Computational Linguistics and Chinese Language Processing
vol. 2, no. 2, August 1997. pp. 1-24.

Building a Bracketed Corpus Using ϕ^2 Statistics

Yue-Shi Lee * Hsin-Hsi Chen +

Abstract

Research based on treebanks is ongoing for many natural language applications. However, the work involved in building a large-scale treebank is laborious and time-consuming. Thus, speeding up the process of building a treebank has become an important task. This paper proposes two versions of probabilistic chunkers to aid the development of a bracketed corpus. The basic version partitions part-of-speech sequences into chunk sequences, which form a partially bracketed corpus. Applying the chunking action recursively, the recursive version generates a fully bracketed corpus. Rather than using a treebank as a training corpus, a corpus, which is tagged with part-of-speech information only, is used. The experimental results show that the probabilistic chunker has a correct rate of more than 94% in producing a partially bracketed corpus and also gives very encouraging results in generating a fully bracketed corpus. These two versions of chunkers are simple but effective and can also be applied to many natural language applications.

Keywords: Bracketed Corpus, Probabilistic Chunkers, Treebank, ϕ^2 Statistics

1. Introduction

Research based on treebanks is ongoing for many natural language applications. Chen and Lee [1995a] introduced a Constrained Grammar extracted from the Lancaster Parsed Corpus and applied it in a linear-time partial parser. Chen and Chen [1994] proposed a probabilistic chunker to decide the implicit boundaries of constituents and utilized the linguistic knowledge to extract the noun phrases by means of a finite state mechanism.n this study, the Susanne Corpus is used as a training corpus. Framis [1994] presented amethodology to learn selectional restrictions at a variable level of abstraction from the Wall Street Journal. Bod [1993] used the ATIS Spoken language corpus as a stochastic

*Dept. of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, ROC. E-mail: leeys@nlg.csie.ntu.edu.tw
+ Dept. of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, ROC. To whom all the correspondence should be sent.

grammar for data-oriented parsing. Pocock and Atwell [1993] applied statistical grammars extracted from Spoken English Corpus to find the grammatically optimal path through a word lattice. All these applications profit from different treebanks and have shown satisfactory results.

However, the work involved in building a large-scale treebank is labor-intensive and tedious. Very few large-scale treebanks are currently available especially for languages other than English. Thus, it is worth while to study a method for developing a bracketed corpus using statistical information extracted from corpora without syntactic structures. This paper proposes two versions of probabilistic chunkers to achieve this a goal. The basic version partitions a part-of-speech sequence into chunks, which is a bracketed sequence. By applying the chunking action recursively, the recursive version generates a fully bracketed corpus. Rather than using a treebank as a 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. In this framework, the Lancaster-Oslo/Bergen (LOB) Corpus and Lancaster Parsed Corpus (LPC)[1] are adopted as the training and the testing corpus, respectively. Then, the basic version of a probabilistic chunker and its extension (recursive version) is described. Before concluding, experimental results are presented.

2. Experimental Framework

In our scheme, the input to the probabilistic chunker is a part-of-speech sequence. The basic chunker partitions this sequence into chunks. That is, each chunk contains one or more parts-of-speech. In the recursive version, a binary tree is generated. Consider an example: "Attorneys for the mayor said that an amicable property settlement has been agreed upon ." The corresponding part-of-speech sequence is listed below:

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

Then, the basic chunker partitions it into chunks, shown as follows:

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

Finally, a binary tree is generated by the recursive chunker as shown below:

1. A brief introduction to these two corpora is given in Appendix A.

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

To evaluate the performance of the chunkers, the LPC Corpus is adopted. However, the tagging sets [Chen and Lee, 1995a; Johansson, 1986] of the LPC Corpus (our testing corpus) and LOB Corpus (our training corpus) are different. The tagging set of the former is extended and modified from the latter.[2] To map a LPC tag sequence into a LOB tag sequence manually is a tedious task. Thus, an automatic tag mapping algorithm is used [Chen and Lee, 1995b]. In summary, the experimental framework is shown in Figure 1.

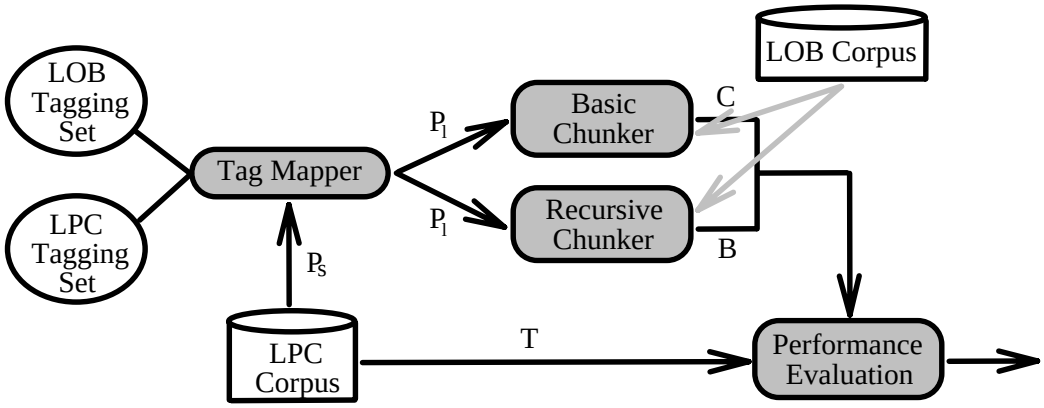

Figure 1 *Experimental Framework*

In our experiments, the test sentence P_s came from the LPC Corpus. It is a part-of-speech sequence. The corresponding syntactic structure T sent to the performance evaluation model is regarded as an evaluation criterion for the two probabilistic chunkers. Using the tag mapper, P_s is converted into P_1. Then, P_1 is input to the chunkers, and two kinds of bracketed results, i.e., partial (C) and full (B) results, are produced. Finally, the performance evaluation model reports the evaluation results according to C (B) and T.

2. In some cases, many LPC tags may be mapped into one LOB tag. For example, three LPC tags, i.e., INF (For as Preposition), INO (Of as Preposition), and INW (With as Prosposition), may be mapped into one LOB tag, i.e., IN (Preposition).

3. The Basic Version of a Probabilistic Chunker

Gale and Church [1991] proposed ϕ^2, a X^2-like statistic, to measure the association between two words. Table 1 illustrates a two-by-two contingency table for words w_1 and w_2.

	Word w_1	
Word w_2	*a*	*b*
	c	*d*

Table 1. *A Two-by-Two Contingency Table for Words w_1 and w_2.*

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 do 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:

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

where 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 below.

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 with 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 the training corpus.

Based on this definition and the ϕ^2 measure, consider the sentence "The Fulton County Grand Jury said Friday an investigation of Atlanta recent primary election produced no evidence that any irregularities took place .", which has the tag sequence "ATI NP NPL JJ NN VBD NR AT NN IN NP$ JJ JJ NN VBD ATI NN CS DTI NNS VBD NN ." Part of its syntactic structure for the first seven words is shown in Figure 2.

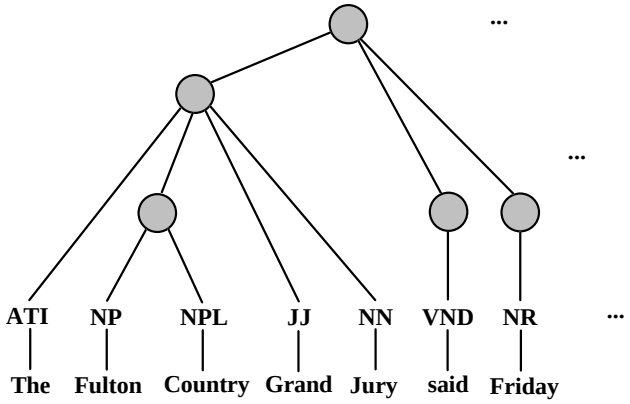

Figure 2 *The Syntactic Structure for the First Seven Words*

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}.

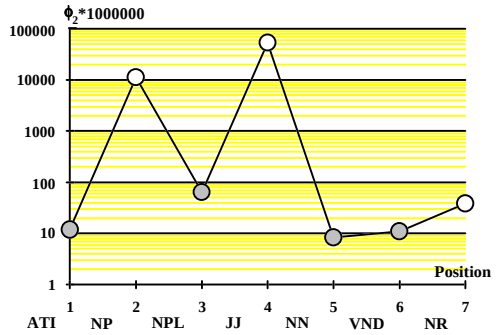

Figure 3 *The ϕ^2 Distribution for the First Seven Words*

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. (NPL and JJ), (NN and VND) and (VND and NR) have similar interpretations. Let us discuss these concepts formally. For a probabilistic chunker, a 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 with which the instance of 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 the training corpus.

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

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

Similarly,

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

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

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

Similarly,

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

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:

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

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

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

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

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

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

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

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

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

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

and

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

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

$$<< \frac{(a^{\&*} d^{\&} - b^{\&*} c^{\&})^2}{(a^{\&}+b^{\&})^* (a^{\&}+c^{\&})^* (b^{\&}+d^{\&})^* (c^{\&}+d^{\&})}$$

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

The above derivation tells us the following: the local minimums of the ϕ^2 distribution are highly correlated to syntactic boundaries between chunks. To reduce the parameters in Definition 2, Definitions 3 and 4 are formulated. In training, only the cases of two parts-of-speech and three parts-of-speech are considered.

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 with which the instance of p_{i+1} follows $p_{i,}$

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

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

BasicChunker1(A_Part_of_Speech_Sequence**)**
Begin

Output("[");

P=1;

Calculate ϕ_a^2 of Position P By Definition 3;

P=P+1;

While (P < L)

Begin

Output(A_Part_of_Speech_Sequence[P-1]);

Calculate ϕ_b^2 of Position P By Definition 3;

If ($\phi_a^2 < \phi_b^2$) **Then** Output("][");

$\phi_a^2 = \phi_b^2$;

P=P+1;

End

/* **Output the Last Word (Part-Of-Speech)** */

Output(A_Part_of_Speech_Sequence[L-1]);
Output("][");

/* **Output the Final Punctuation Mark** */

```
    Output(A_Part_of_Speech_Sequence[L]);
    Output("]");
End
```

It is clear that Definition 3 is the same as Definition 1. Based on Definition 3, the probabilistic chunker, BasicChunker1, is presented above. Note that variable L is the length of the part-of-speech sequence, and variable P denotes the current position. The comments are given between the symbols "/*" and "*/".

Definition 4: (For Three Parts-of-Speech)

Left Chunk

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

Right Chunk

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

where p_i denotes part-of-speech i,

$F([p_{i-1}, p_i],[p_{i+1}])$ is the frequency with which the instance of p_i, p_{i+1} follows p_{i-1},
$F([p_i],[p_{i+1}, p_{i+2}])$ is the frequency with which the instance of p_{i+1}, p_{i+2} follows p_i,
$F([p_{i+1}])$ and $F([p_{i-1}, p_i])$ are the frequencies of p_{i+1} and (p_{i-1}, p_i), respectively,
$F([p_i])$ and $F([p_{i+1}, p_{i+2}])$ are the frequencies of p_i and (p_{i+1}, p_{i+2}), respectively,
N is the corpus size in terms of the number of words in the training corpus.

BasicChunker2(A_Part_of_Speech_Sequence)
Begin
Output("[");
P=1;
Calculate ϕ_a^2 of Position P By the Right Chunk of Definition 4;
P=P+1;
While (P < L)
Begin
Output(A_Part_of_Speech_Sequence[P-1]);
Calculate ϕ_l^2 of Position P By the Left Chunk of Definition 4;
If (P == (L-1)) **Then** ϕ_r^2 =0;
Else Calculate ϕ_r^2 of Position P By the Right Chunk of Definition 4;
ϕ_b^2 =**max**(ϕ_l^2 , ϕ_r^2);
If (ϕ_a^2 < ϕ_b^2) **Then** Output("][");
ϕ_a^2 = ϕ_b^2 ;
P=P+1;
End
Output(A_Part_of_Speech_Sequence[L-1]); /* **Output the Last Part-Of-Speech** */
Output("][");
Output(A_Part_of_Speech_Sequence[L]); /* **Output the Final Punctuation Mark** */
Output("]");
End

The probabilistic chunker, BasicChunker2, presented above is based on Definition 4. The probabilistic chunker based on Definition 3 concerns the ϕ^2 distribution between two parts-of-speech. In order to approximate Definition 2 more accurately, Definition 4 uses three parts-of-speech instead of two parts-of-speech. Based on Definition 4, there are two possible ϕ^2 measures for each position i. They are listed below:

ϕ^2_l (for $[p_{i-1}, p_i],[p_{i+1}]$)

ϕ^2_r (for $[p_i],[p_{i+1}, p_{i+2}]$).

In BasicChunker2, we use the maximum of these two statistic measures, i.e.,max(ϕ^2_l , ϕ^2_r), as the ϕ^2 measure of position i. Note that the last chunk produced by these two chunkers is always a one-part-of-speech chunk (i.e., a punctuation mark).

4. The Recursive Version of a Probabilistic Chunker

The result produced by the basic chunker has only one level of brackets. Based on these similar concepts, it can easily be extended to a recursive chunker shown below. This chunker adopts Definition 3. In this algorithm, variable P denotes the current global minimum position. Variables LT and RT record the left and the right indices of the current part-of-speech sequence.

RecursiveChunker(LT, RT, A_Part_of_Speech_Sequence)

Begin
If (LT = RT) **Then** Output(A_Part_of_Speech_Sequence[LT]);
Else
Begin

Calculate ϕ^2_{min} of Position LT By Definition 3;

P=LT;

For I = (LT+1), ..., (RT-1) **Do**

Begin

Calculate ϕ^2_{new} of Position I By Definition 3

If ($\phi^2_{new} < \phi^2_{min}$) **Then**

Begin

$\phi^2_{min} = \phi^2_{new}$;

P=I

End

End

Output("[");

RecursiveChunker(LT, P, A_Part_of_Speech_Sequence);

Output("][");

RecursiveChunker(P+1, RT, A_Part_of_Speech_Sequence);

Output("]");

End

End

This chunker is triggered in the following way. Note that variable L is the length of the part-of-speech sequence:

```
Output("[");

RecursiveChunker(1,L,A_Part_of_Speech_Sequence);

Output("]")
```

As mentioned above, the input of the recursive chunker is a part-of-speech sequence, and the output is a binary tree. It first searches for the global minimum position of the $\phi 2$ distribution. Then, the current part-of-speech sequence is partitioned into two segments according to this position. This procedure is repeated until the length of the remaining part-of-speech sequence is equal to 1.

5. Experimental Results

In our experiments, the LOB Corpus and LPC Corpus as described in Appendix A (Tables 10 and 11) were adopted as training data and testing data, respectively. To evaluate the performance of the chunker, a criterion, crossing brackets [Black et al. 1991], was adopted. The definition is listed below:

crossing brackets = no. of constituents which violate constituent boundaries with a constituent in the treebank parse.

Consider an example. Assume that the correct result is "[A B [[C D] E [F]] [G H I] J]". The corresponding tree is shown in Figure 4.

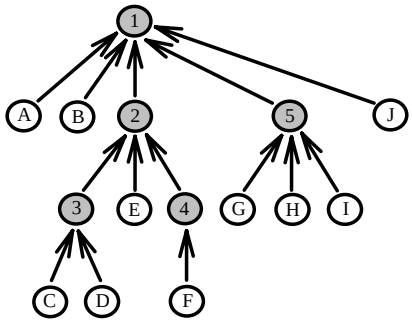

Figure 4 *A Correct Result*

If the chunked result is "[A B] [C] [D E] [F] [G H] [I J]", then 2 chunks, i.e., [D E] and [I J], are wrong. The chunk [D E] is wrong because it is dominated by two non-terminal nodes, i.e., 3 and 2. Similarly, the chunk [I J] is wrong because it is also dominated by two non-terminal nodes, i.e., 5 and 1. The other chunks, i.e., [A B], [C], [F] and [G H], are correct because they are dominated by only one non-terminal node, i.e., 1, 3, 4 and 5, respectively. Based on this criterion, the experimental results of the chunkers for Definitions 3 and 4 are shown in Tables 2 and 3. In the experiments, only four files (A01, G01, J01 and N01) randomly selected from the LPC corpus were chunked.Experimental Results of the Chunker for Definition 3

File	Total Words	Total Sent.	Total Chunks	Correct Chunks	Correct Sent.	Chunk Correct Rate for Def. 3	Sentence Correct Rate for Def. 3
A01	8,241	655	4,628	3,733	239	80.66%	36.49%
G01	6,225	367	3,339	2,533	72	75.86%	19.62%
J01	8,270	498	4,497	3,274	107	72.80%	21.49%
N01	11,653	958	6,516	5,488	415	84.22%	43.32%
Average	34,389	2,478	18,980	15,028	833	79.18%	33.62%

Table 2. *Experimental Results of the Chunker for Definition 3*

File	Total Words	Total Sent.	Total Chunks	Correct Chunks	Correct Sent.	Chunk Correct Rate for Def. 4	Sentence Correct Rate for Def. 4
A01	8,241	655	4,466	3,745	274	83.86%	41.83%
G01	6,225	367	3,303	2,653	86	80.32%	23.43%
J01	8,270	498	4,271	3,250	123	76.09%	24.70%
N01	11,653	958	6,363	5,579	491	87.68%	51.25%
Average	34,389	2,478	18,403	15,227	974	82.74%	39.31%

Table 3. *Experimental Results of the Chunker for Definition 4*

Tables 2 and 3 show the chunk correct rates and the sentence correct rates for Definitions 3 and 4. A chunked sentence is correct only if all the chunks in this sentence are all correct. The experimental results demonstrate that Definition 4 (three parts-of-speech) is more powerful than Definition 3 (two parts-of-speech). This is because the former has a wider training window size than does the latter. Assume that the chunk length is the number of parts-of-speech in a chunk. The distribution of the length of the correct chunks is listed in Tables 4 and 5.

File\Length	1	2	3	4	5	6	7	8
A01	2,213 (47.82%)	1,534 (33.15%)	633 (13.68%)	195 (4.21%)	41 (0.89%)	9 (0.19%)	2 (0.04%)	1 (0.02%)
G01	1,471 (44.06%)	1,138 (34.08%)	517 (15.48%)	152 (4.55%)	48 (1.44%)	12 (0.36%)	1 (0.03%)	0 (0%)
J01	2,065 (45.92%)	1,460 (32.47%)	669 (14.88%)	249 (5.54%)	44 (0.98%)	8 (0.17%)	2 (0.04%)	0 (0%)
N01	3,064 (47.02%)	2,136 (32.78%)	1,014 (15.56%)	244 (3.74%)	49 (0.75%)	9 (0.14%)	1 (0.01%)	0 (0%)

Table 4. *The Distribution of Chunk Length for Definition 3*

File\Length	1	2	3	4	5	6	7	8
A01	2,100 (47.02%)	1,357 (30.39%)	716 (16.03%)	213 (4.77%)	59 (1.32%)	16 (0.36%)	4 (0.09%)	1 (0.02%)
G01	1,519 (45.99%)	981 (29.70%)	550 (16.65%)	193 (5.84%)	43 (1.30%)	12 (0.36%)	5 (0.16%)	0 (0%)
J01	1,950 (45.66%)	1,170 (27.39%)	741 (17.35%)	320 (7.49%)	68 (1.59%)	17 (0.40%)	5 (0.12%)	0 (0%)
N01	3,048 (47.90%)	1,848 (29.04%)	1,072 (16.85%)	300 (4.71%)	80 (1.26%)	13 (0.20%)	1 (0.02%)	1 (0.02%)

Table 5. *The Distribution of Chunk Length for Definition 4*

One-part-of-speech chunks cover about 46%. We further analyzed what grammatical components constituted the one-part-of-speech chunks and found that most of these chunks contain punctuation marks, nouns and verbs. This is because a proper name forms the bare subject or object. A verb is presented in the form of third person and singular, past tense, or base form. These three cases form about 68% of the one-part-of-speech chunks.[3] That shows our chunker is useful for segmenting the part-of-speech sequence into significant chunks.

After analyzing the error chunked results, we find that many errors result from conjunctions. For example, consider the following example:

Instead, the kings will remain in London and wait to hear the conference's proposals.

3. The remaining 32% of the cases are randomly distributed.

The BasicChunker will propose a wrong chunk due to the conjunction "and". This is shown below:

Wrong Chunk : [in London and]

Furthermore, some parts-of-speech cannot be located at the end of chunks. Therefore, a heuristic rule is applied to improve the performance. The parts-of-speech that cannot be located at the end of chunks are partially listed as follows :[4]

(01) ABL (Pre-Qualifier)	(02) ABX (Pre-Qualifier/Double Conj.)
(03) AT (Singular Article)	(04) ATI (Singular or Plural Article)
(05) CC (Coordinating Conjunction)	(06) CD-CD (Hyphenated Pair of Cardinals)
(07) CS (Subordinating Conjunction)	(08) DOZ (does)
(09) OD (Ordinal)	(10) QL (Qualifier)
(11) PP$ (Possessive Determiner)	(12) TO (Infinitival to).

After application of this heuristic rule, Tables 6 and 7 show that the performance increases 14.43% (11.72%) and 25.31% (24.75%) for the chunk and sentence correct rates of Definition 3 (4), respectively.

File	Total Words	Total Sent.	Total Chunks	Correct Chunks	Correct Sent.	Chunk Correct Rate for Def. 3	Sentence Correct Rate for Def. 3
A01	8,241	655	5,473	5,138	410	93.88%	62.60%
G01	6,225	367	4,100	3,794	178	92.54%	48.50%
J01	8,270	498	5,560	5,129	221	92.25%	44.38%
N01	11,653	958	7,612	7,231	658	94.99%	68.68%
Average	34,389	2,478	22,745	21,292	1,467	93.61%	59.20%

Table 6. *Results of the Chunker after Applying the Heuristic Rule for Definition 3*

4. These patterns were learned from the rest of the files in the LPC Corpus; i.e., A01, G01, J01 and N01 were not included.

File	Total Words	Total Sent.	Total Chunks	Correct Chunks	Correct Sent.	Chunk Correct Rate for Def. 4	Sentence Correct Rate for Def. 4
A01	8,241	655	5,191	4,920	442	94.78%	67.48%
G01	6,225	367	3,952	3,726	203	94.28%	55.31%
J01	8,270	498	5,146	4,746	232	92.23%	46.59%
N01	11,653	958	7,249	6,952	717	95.90%	74.84%
Average	34,389	2,478	21,538	20,344	1,594	94.46%	64.33%

Table 7. *Results of the Chunker after Applying the Heuristic Rule for Definition 4*

Next, the same criterion was adopted to calculate the precision rate and the recall rate of the recursive chunker. The reason why we did not count how many bracket-pairs produced by the RecursiveChunker exactly matched the bracket-pairs in the treebank is that the RecursiveChunker produces binary trees as the parsed results, but the parsing trees in the treebank are the n-ary trees as described in Figures 4 and 5. Thus, it is not suitable to adopt such an evaluation criterion in the current situation. The reported results, however, may be too optimistic.

Assume that the correct result T is the same as the above example (see Figure 4), and that the chunked result B is "[[[A] [B]] [[C] [[[D] [E]] [[F] [[[G] [H]] [[I] [J]]]]]]]" as shown in Figure 5. To compute the precision rate, the non-terminal nodes in B are verified. That is, nodes 1-9 in B are verified. After verification, we find that non-terminal nodes, 4, 5, 6 and 9, are wrong. This is because the terminal nodes, D, E, F, G, H, I and J, are dominated by the non-terminal node 4 in B, but these terminal nodes are dominated by three non-terminal nodes (i.e., 1, 2 and 3) in T. The other three non-terminal nodes (i.e., 5, 6 and 9) in B have the similar interpretations. Thus, the precision rate of the RecursiveChunker is 0.56 (5/9) in this case.

To compute the recall rate, the roles of B and T are reversed. In other words, B is regarded as the correct answer, and the non-terminal nodes in T are verified. That is, nodes 1-5 in T are verified. After verification, we find that non-terminal nodes, 2, 3 and 5, are wrong. This is because the terminal nodes, C, D, E and F, are dominated by the non-terminal node 2 in T, but these terminal nodes are dominated by three non-terminal nodes (i.e., 3, 4 and 6) in B. The other two non-terminal nodes (i.e., 3 and 5) in T have the similar interpretations. Thus, the recall rate of the RecursiveChunker is 0.4 (2/5) in this case.

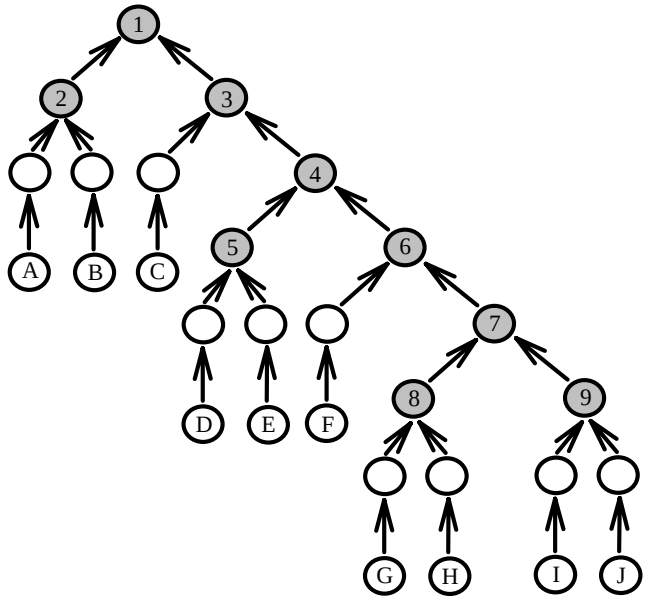

Figure 5 *A Chunked Result*

Based on these two measures, the experimental results are shown in Tables 8 and 9. These experiments were based on the Definition 4

File\Sentence Length	1-10	1-20	1-30	1-40
A01	83.30%	72.95%	68.45%	62.26%
G01	79.09%	71.28%	63.45%	60.17%
J01	77.52%	71.09%	61.14%	56.56%
N01	90.66%	76.20%	71.62%	69.14%
Average	86.14%	74.26%	65.96%	61.20%

Table 8. *The Recall Rate of the Recursive Chunker*

File\Sentence Length	1-10	1-20	1-30	1-40
A01	90.53%	84.91%	79.76%	77.92%
G01	89.59%	82.19%	78.26%	75.84%
J01	87.94%	81.78%	77.77%	75.38%
N01	94.59%	86.64%	81.41%	79.26%
Average	92.04%	83.58%	79.18%	76.89%

Table 9. *Precision Rate of the Recursive Chunker*

From Tables 8 and 9, the performance decreased with the length of the sentences. This is because the longer the sentence is, the more complex the syntactic structure is. Also, when the length of the chunks gets larger, the two/three parts-of-speech model will obtain a worse approximation. Even so, the chunker is still a very useful tool when large-scale treebanks are unavailable for grammar inference methods [Bod, 1993; Schabes et al., 1993]. One feasible way to improve the performance of our chunker is to enlarge the training corpus. Enlarging the training corpus will make a larger window size more affordable. Thus, we can approximate Definition 2 more accurately. Corpus-sharing and recent technologies such as automatic tagging [Church, 1988; Brill, 1992; Cutting et al., 1992; Elworthy, 1994; Merialds, 1994; Tapanainen and Voutilainen, 1994] and tag mapping [Chen and Lee, 1995b; Atwell et al., 1994] are suitable for a very large-scale part-of-speech-tagged corpus, and so are part-of-speech-based chunkers.

6. Concluding Remarks

This paper has proposed two versions of probabilistic chunkers to develop different levels of bracketed corpora. The basic chunker has a correct rate of more than 94% in producing a partially bracketed corpus, and the recursive chunker also gives very encouraging results in generating a fully bracketed corpus.

Besides being a milestone in the development of a treebank, this approach can be applied to many natural language applications, such as extracting noun phrases [Chen and Chen, 1994] and predicate-argument structures [Church, 1988; Church et al., 1989], grouping words [Hindle, 1990] and gathering collocation [Smadja, 1993]. Because no grammar rules are needed and part-of-speech-based corpora are much easier to get than treebanks, the chunkers presented in this paper have potential for use in the future.

Although these chunker algorithms performed well, some problems must be considered. For example, BasicChunker1 inserts a bracket at every position as the ϕ^2 statistics keep increasing; thus, it tends to produce many false alarm errors. One feasible way to solve this problem is to insert brackets only when their ϕ^2 statistic measures fall below a pre-set threshold. Otherwise, a bracket might be inserted at a particular point even though the degree of coherence is high at that point, with the ϕ^2 statistic measure being only a little less than that of the next position. However, the pre-set threshold value is also difficult to decide. This value depends on the corpus size and the domain of the corpus. This is because different corpus sizes and domains may produce different probability distributions. Thus, this threshold value is variant under these two conditions. How to formulate this threshold value as a universal value by incorporating these two features (corpus size and domain) must be investigated further.

Acknowledgments

We would like to express our gratitude to the reviewers for their valuable and constructive comments.

References

Atwell, E. et al., "AMALGAM: Automatic Mapping Among Lexico-Grammatical Annotation Models," *Proceedings of the Balancing Act - Combining Symbolic and Statistical Approaches to Language*, 1994, pp. 11-20.

Black, E. et al., "A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars," *Proceedings of DARPA Speech and Natural Language Workshop*, 1991, pp. 206-311.

Bod, R., "Using an Annotated Corpus as a Stochastic Grammar," Proceedings of 6th European Chapter of *ACL*, 1993, pp. 37-44.

Brill, E., "A Simple Rule-Based Part-of-Speech Tagger," *Proceedings of Applied Natural Language Processing*, 1992, pp. 152-155.

Brill, E., "Automatic Grammar Induction and Parsing Free Text: A Transformation-Based Approach," *Proceedings of 33rd Annual Meeting of ACL*, 1993, pp. 259-265.

Chen, K.H. and H.H. Chen, "Extracting Noun Phrases from Large-Scale Texts: A Hybrid Approach and Its Automatic Evaluation," *Proceedings of 34th Annual Meeting of ACL*, 1994, pp. 234-241.

Chen, H.H. and Y.S. Lee, "A Chunking-and-Raising Partial Parser," *Proceedings of 4th International Workshop on Parsing Technologies*, 1995a, pp. 71-78.

Chen, H.H. and Y.S. Lee, "Development of a Partially Bracketed Corpus with Part-of-Speech Information Only," *Proceedings of 3rd Workshop on Very Large Corpora*, 1995b, pp. 162-172.

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

Church, K.W. et al., "Parsing, Word Association and Typical Predicate-Argument Relations," *Proceedings of 1st International Workshop on Parsing Technologies*, 1989, pp. 389-398.

Cutting, D. et al., "A Practical Part-of-Speech Tagger," *Proceedings of Applied Natural Language Processing, 1992*, pp. 133-140.

Elworthy, D., "Does Baum-Welch Re-Estimation Help Taggers?" *Proceedings of Applied Natural Language Processing*, 1994, pp. 53-58.

Framis, F.R., "An Experiment on Learning Appropriate Selectional Restrictions from a Parsed Corpus," *Proceedings of 15th International Conference on Computational Linguistics*, 1994, pp. 769-774.

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

Hindle, D., "Noun Classification from Predicate-Argument Structures," *Proceedings of 30th Annual Meeting of ACL*, 1990, pp. 268-275.

Johansson, S., *The Tagged LOB Corpus: Users' Manua*l, Bergen: Norwegian Computing Center for Humanities, 1986.

Merialds, B., "Tagging English Text with a Probabilistic Model," *Computational Linguistics*, Vol. 20.2(199): 155-171.

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

Schabes, Y. et al., "Parsing the Wall Street Journal with the Inside-Outside Algorithm," *Proceedings of 6th European Chapter of ACL*, 1993, pp. 341-347.

Smadja, F., "Retrieving Collocations from Text: Xtract," *Computational Linguistic*s, 19. 1(1993): 143-178.

Tapanainen, P. and A. Voutilainen, "Tagging Accurately - Do'nt Guess If You Know," Proceedings of Applied Natural Language Processing, 1994, pp. 47-52.

Appendix A: A Brief Introduction of LOB Corpus and LPC Corpus

The Lancaster-Oslo/Bergen (LOB) Corpus is a million-word collection of present-day British English texts. It contains 500 text samples of approximately 2,000 words dis-

tributed over 15 categories. In this corpus, each word is accompanied by a part-of-speech tag. There is no syntactic bracketing. Table 10 gives an overview of the Lancaster Parsed Corpus.

Category	# of Words	Category	# of Words	Category	# of Words	Category	# of Words
A	89,139	E	76,916	J	161,907	N	59,390
B	54,447	F	89,094	K	59,205	P	59,382
C	34,321	G	155,342	L	49,145	R	18,203
D	34,388	H	60,769	M	12,120	**Total**	1,013,768

Table 10. *An Overview of the LOB Corpus*

The Lancaster Parsed Corpus is a modified and condensed version of the LOB Corpus. It only contains one sixth of the LOB Corpus but involves more information than the LOB Corpus. The corpus consists of fifteen kinds of texts (about 150,000 words). Each category corresponds to one file. The following shows a snapshot of the Lancaster Parsed Corpus.

A01 1

[S[P by_IN [N Trevor_NP Williams_NP N]P] ._. S]

A01 2

[S[N a_AT move_NN [Ti[Vi to_TO stop_VB Vi][N \0Mr_NPT Gaitskell_NP N][P from_IN [Tg[Vg nominating_VBG Vg][N any_DTI more_AP labour_NN life_NN peers_NNS N]Tg]P]Ti]N][V is_BEZ V][Ti[Vi to_TO be_BE made_VBN Vi][P at_IN [N a_AT meeting_NN [Po of_INO [N labour_NN \0MPs_NPTS N]Po]N]P][N tomorrow_NR N]Ti] ._. S]

A01 3

[S&[N \0Mr_NPT Michael_NP Foot_NP N][V has_HVZ put_VBN V][R down_RP R][N a_AT resolution_NN [P on_IN [N the_ATI subject_NN N]P]N][S+ and_CC [Na he_PP3A Na][V is_BEZ V][Ti[Vi to_TO be_BE backed_VBN Vi][P by_IN [N \0Mr_NPT Will_NP Griffiths_NP ,_, [N \0MP_NPT [P for_IN [N Manchester_NP Exchange_NP N]P]N]N]P]Ti]S+] ._. S&]

A01 4

[S[Fa though_CS [Na they_PP3AS Na][V may_MD gather_VB V][N some_DTI left-wing_JJB

support_NN N]Fa] ,_, [N a_AT large_JJ majority_NN [Po of_INO [N labour_NN \0MPs_NPTS N]Po]N][V are_BER V][J likely_JJ J][Ti[Vi to_TO turn_VB Vi][R down_RP R][N the_ATI Foot-Griffiths_NP resolution_NN N]Ti] ._. S]

A01 5

*'_*' [S[V abolish_VB V][N Lords_NPTS N] **'_**' ._. S]

These are extracted from the first five sentences of category A. Before each sentence, a unique reference number, e.g., "A01 1", denotes its source. Each word is appended with a lexical tag, e.g., "by_IN", "Trevor_NP". The syntactic tag is shown by means of opening and closing brackets.

To indicate that phrases or clauses are coordinated, the symbols "&", "-" or "+" will be used at the end of a phrase or a clause tag. An example is listed as follows:

> [N& mothers_NNS,_,[N- children_NNS N-] [N+and _CC sick_JJ people_NNS N+] N&]

The first coordinated phrase is not labeled with any tag. The second and the third coordinated phrases are labeled N- and N+, respectively. This is because N- or N+ tends to include an ellipsis. Table 11 gives an overview of the Lancaster Parsed Corpus. In our experiments, those parsed sentences that did not begin with "[S" and end with "S]" were removed from the corpus. Thus, "A01 5" was deleted.

Category	# of Words	Category	# of Words	Category	# of Words	Category	# of Words
A	9,410	E	9,356	J	8,336	N	15,751
B	9,999	F	8,562	K	13,587	P	16,766
C	8,225	G	6,813	L	15,556	R	9,443
D	10,110	H	6,524	M	9,179	**Total**	157,617

Table 11. *An Overview of the LPC Corpus*