrow \gamma)}
\]

In there: \( C(*) \) is the number of appearance times of (*) rule in the VTC.

**Step 4:** For each rule, add 1 to its probability to create PCFG parameters. Save all rules and their PCFG parameter into set of PCFG rules.

3) **The Syntax Parsing Algorithm with PCFG**

This Syntax parsing algorithm is combined extended CYK algorithm [5,6] and the PCFG parameters.

**Input:** A group of words (List of words in the input sentence).

**Output:** None or only one output parsed tree.

**Step 1:** Building up the 2 dimensions matrix \( M[n+1,n+1] \) to store location of groups of words in list of group words of the sentence. \( M[i,j] \) has a value which is location of group words starts from \( i \) and ends at \( j \). \( N \) is the number of words appearing in the input sentence.

Initiate \( M \), set all elements to -1. Determine all of able Pos-tag-speech of each words in the sentence, put this element that spans from \( i \) to \( j \) in the sentence on \( M[i,j] \). After this step, all initial values will lie on the main diagonal of the matrix.

**Step 2:** On each labels \( L \) of list meaning of each current element, carry out finding all rules \( A \rightarrow L \). If \( A \) exists, then compare the PCFG parameter of two rules have the same left side \( A \): the rule with the higher value will be stored. If \( A \) does not exist then add \( A \) to the list, and attach a parameter to \( A \) which is counted by multipling the parameter of \( A \rightarrow L \) and the parameter of \( L \). Repeat step 2 until none label is found out.

**Step 3:** Determine gradually elements of diagonals from the main diagonal forward. The \( M[k,1] \) is composed by \( M[k,1+i] \) and \( M[k+i+1,1] \) with \( i \) changes in range \([0,1-k]\).

With each label \( M1 \) belongs to element \( M[k,k+i] \) and \( M2 \) belongs to \( M[k+i+1,1] \), if exist rules likes \( X \rightarrow M1M2[TY] \) then create new label \( X \) and \( [TY] \) attached to \( X \) as comment part. With this \( X \) label, the probability parameter is similarly...
counted by multiplying parameters of M1 and M2, then compare with the existing X.
With each label M1 belongs to list of labels of M[k,l+i] and having comment part not null (e.g OP), if we find out a rule like M1O, we will create the second label M1 as a copy of the first, however the comment part of the second = comment \ \{O\} = P and the probability parameter of the second label M1 will be multiplied with parameter of O label.
Repeat this step for each M[k,l] elements.

**Step 4:** If the value M[0,n+1] >0 then the input sentence has been parsed. Get out the elements that have value of left side is “câu” /”sentence” in the list of the meaning of the M[0,n+1]. Build up the parsed tree of the input sentence rely on traces stored in the list of meaning.

### IV. THE RESULTS OF VIETNAMESE SYNTAX PARSING APPLIED PCFG

**A. Building the training corpus and experimenting**

We have collected and selected a set of standard Vietnamese sentences from many materials: [1], [2], [3], Internet… This corpus has over 2000 sentences include various forms: simple sentence, special simple sentence, exclamative sentence, compound sentence, …

From this corpus, we have built up a Treebank that has about 500 parsed trees to use for vietnamese syntax parsing. On the other hand, we have had a test linguistic data that has over 1500 sentences to compare and evaluate the performance between our model and some published results.

This below table shows the list of materials and the number of sentences that we have had:

<table>
<thead>
<tr>
<th>Material</th>
<th>Number of sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vietnamese Grammar – Diệp Quang Ban</td>
<td>200</td>
</tr>
<tr>
<td>Vietnamese Grammar – Education Publisher</td>
<td>300</td>
</tr>
<tr>
<td>Communist party’s website</td>
<td>250</td>
</tr>
<tr>
<td>From internet: <a href="http://vnn.vn">http://vnn.vn</a></td>
<td>500</td>
</tr>
<tr>
<td>6 annotated corpora[16]</td>
<td>1000</td>
</tr>
<tr>
<td>Sum</td>
<td>2250</td>
</tr>
</tbody>
</table>

**Table 2. The standard sample corpus**

**B. Comparing the performance of parsing models**

1) The rate of parsing

Figure 4 shows the comparison result for parsing speed between three parsing phases according to the number of words in input sentence. In there, the basic phase is syntax parsing phase that parsing without resolving the ambiguity, this phase has fairly great results and long parsing time. The POS Tagger phase is syntax parsing phase that resolve ambiguity by using POS Tagger approach, and the PCFG phase is our model that using PCFG model to resolve ambiguity. All of three phases are set up and independently tested on the same computer to have exact result of the comparison.

As diagram below, we can see that the number of words of input sentence, in other words the length of the input sentence does not affect too much to parsing time of PCFG parsing phase. And the basic parsing phase and POS Tagger parsing phase can not carry out with long sentences. This is a strength demonstrates the effect of using PCFG as input process for other big natural language processing systems, because it does not harm to the time process of all system.

**Fig 4. Time comparison diagram**

<table>
<thead>
<tr>
<th>Criteria of comparison</th>
<th>Syntax Parsing Phases</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basic</td>
</tr>
<tr>
<td>Accuracy</td>
<td>43%</td>
</tr>
<tr>
<td>Average of Speed</td>
<td>2.356s</td>
</tr>
<tr>
<td>Average of ambiguity output trees</td>
<td>500 trees</td>
</tr>
</tbody>
</table>

**Table 3: The Comparison of performance of parsing models**

Table 3 shows that the accuracy of syntax parsing model applied PCFG has been raised, essentially the parsing speed has been greatly improved and the output has no ambiguity. Thus we can conclude that the ambiguity solution model that we propose is more quite effective than other currently available models for Vietnamese. The result of Vietnamese syntax parsing has been greatly improved.

**C. Result of PCFG rules**

With a sample corpus that has about 2250 sentences, we has built up our Treebank from 500 sentences. The result of building Treebank process is shown below:

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual parsed sentences</td>
<td>500 (tree)</td>
</tr>
<tr>
<td>Number of used rules</td>
<td>3900 (rule)</td>
</tr>
<tr>
<td>Number of types of rules</td>
<td>121/181</td>
</tr>
</tbody>
</table>

**Table 4. Information about PCFG rules.**

Since almost sentences that used to build Treebank are common sentences so the serial number of PCFG rules is not enough. However, our model still works well and the PCFG rules will be improved by expanding the size of Treebank.

**D. Evaluating the performance of parsing**

To evaluate the performance of syntax parsing, we use recall, precision and F parameters. In there

\[
\text{recall} = \frac{\text{Number of correctly parsed sentences (D)}}{\text{Total number of manually parsed sentences}}
\]
Threshold of sentences (D)

\[ \text{Recall} = \frac{\text{Number of correct sentences (B)}}{\text{Number of parsed sentences (C)}} \]

Precision = \frac{\text{Number of correctly parsed sentences (D)}}{\text{Number of parsed sentences (C)}}

\[ F = \frac{1}{ \frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}} \]

In our experiment, A = 1000, B = 800, C = 785 and D = 670. Thus, our parsing syntax model gives:

Table 5. The performance of the parsing model.

<table>
<thead>
<tr>
<th>Vietnamese Syntax Parsing Model applied PCFG.</th>
<th>Recall</th>
<th>Precision</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>85.35%</td>
<td>83.75%</td>
<td>84.54</td>
</tr>
</tbody>
</table>

V. CONCLUSION

In this paper, we have proposed our experiments on applying the PCFG model in Vietnamese syntax parsing. We have built a set of rules for Vietnamese consisting 181 rules and including almost common Vietnamese sentences. We have built a corpus that comprises 2250 standard Vietnamese sentences as well. And we have constructed a linguistic data-Treebank consisting about 500 parsed Vietnamese sentences with XML format to enhance the standard and wide community. Rely on this results, we have improved the Vietnamese syntax parsing better, especially the ambiguity has been resolved well by applying the PCFG model and become a good ambiguity resolving model. The results of experiment has represented that the Vietnamese syntax parsing applied the PCFG model is an effective approach, richly deserves to be taken interest in researching at a higher level.

In the future work, there are many matters we can perform to improve the quality of the Vietnamese syntax parsing, such as: improving the syntax rules with the help of linguist, completing the standard Vietnamese sentences with high quantity to enlarge the Vietnamese Treebank, researching other probability models such as: HPCFG, HMM...

REFERENCES