Development of a Partially Bracketed Corpus with Part-of-Speech Information Only

Hsin-Hsi Chen and **Yue-Shi Lee**

Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
E-mail: hh_chen@csie.ntu.edu.tw

Abstract

Research based on a treebank is active for many natural language applications. However, the work to build a large scale treebank is laborious and tedious. This paper proposes a probabilistic chunker to help the development of a partially bracketed corpus. The chunker partitions the part-of-speech sequence into segments called chunks. Rather than using a treebank as our training corpus, a corpus which is tagged with part-of-speech information only is used. The experimental results show the probabilistic chunker has more than 92% correct rate in outside test. The well-formed partially bracketed corpus is a milestone in the development of a treebank. Besides, the simple but effective chunker can also be applied to many natural language applications.

1. Introduction

Research based on a treebank, i.e., a corpus annotated with syntactic structures, is active for many natural language applications [1-5]. Framis [1] proposes a methodology to extract selectional restrictions at a variable level of abstraction from the Penn Treebank. Chen and Chen [2] propose a probabilistic chunker to decide the implicit boundaries of constituents and utilize the linguistic knowledge to extract the noun phrases by a finite state mechanism. In their study, Susanne Corpus is used as a training corpus for their chunker. Pocock and Atwell [3] investigate statistical grammars extracted from Spoken English Corpus (SEC), and apply these grammars to find the grammatically optimal path through a word lattice. The stochastic parsers are also developed in [4,5]. All these applications employ the syntactic information extracted from different treebanks and show the satisfactory results.

However, the work to build a large scale treebank is laborious and tedious. Very few large-scale treebanks are currently available especially for languages other than English. In this paper, we propose a probabilistic chunker to help the development of a partially bracketed corpus, i.e., a simpler version of a treebank. The chunker partitions the part-of-speech sequence into segments called *chunks*. Rather than using a treebank as our training corpus, a corpus which is tagged with part-of-speech information only is used. In the following sections we first introduce the experimental framework of our model. Lancaster-Oslo/Bergen (LOB) Corpus and Susanne Corpus are adopted. Then a tag mapper and a probabilistic chunker are described. Before concluding the experimental results are demonstrated.

2. Experimental Framework

Because the probabilistic chunker proposed in this paper is based on syntactic tags (parts of speech), a part-of-speech tagger is needed. A word sequence W is input to the part-of-speech tagger and a part-of-speech sequence P is generated. The output of the tagger is the input of the chunker. The probabilistic chunker partitions P into C, i.e., a sequence of chunks. Each chunk contains one or more parts of speech.

Consider the example "Attorneys for the mayor said that an amicable property settlement has been agreed upon .". This 15-word sentence is input to the part-of-speech tagger and a part-of-speech sequence "NNS IN ATI NPT VBD CS AT JJ NN NN HVZ BEN VBN IN ." is generated. The probabilistic chunker then partitions this sequence into several chunks. The chunked result is shown as follows.

[NNS] [IN ATI NPT] [VBD] [CS] [AT] [JJ NN NN] [HVZ BEN] [VBN] [IN] [.]

However, the performance evaluation of the chunker is a sticky work. To evaluate the performance of the chunker, Susanne Corpus, which is a modified and condensed version of Brown Corpus, is adopted. But, the tagging sets [6,7] of LOB Corpus and Susanne Corpus are different. The latter has finer tags than the former. Thus, a tag mapper is introduced in the experimental framework shown as Figure 1.

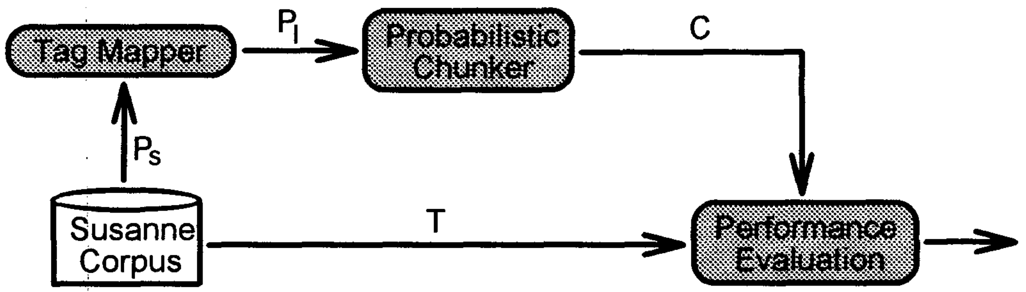

Figure 1. Experimental Framework

In our experiments, the test sentence P_S comes from Susanne Corpus. It is a part-of-speech sequence. The corresponding syntactic structure T is regarded as an evaluation criterion for the probabilistic chunker. It is sent to the performance evaluation model. The tag mapper in this figure is used to transform the Susanne part-of-speech into LOB part-of-speech. Through the tag mapper, P_S is converted into P_l. Then, P_l is input to the probabilistic chunker and a chunk sequence C is produced. Finally, the performance evaluation model reports the evaluation results according to C and T.

3. A Tag Mapper

The tagging set of Susanne Corpus is extended and modified from LOB Corpus. They have 424 and 153 tags, respectively. To map a Susanne tag into a LOB tag manually is a tedious work. Thus, an automatic tag mapping algorithm is provided. By our investigation, we found that words are good clues to relate these two tagging sets. Therefore, the first step in automatic tag mapping is to collect words from Susanne Corpus for each Susanne tag. Table 1 lists some examples.

Table 1. Words Extracted from Susanne Corpus for Some Susanne Tags

Susanne Tags	Words	LOB Tags
CC	and plus & And ond	CC
IW	with WITHOUT without WITH With	IN
NN1ux	physics math politics mathematics Athletics	NN
NNJ2	associates	NNS
VBM	<apos>m am ai	BEM
YTL	<bbold> <bital>	<No Match>

Column three in Table 1 denotes the correct mapping to LOB tags. The second step is to find the corresponding LOB tags from LOB Corpus for each word collected at the first step. Table 2 shows the sample results.

Table 2. LOB Tags Extracted from LOB Corpus for Each Word in Table 1

Susanne Tags	Words (LOB Tags)
CC	and (CC RB" RB NC) plus (IN JJ NN &FW) & (CC)
IW	with (IN IN" RI NC) without (IN RI)
NN1ux	physics (NN) politics (NN NNS) mathematics (NN)
NNJ2	associates (NNS VBZ)
VBM	am (BEM &FW) ai (HVZ BEZ BER)
YTL	*

Those words which cannot be found in LOB Corpus are removed. Symbol * denotes that all the words cannot be found in LOB Corpus. The third step is to find the corresponding LOB tag for each Susanne tag. For each Susanne tag, the frequency of LOB tags is calculated and the most frequent LOB tag is regarded as the result. For example, LOB tags NN and NNS in row three of Table 2 appear three and one times, respectively. Thus, Susanne tag NN1ux is mapped to LOB tag NN. After examining all the Susanne tags by these three steps, three cases have to be considered:

(1) Unique Tag. Only one LOB tag remains.

(2) Multiple Tags. More than one LOB tags remain.

(3) No Match.

When all the words extracted from Susanne Corpus for a Susanne tag cannot be found in LOB Corpus, the Susanne tag is mapped to "No Match". Some of these words are characteristic words such as YTL[1].

The experimental results are shown in Table 3.

Table 3. Experimental Results for Tag Mapping

Mapping Types	Subtypes	Number of Mapping
Unique Tag	Correct	151
	Wrong	7
Multiple Tags	Include	113
	Exclude	3
No Match	Correct	10
	Wrong	26

In Table 3, "Include" denotes that the correct tag belongs to the remaining multiple tags and " Exclude" denotes that the correct tag is not included in the remaining tags. Note that the *ditto* tags are not considered in this experiment. This is because the mapping for ditto tags can be obtained by human easily. Therefore, only 310 Susanne tags are resolved in this experiment. The experimental results show that the number of multiple tags is large. Thus, two heuristic rules are introduced to reduce the number of multiple tags.

[1]Tag YTL means "begin italics/boldface".

First, those LOB tags which are similar to Susanne tag are selected. For example, Susanne tag NNJ2 can be mapped to LOB tags NNS or VBZ in the above experiment. NNS has two common characters with NNJ2, so that Susanne tag NNJ2 is mapped to LOB tag NNS. Under this heuristic rule, the experimental results are shown in Table 4.

Table 4. Experimental Results After Applying the First Heuristic Rule

Mapping Types	Subtypes	Number of Mapping
Unique Tag	Correct	222
	Wrong	22
Multiple Tags	Include	28
	Exclude	2
No Match	Correct	10
	Wrong	26

Next, let us consider an example. Susanne tag IW can be mapped to LOB tags IN or RI in the above experiment. Thus, the first heuristic rule has no effects. We examine the tag mapping for the preceding and subsequent three tags of IW. They are listed as follows.

(-1)	Susanne Tag IIt	is mapped to	LOB Tag IN.
(-2)	Susanne Tag IIx	is mapped to	LOB Tag IN.
(-3)	Susanne Tag IO	is mapped to	LOB Tag IN.
(**)	Susanne Tag IW	is mapped to	LOB Tag IN RI.
(+2)	Susanne Tag JB	is mapped to	LOB Tag JJ.
(+3)	Susanne Tag JBo	is mapped to	LOB Tag AP.

Note that only tags which have the same first character as IW are considered, that is, only (-1), (-2) and (-3) are considered. In these three mappings, LOB tag IN is the most frequent and the only one mapping, and IN is a candidate for IW. Thus, Susanne tag IW is mapped to LOB tag IN. The above procedure forms the second heuristic rule. The experimental results after applying two heuristic rules are shown as follows.

Table 5. Experimental Results After Applying Two Heuristic Rules

Mapping Types	Subtypes	Number of Mapping
Unique Tag	Correct	232
	Wrong	22
Multiple Tags	Include	18
	Exclude	2
No Match	Correct	10
	Wrong	26

Three tags - say, FA, FB and GG, must be treated in particular. For example, Susanne Corpus tags genitive case noun as [John_NP 's_GG], but LOB Corpus tags it as [John's_PN$]. Two Susanne tags may be mapped into one LOB tag. Ignoring these three special tags, only nineteen Susanne tags have wrong mapping in Unique-Tag case.

4. A Probabilistic Chunker

Gale and Church [8] propose ϕ^2, a χ^2-like statistic, to measure the association between two words. Table 6 illustrates a two-by-two contingency table for words w_1 and w_2.

Table 6. A Contingency Table

	Word w_1	
Word w_2	a	b
	c	d

Cell a counts the number of sentences that contain both w_1 and w_2. Cell b (c) counts the number of sentences that contain w_2 (w_1) but not w_1 (w_2). Cell d counts the number of sentences that does not contain both w_1 and w_2. That is, if N is the total number of sentences, d=N-a-b-c. Based on this contingency table, ϕ^2 is defined as follows:

$$\phi^2 = \frac{(a*d+b*c)^2}{(a+b)*(a+c)*(b+d)*(c+d)}$$

ϕ^2 is bounded between 0 and 1. For different applications, there are different definitions for the contingency table. Instead of using the above definition, a modified version is shown as follows.

Definition 1: (For Two Parts of Speech)

$a=F(p_1,p_2)$

$b=F(p_2)-F(p_1,p_2)$

$c=F(p_1)-F(p_1,p_2)$

$d=N-a-b-c$

where p_i denotes part-of-speech i,

$F(p_1,p_2)$ is the frequency of which p_2 follows p_1,

$F(p_1)$ and $F(p_2)$ are the frequencies of p_1 and p_2, and

N is the corpus size in terms of the number of words in training corpus.

Based on this definition and ϕ^2 measure, consider the sentence "The Fulton County Grand Jury said Friday an investigation ...", which has tag sequence "ATI NP NPL JJ NN VBD NR AT NN ...". Its syntactic structure for the first seven words is shown in Figure 2.

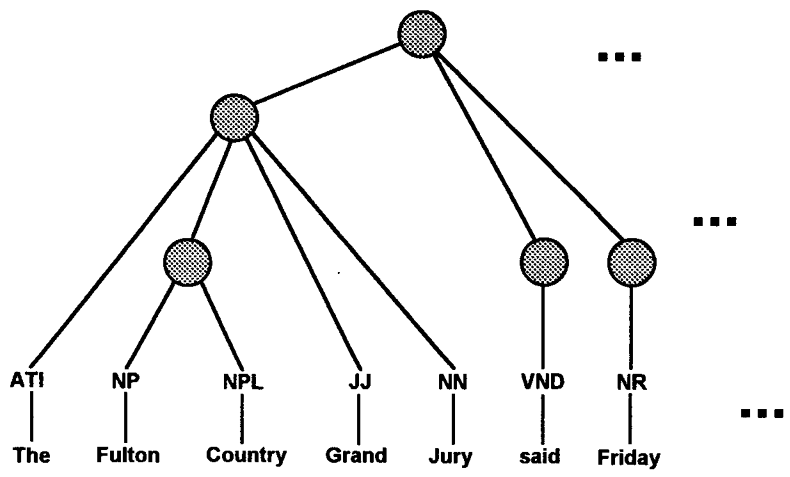

Figure 2. The Syntactic Structure for the First Seven Words

The ϕ^2 distribution for these parts of speech is shown in Figure 3. Position i (x axis) is the location between parts of speech p_i and p_{i+1}.

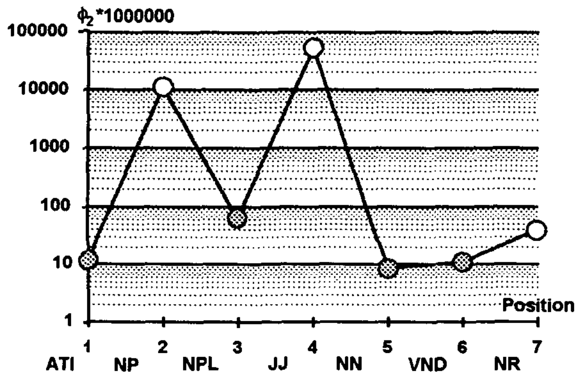

Figure 3. The ϕ^2 Distribution for the First Seven Words

Figure 3 shows that there are four local minimal positions, i.e., positions 1, 3, 5 and 6. They can be regarded as the boundaries of chunks. That is, ATI and NP belong to different chunks. Similarly, (NPL and JJ), (NN and VND) and (VND and NR) have the same situation. Let us discuss these concepts formally. For a probabilistic chunker, the generalized contingency table is defined as follows.

Definition 2: (For Two Chunks)

$a=F(c_1,c_2)$

$b=F(c_2)-F(c_1,c_2)$

$c=F(c_1)-F(c_1,c_2)$

$d=N-a-b-c$

where c_i denotes chunk i,

$F(c_1,c_2)$ is the frequency of which c_2 follows c_1,

$F(c_1)$ and $F(c_2)$ are the frequencies of c_1 and c_2, and

N is the corpus size in terms of the number of words in training corpus.

Let the tag sequence P be $p_1, p_2, \ldots, p_n$. Assume there are two possible chunked results. The first is composed of two chunks, i.e., $[p_1, p_2, \ldots, p_i]$ and $[p_{i+1}, p_{i+2}, \ldots, p_n]$, and is regarded as a correct result. The second is also composed of two chunks, i.e., $[p_1, p_2, \ldots, p_{i-1}]$ and $[p_i, p_{i+1}, \ldots, p_n]$, but is regarded as a wrong result. Since $[p_1, p_2, \ldots, p_i]$ is a chunk, $[p_1, p_2, \ldots, p_{i-1}]$ is very likely to be followed by p_i. In other words,

$$F([p_1, p_2, \ldots, p_{i-1}]) \approx F([p_1, p_2, \ldots, p_i]) \quad \text{.......}(1)$$

Similarly,

$$F([p_{i+1}, p_{i+2}, \ldots, p_n]) \approx F([p_{i+2}, p_{i+3}, \ldots, p_n])$$

Because p_i and p_{i+1} are in two different chunks,

$$F([p_i, p_{i+1}, \ldots, p_n]) << F([p_{i+1}, p_{i+2}, \ldots, p_n]) \quad \text{.......}(2)$$

Similarly,

$$F([p_1, p_2, \ldots, p_{i+1}]) << F([p_1, p_2, \ldots, p_i])$$

For the first chunked result, we can obtain the following contingency table:

$a^{\#} = F([p_1, p_2, ..., p_i],[p_{i+1}, p_{i+2}, ..., p_n])$

$b^{\#} = F([p_1, p_2, ..., p_i]) - F([p_1, p_2, ..., p_i],[p_{i+1}, p_{i+2}, ..., p_n])$

$c^{\#} = F([p_{i+1}, p_{i+2}, ..., p_n]) - F([p_1, p_2, ..., p_i],[p_{i+1}, p_{i+2}, ..., p_n])$

$d^{\#} = N - a^{\#} - b^{\#} - c^{\#}$

Similarly, the following contingency table is obtained for the second chunked result:

$a^{\&} = F([p_1, p_2, ..., p_{i-1}],[p_i, p_{i+1}, ..., p_n])$

$b^{\&} = F([p_1, p_2, ..., p_{i-1}]) - F([p_1, p_2, ..., p_{i-1}],[p_i, p_{i+2}, ..., p_n])$

$c^{\&} = F([p_i, p_{i+1}, ..., p_n]) - F([p_1, p_2, ..., p_{i-1}],[p_i, p_{i+2}, ..., p_n])$

$d^{\&} = N - a^{\&} - b^{\&} - c^{\&}$

It is obvious that $a^{\#} = a^{\&}$. By formula (1), we know that $b^{\#} \approx b^{\&}$. By formula (2), we can derive $c^{\#} >> c^{\&}$. Since N >> a, b and c, $d^{\#} \approx d^{\&}$. Therefore,

$(a^{\#} * d^{\#} - b^{\#} * c^{\#}) << (a^{\&} * d^{\&} - b^{\&} * c^{\&})$

$(a^{\#} + b^{\#}) \approx (a^{\&} + b^{\&})$

$(a^{\#} + c^{\#}) >> (a^{\&} + c^{\&})$

$(b^{\#} + d^{\#}) \approx (b^{\&} + d^{\&})$

$(c^{\#} + d^{\#}) \approx (c^{\&} + d^{\&})$

and

$$\begin{aligned}
&\phi^2([p_1, p_2, ..., p_i],[p_{i+1}, p_{i+2}, ..., p_n]) \\
&= \frac{(a^{\#} * d^{\#} - b^{\#} * c^{\#})^2}{(a^{\#} + b^{\#}) * (a^{\#} + c^{\#}) * (b^{\#} + d^{\#}) * (c^{\#} + d^{\#})} \\
&<< \frac{(a^{\&} * d^{\&} - b^{\&} * c^{\&})^2}{(a^{\&} + b^{\&}) * (a^{\&} + c^{\&}) * (b^{\&} + d^{\&}) * (c^{\&} + d^{\&})} \\
&= \phi^2([p_1, p_2, ..., p_{i-1}],[p_i, p_{i+1}, ..., p_n])
\end{aligned}$$

The above derivation tells us: the local minimums of the ϕ^2 distribution denote plausible boundaries of two chunks. To simplify Definition 2, Definitions 3 and 4 are formulated.

Definition 3: (For Two Parts of Speech)

$a = F([p_i],[p_{i+1}])$

$b = F([p_i]) - F([p_i],[p_{i+1}])$

$c = F([p_{i+1}]) - F([p_i],[p_{i+1}])$

$d = N - a - b - c$

where p_i denotes part-of-speech i,

$F([p_i],[p_{i+1}])$ is the frequency of which p_{i+1} follows p_i,

$F([p_i])$ and $F([p_{i+1}])$ are the frequencies of p_i and p_{i+1}, and

N is the corpus size in terms of the number of words in training corpus.

It is clear that Definition 3 is the same as Definition 1. Based on Definitions 3, the probabilistic chunker is presented as follows. Note that N is the length of the tag sequence and the last chunk is always a one-tag chunk (punctuation).

```
Probabilistic_Chunker(A_Sequence_Of_Tags)
Begin
    Output("[");
    Position=1;
    Calculate $\phi^2_a$ for Current Position By Definition 3;
    Position=Position+1;
    While(Position<N)
    Begin
        Calculate $\phi^2_b$ for Current Position By Definition 3;
        Output(A_Sequence_Of_Tags[Position-1]);
        If ($\phi^2_a < \phi^2_b$) Then Output("][");
        $\phi^2_a=\phi^2_b$;
        Position=Position+1;
    End
    Output(A_Sequence_Of_Tags[N-1]);
    Output("][");
    Output(A_Sequence_Of_Tags[N]);
    Output("]");
End
```

Definition 4: (For Three Parts of Speech)

Left Chunk

$a = F([p_i, p_{i+1}],[p_{i+2}])$

$b = F([p_{i+2}]) - F([p_i, p_{i+1}],[p_{i+2}])$

$c = F([p_i, p_{i+1}]) - F([p_i, p_{i+1}],[p_{i+2}])$

$d = N - a - b - c$

where p_i denotes part-of-speech i,

$F([p_i, p_{i+1}],[p_{i+2}])$ is the frequency of which p_{i+1},p_{i+2} follows p_i,

$F([p_i, p_{i+1}])$ and $F([p_{i+2}])$ are the frequencies of (p_i, p_{i+1}) and p_{i+2}, and

N is the corpus size in terms of the number of words in training corpus.

Right Chunk

$a = F([p_i],[p_{i+1}, p_{i+2}])$

$b = F([p_{i+1}, p_{i+2}]) - F([p_i],[p_{i+1}, p_{i+2}])$

$c = F([p_i]) - F([p_i],[p_{i+1}, p_{i+2}])$

$d = N - a - b - c$

where p_i denotes part-of-speech i,

$F([p_i],[p_{i+1}, p_{i+2}])$ is the frequency of which p_{i+1},p_{i+2} follows p_i,

$F([p_i])$ and $F([p_{i+1}, p_{i+2}])$ are the frequencies of p_i and (p_{i+1}, p_{i+2}), and

N is the corpus size in terms of the number of words in training corpus.

Based on Definitions 4, the probabilistic chunker is presented as follows.

```
Probabilistic_Chunker(A_Sequence_Of_Tags)
Begin
    Output("[");
    Position=1;
    While(Position<(N-1))
    Begin
        Calculate $\phi^2_a$ for Current Position By Left Chunk of Definition 4;
        Calculate $\phi^2_b$ for Current Position By Right Chunk of Definition 4;
        Output(A_Sequence_Of_Tags[Position-1]);
        If ($\phi^2_a < \phi^2_b$) Then Output("][");
        Position=Position+1;
    End
    Output(A_Sequence_Of_Tags[N-1]);
    Output("][");
    Output(A_Sequence_Of_Tags[N]);
    Output("]");
End
```

Probabilistic chunker based on Definition 3 concerns the ϕ^2 distribution between two parts of speech. For each while loop, probabilistic chunker based on Definition 4 processes three parts of speech and concerns the ϕ^2 distribution between them.

5. Experimental Results

LOB Corpus, which is a million-word collection of present-day British English texts, is adopted as the source of training data. Susanne Corpus is adopted as the source of testing data for evaluating the performance of our probabilistic chunker. This corpus contains one tenth of Brown Corpus, but involves more syntactic and semantic information than Brown Corpus.

For evaluating the performance, a criterion [2], i.e., the content of each chunk should be dominated by one non-terminal node in Susanne parse field, is adopted. The performance evaluation model compares the chunked result C with the corresponding syntactic structure T. According to this criterion, the experimental results for Definitions 3 and 4 are shown in Table 7 as follows.

Table 7. Experimental Results for Definition 3 and 4

File	Correct Rate for Definition 3	Correct Rate for Definition 4
A01	80.31%	79.71%
G01	79.28%	79.72%
J01	76.42%	77.17%
N01	87.82%	90.10%
Average	81.13%	81.91%

The experimental results demonstrate that Definition 4 (three parts of speech) is more powerful than Definition 3 (two parts of speech). Assume the chunk length is the number of tags in a chunk. The distribution of chunk length is listed in Tables 8 and 9.

Table 8. The Distribution of Chunk Length for Definition 3

File\Length	1	2	3	4	5	6	7	8	9
A01	654	392	180	38	10	0	1	0	0
G01	715	431	167	37	09	1	1	0	0
J01	645	392	162	57	13	2	1	0	0
N01	777	418	172	55	05	1	1	0	0

Table 9. The Distribution of Chunk Length for Definition 4

File\Length	1	2	3	4	5	6	7	8	9
A01	543	373	171	56	23	05	1	0	1
G01	652	395	163	64	13	04	0	1	0
J01	573	336	190	51	21	10	5	1	0
N01	685	386	197	49	21	04	2	0	0

One-tag chunks cover about 50%. We further analyze what grammatical components constitute the one-tag chunks and find that most of the one-tag chunks contain punctuation marks, nouns and verbs. This is because proper name forms the bare subject or object. Verb is presented in the form of third person and singular, past tense, or base form. These three cases form about 62% of one-tag chunks.

By analyzing the error chunked results, we find that many errors result from conjunctions. Besides, some tags cannot be located at the end of the chunks. Therefore, the heuristic rule is applied to improve the performance. The tags that cannot be located at the end of chunks are listed as follows:

(01) AT (Singular Article)	(02) ATI (Singular or Plural Article)
(03) BED (were)	(04) BEDZ (was)
(05) BEG (being)	(06) BEM (am, 'm)
(07) BER (are, 're)	(08) BEZ (is, 's)
(09) CC (Coordinating Conjunction)	(10) CS (Subordinating Conjunction)
(11) IN (Preposition)	(12) PP$ (Possessive Determiner)
(13) WDTR (WH-Determiner)	

Applying this heuristic rule, the experimental results are listed in Table 10. It shows the usefulness of the heuristic rule. The performance increases about 10%.

Table 10. Experimental Results after Applying the Heuristic Rule

File	Correct Rate for Definition 3	Correct Rate for Definition 4
A01	92.39%	89.19%
G01	93.15%	90.92%
J01	91.30%	89.89%
N01	94.86%	94.69%
Average	92.95%	91.23%

6. Concluding Remarks

To process real text is indispensable for a practical natural language system. Probabilistic method provides a robust way to tackle with the unrestricted text. This paper proposes a probabilistic chunker to help the development of a partially bracketed corpus. Rather than using a treebank as our training corpus, LOB Corpus which is tagged with part-of-speech information only is used. The experimental results show the probabilistic chunker has more than 92% correct rate in outside test. The well-formed partially bracketed corpus is a milestone in the development of a treebank. In addition, the simple but effective chunker can also be applied to many natural language applications such as extracting the predicate-argument structures [9,10], grouping words [11] and gathering collocation [12].

The evaluation criterion adopted in this paper is not very strict. Under a strict criterion, the method proposed in this paper may not be suitable for short-fat trees. That is, it is suitable for tall-thin trees. To solve this problem, a more general definition which considers more parts of speech in contingency table is needed. However, that introduces another problem: the more the general definitions we use, the larger the tagged corpus we need. This paper also presents a tag mapper. It sets up the mapping between different tagging sets. Such an algorithm facilitates the development of a large-scale tagged corpus from different sources. By the way, much more reliable statistic information can be trained from the large-scale tagged corpus, so that the feasibility of the chunker is assured. Besides the above problem, the critical points for local minimum are not obvious in some cases. Thus their determination is also demanded in the future.

References

[1] Framis, F.R. (1994) "An Experiment on Learning Appropriate Selectional Restrictions from a Parsed Corpus," *Proceedings of COLING*, pp. 769-774, 1994.

[2] Chen, K.H. and Chen, H.H. (1994) "Extracting Noun Phrases from Large-Scale Texts: A Hybrid Approach and its Automatic Evaluation," *Proceedings of ACL*, pp. 234-241, 1994.

[3] Pocock, R.J. and Atwell, E.S. (1993) "Treebank-Trained Probabilistic Parsing of Lattices," *Technical Report 93.30*, School of Computer Studies, Leeds University, 1993.

[4] Pereira, F. and Schabes, Y. (1992) "Inside-Outside Reestimation from Partially Bracketed Corpora," *Proceedings of ACL*, pp. 128-135, 1992.

[5] Weischedel, R., *et al.* (1991) "Partial Parsing: A Report of Work in Progress," *Proceedings of DARPA Speech and Natural Language Workshop*, pp. 204-209, 1991.

[6] Sampson, G. (1993) "The Susanne Corpus," *ICAME Journal*, 17, 125-127, 1993.

[7] Johansson, S. (1986) *The Tagged LOB Corpus: Users' Manual*, Bergen: Norwegian Computing Center for Humanities, 1986.

[8] Gale, W.A. and Church, K.W. (1991) "Identifying Word Correspondences in Parallel Texts," *Proceedings of DARPA Speech and Natural Language Workshop*, pp. 152-157, 1991.

[9] Church, K.W. (1988) "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," *Proceedings of Applied Natural Language Processing*, pp. 136-143, 1988.

[10] Church, K.W., *et al.* (1989) "Parsing, Word Association and Typical Predicate-Argument Relations," *Proceedings of Parsing Technologies Workshop*, pp. 389-398, 1989.

[11] Hindle, D. (1990) "Noun Classification from Predicate-Argument Structures," *Proceedings of ACL*, pp. 268-275, 1990.

[12] Smadja, F. (1993) "Retrieving Collocations from Text: Xtract," *Computational Linguistics*, 19(1), pp. 143-178, 1993.