Automatic Learning of Context-Free Grammar

Tai-Hung Chen Chun-Han Tseng
M9430400{71,41}@student.nsysu.edu.tw
Chia-Ping Chen
cpchen@cse.nsysu.edu.tw
Department of Computer Science and Engineering
National Sun Yat-Sen University

Abstract

In this paper we study the problem of learning context-free grammar from a corpus. We investigate a technique that is based on the notion of minimum description length of the corpus. A cost as a function of grammar is defined as the sum of the number of bits required for the representation of a grammar and the number of bits required for the derivation of the corpus using that grammar. On the Academia Sinica Balanced Corpus with part-of-speech tags, the overall cost, or description length, reduces by as much as 14% compared to the initial cost. In addition to the presentation of the experimental results, we also include a novel analysis on the costs of two special context-free grammars, where one derives only the set of strings in the corpus and the other derives the set of arbitrary strings from the alphabet.

Index Terms: context-free grammar, Chinese language processing, description length, Academia Sinica Balanced Corpus.

1 Introduction and Overview

In this paper we study the problem of learning context-free grammar (CFG) [1] from a corpus of part-of-speech tags. The framework of CFG, although not complex enough to enclose all human languages [2], is an approximation good enough for many purposes. For a natural language, a "decent" CFG can derive most sentences in the language. Put differently, with high probability, a sentence can be parsed by a parser based on the CFG.

The main issue with CFG is how to get one. Generally speaking, learning context-free grammar from sample text is a difficult task. In [3], a context-free grammar which derives exactly one string is reduced to a simpler grammar generating the same string. This achieves a lossless data compression. In [4], an algorithm of time complexity $O(N^2)$ for learning stochastic context-free grammar (SCFG) is proposed, where N is the number of non-terminal symbols. This is a great reduction from the inside-outside algorithm which requires $O(N^3)$.

Context-free grammars can be used in many applications. In [5], an automatic speech recognition system uses a dynamic programming algorithm for recognizing and parsing spoken word strings of a context-free grammar in the Chomsky normal form. CFG can

also be used in software engineering. In [6], the components in a source code that need to be renovated are recognized and new code segments are generated from context-free grammars. In addition, since parsing outputs larger and less-ambiguous meaning-bearing structures in the sentence, for high-level natural language processing tasks such as question answering [7] and interactive voice response [8] systems, the design and implementation of CFG can be crucial to their success.

If the goal of learning is to acquire a grammar that derives most sentences in the domain of interest, then a good one is apparently domain-specific. An all-purpose CFG is not likely to be the best since it tends to derive a much larger set than is necessary. We thus propose to learn CFGs from corpus. The basic problem is this: *Given a set of sentences, we want to find a set of derivation rules that can derive the original set of sentences.* Note that there are infinitely many CFGs from which the original set of sentences can be derived. To discriminate one CFG from another, we will consider the costs they incur in deriving the original corpus. The cost functions will be defined shortly. Thus, we are proposing to *find the set of rules that can derive the original language with the minimum cost.*

This paper is organized as follows. Following this introduction and review, we analyze two special cases of CFG and the proposed rules in Section 2. The experimental results are presented in Section 3 followed by discussion and comments. In Section 4, we summarize our work.

2 Mathematical Analysis

2.1 The Cost Functions

There are two different kinds of costs in the description of a corpus by a CFG. The first kind is incurred from the representation of the CFG. A rule in a CFG is of the form

$$A \to \beta. \tag{1}$$

It consists of a non-terminal symbol A on the left-hand side and a string of symbols β on the right-hand side. The cost of a rule is the number of bits needed to represent the left-hand side and right-hand side. For (1), this is

$$C_R = (1 + |\beta|) \log |\Sigma|, \tag{2}$$

where Σ is the symbol set and $|\Sigma|$ is the number of symbols in Σ.

The second kind is the cost to derive the sentences given the rules. In order to derive a sentence W, the sequence of rules must be specified in the derivation from S [1] to W,

$$S \Rightarrow \alpha_1 \Rightarrow \cdots \Rightarrow W, \quad \text{or } S \stackrel{*}{\Rightarrow} W, \tag{3}$$

where we have adopted the notation defined in [1]. The sequence of rules always starts with one of the S-derivation rules[2],

$$S \to \alpha. \tag{4}$$

This step results in a derived string α. If there is no non-terminal symbols in α, we are done with the derivation. Otherwise, we expand the left-most non-terminal symbol, say X,

[1] S is known as the sentence symbol or the start symbol.

[2] The Z-derivation rules are those with Z as the left-hand side.

in α by one of its derivation bodies[3]. The process continues until there is no non-terminal symbols in the derived string, which will be the sentence W at that point. To illustrate, suppose we are given the CFG

$$\begin{cases} R_1(S): & S \to XXC \\ & \vdots \\ R_1(X): & X \to AB \\ & \vdots \end{cases}$$

and we want to derive the sentence $W = ABABC$. For this example, one can verify that the derivation sequence is $R_1(S)R_1(X)R_1(X)$, where $R_t(Z)$ represents the tth Z-derivation rule. The cost is

$$C_D = \sum_{k=1}^{m} \log |R(s_k)| = \log |R(S)| + \log |R(X)| + \log |R(X)|, \tag{5}$$

where m is the number of rules in the derivation sequence, s_k is the non-terminal symbol for the kth derivation, and $|R(s_k)|$ is the number of rules in the CFG using s_k as the left-hand side.

Combining (2) and (5), the total cost is

$$C = \sum_{i=1}^{p} C_R(i) + \sum_{j=1}^{q} C_D(j) = \sum_{i=1}^{p} n_i \log |\Sigma| + \sum_{j=1}^{q} \sum_{k=1}^{m_j} \log |R(s_k)|, \tag{6}$$

where p is the number of rules, q is the number of sentences, n_i is the number of symbol tokens in rule i, and m_j is the length of the derivation sequence for sentence j.

2.2 Special-Case Analysis

We will analyze the costs for two special CFGs in this section. The first CFG, which we call the *exhaustive* CFG, uses every distinct sentence in the corpus as a direct derivation body of the start symbol S. The corpus is thus covered trivially. To compute the cost, we first rearrange the sentences in the lexicographic order and then move the repeated sentences to the back. The number of symbols for a rule is simply the number of words of the corresponding sentence n_w, plus 1 (for the start symbol S), and $|\Sigma|$ is the vocabulary size $|V|$ of the corpus plus 1 (again for the start symbol). Thus the rule cost is

$$C_R = n \log |\Sigma| = (n_w + 1) \log(|V| + 1). \tag{7}$$

In this case, each sentence is derived from S in one step, by specifying the correct one out of the $|R(S)|$ rules. Thus the derivation cost for a sentence is

$$C_D = \log |R(S)|. \tag{8}$$

Note that q is generally not equal to $|R(S)|$ as there may be repeated sentences. Combining (7) and (8), the total cost for the exhaustive CFG is

$$C = \sum_{i=1}^{|R(S)|} C_R(i) + \sum_{j=1}^{q} C_D(j) = \sum_{i=1}^{|R(S)|} (n_w(i) + 1) \log(|V| + 1) + q \log |R(S)|. \tag{9}$$

[3]This is also known as the leftmost derivation.

The second case, which we call the *recursive* CFG, uses recursive derivation for S,

$$S \to AS, \tag{10}$$

where the non-terminal A can be expanded to be any word in the vocabulary. Combined with the rule $S \to \epsilon$, this CFG clearly covers any string of the alphabet, Σ^*, which is a much larger set than any real corpus.

The rule cost is significantly smaller in recursive CFG than that of the exhaustive CFG. The only rules are the two instances of S-derivation and the $|V|$ instances of A-derivation, so the rule cost is

$$C_R = n \log |\Sigma|, \tag{11}$$

where n can be $1, 2$ or 3 depending on the rule. The derivation cost, however, is much larger. To derive a sentence W of n_w words, the recursive rule of S and substitution rule of A have to be applied alternatively for n_w times, followed by a final rule of $S \to \epsilon$. Thus the derivation cost for a sentence is

$$C_D = n_w(1 + \log |V|) + 1. \tag{12}$$

Combining (11) and (12), the total cost for the recursive CFG is

$$\begin{aligned} C &= \sum_{i=1}^{2+|V|} n_i \log |\Sigma| + \sum_{j=1}^{q} C_D(j) \\ &= (4 + 2|V|) \log(|V| + 2) + \sum_{j=1}^{q} [n_w(j)(1 + \log |V|) + 1]. \end{aligned} \tag{13}$$

In Table 1 we list the costs of these cases computed on the Academia Sinica Balanced Corpus [9] (ASBC). The exhaustive CFG has a large rule cost (28.1 million bits) and a small derivation cost (4.1 mb). The recursive CFG has an extremely small rule cost (merely 607 bits) and an extremely large derivation cost (88.4 mb). To overall cost is higher for the recursive CFG (88.4 mb) than the exhaustive CFG (32.2 mb). From this table, one can see that there is a trade-off between the rule cost and the derivation cost. In addition, the numbers illustrate the important point that minimizing the rule cost alone will lead to a CFG that is inappropriate.

The exhaustive CFG is too restricted in the sense that it covers only those sentences seen in the learning corpus. The recursive CFG is too broad in the sense that it covers all sentences including the non-sense ones. Our goal is to strike a balance between these two extremes.

2.3 Proposed Rules

The special cases we analyze above do not have the minimum cost of all possible CFGs from which the corpus can be derived. To reduce the overall cost, we start with the initial CFG and then iteratively look for a new CFG rule. The kind of rules we investigate in this study is of the form

$$X \to YZ.$$

The introduction of such a rule to the exhaustive CFG described in Section 2.2 has the following impacts on the cost:

- Each occurrence of YZ is replaced by X, so the total number of symbol tokens in the S derivation rules is reduced.
- $|\Sigma|$ is incremented by 1.
- The derivation cost may or may not change, depending on whether two or more of the S-derivation rules become identical.

Since there are two symbols on the right-hand side, the number of candidate rules is $|\Sigma \times \Sigma| = |\Sigma|^2$, where Σ is the current symbol set. To choose one, we compute the bigram counts of all bigrams and use the bigram with the highest count as the right-hand side of the new rule, whose left-hand side is a new symbol.

3 Experiments

3.1 Data Preparation

We use the ASBC corpus for our experiments. In this corpus, the part-of-speech tag is labeled for each word. On the raw text data, we apply the following pre-processing steps:

1. The punctuation of period, question mark and exclamation mark are used to segment a sentence into multiple sentences.
2. The parenthesis tags are discarded.
3. The part-of-speech tag sequence is extracted for each sentence.

The initial statistics of the data after pre-processing is summarized in Table 2. A total of 229852 sentences are extracted and 203651 of them are distinct. The total number of tokens is 4.84 millions. Note that in the experiments, the symbols are the part-of-speech tags rather than the words for our CFG learning algorithm. This approach focuses more directly on the syntax and alleviates the issue of data sparsity.

3.2 Results

The learning process is an iterative algorithm. We start with the exhaustive CFG introduced in Section 2.2. In each epoch, we

1. compute the bigram counts for each bigram,
2. make a new rule with the bigram of the largest count as the right-hand side,
3. update the alphabet (symbol set), rules and derivations,
4. update the costs.

The representation cost as a function of the number of learned rules is presented in Figure 1. There are three curves in the plot, representing the rule cost, the derivation cost and the total cost. The initial cost is 32.2 million bits, as we show in Section 2.2. As the learning process progresses, the two kinds of cost behave in different ways: the derivation cost stays

constant while the rule cost decreases. The derivation cost is invariant for two reasons: 1) the number of S-derivation rules does not change and 2) there is no ambiguity in expanding non-S symbols, in our current learning scheme. The rule cost reduces because the decrease in the number of tokens in the rules outweighs the increase in the size of symbol set. As a result, the total cost reaches a minimum of 27.7 million bits when the 92nd rule is learned. The cost reduction is 14.0%. After the 92nd rule, the largest bigram count is not high enough for the reduction of the number of tokens to outweigh the increase in the alphabet, so the cost increases. The maximum bigram count is plotted against the epoch (number of rules learned) in Figure 2. From this figure, one can see that the maximum bigram count decreases very fast.

The top-20 rules learned from ASBC are listed in Table 3. In this table, we also include examples of words and sentences from ASBC. In addition, the definition and more examples of the part-of-speech tags are listed in Table 4. From Table 3, one can see that the new symbols (M1, ..., M20) here indeed represents larger phrasal structures than the basic part-of-speech tags. Furthermore, M7 and M9 embed M1, giving evidence for a deep parsing structure. In Figure 3, two sentences in ASBC parsed based on the learned CFG (left) and parsed manually (right) are shown. We can see that the verb phrase (VP) structure of sentence (a) in both parses. For sentence (b), the VP is scattered in two subtrees $M40$ and $M66$. The symbol $M66$ can be identified as a noun phrase (NP).

4 Summary

The construction of a context-free grammar for a specific domain is a non-trivial task. To learn a CFG automatically from corpus, we define a cost function as the number of bits for the representation of CFG and sentence derivation. Our objective is to find a grammar that covers the learning corpus with the minimum cost. We analyze two extreme cases to illustrate the framework. The proposed rules are learned from heuristic bigram counting. The results show that on ASBC corpus, the reduction of cost is 14.0% of the initial cost.

There are other kinds of CFG rules that are not considered in this study, such as the $A \rightarrow B|C$ rules. The candidate set of rules should be enlarged for more descriptive power. Another line of research is to extend the current work to the word level (as opposed to the part-of-speech level). This should be doable at least in a restricted domain. Finally, from the data compression and information theory [10], one can design a different cost function that takes the symbol frequencies into account and achieves further reduction on the number of bits.

5 Acknowledgement

This work is supported by National Science Council under grant number 94-2213-E-110-061. We thank Sheng-Fu Wang and Chiao-Mei Wang for inspirational discussions. We also thank the reviewers for the thorough comments.

Table 1: Costs in bits of exhaustive (G1) and recursive (G2) CFGs.

	rule cost	derivation cost	total cost
G1	28.1m	4.1m	32.2m
G2	607	88.4m	88.4m

Table 2: Initial data statistics for ASBC after text pre-processing. $|V|$ is the vocabulary size, q is the total number of sentences, $|R(S)|$ is the total number of distinct sentences, N_q is the total number of tokens in the corpus, and N_R is the total number of tokens in the distinct sentences.

$\|V\|$	q	$\|R(S)\|$	N_q	N_R
51	229852	203651	4838540	4729276

Table 3: Top-20 rules learned from the ASBC corpus.

X → Y+Z	例子(Y)	例子(Z)	例句
M1 → DE+Na	之	需要	研究計畫之需要
M2 → Na+Na	人際	關係	與你暢談人生、信仰、求學、工作、愛情、人際關係
M3 → Neu+Nf	第一	次	有三成三的大學生第一次有投票權
M4 → Na+D	領域	已	在此一領域已有傑出之研究成果
M5 → D+D	應	不致於	但黃國章應不致於爲此遭人持槍狙擊
M6 → D+VC	應	擬定	政府應擬定合理的都市政策
M7 → Na+M1	計畫	之需要	「台灣與東南亞土著文化與血緣關係」主題研究計畫之需要
M8 → Na+VC	院校	發出	十所大學院校發出四百份問卷
M9 → VH+M1	冷靜	的判斷力	冷靜的判斷力
M10 → DE+Nv	之	研究	在此一領域已有傑出之研究成果
M11 → VH+Na	有效	問卷	有效問卷爲三百八十五份
M12 → P+Na	在	團體	在團體中被依賴
M13 → P+Nc	對	台大	對台大、政大等十所大學院校發出四百份問卷
M14 → Nh+D	他人	一同	和他人一同行動時
M15 → Nep+Nf	這	種	正屬於這種類型
M16 → VC+Na	領導	者	你是屬於領導者型
M17 → Nc+Na	大學	院校	十所大學院校發出四百份問卷
M18 → Dfa+VH	太	遠	回家路途太遠
M19 → D+VH	不	怒	也是不怒而威
M20 → D+SHI	就	是	這就是AB型-牡羊座的一般傾向

Table 4: Selected part-of-speech tags used in the ASBC corpus.

Name	詞性	例子
A	形容詞	特有，一般，主要
D	副詞	應，已，一邊，再，就，不，只有，常常，也，都，必須
DE	語助詞	的，之，地，得
Dfa	副詞（前置）	十分，相當，很，非常，過於，最，太，較，過度，極
Na	名詞	人際，關係，父子，問題，原因，暴動，宗教
Nc	名詞（地方）	出發點，天下，家，醫院，全身，廚房，學校，印尼
Neu	定詞（數量）	一，第一，七十五，兩，幾，十餘，九萬
Nep	定詞	這，其，那，此，其中，什麼，哪
Nf	量詞	個，些，步，歲，次，種，件，位，項
Nh	名詞（代名詞）	我們，對方，自己，別人，大家，雙方
Nv	可當動詞或名詞	研究，存在，否定，演奏，製作，輔導，離婚
P	介詞	於，至，關於，和，譬如，以，將，爲
SHI	及物動詞	是
VH	不及物動詞	普遍，好，疏離，可怕，直接，相當，慈愛
VC	及物動詞	寫好，丟，翻，完成，求，存，做，出

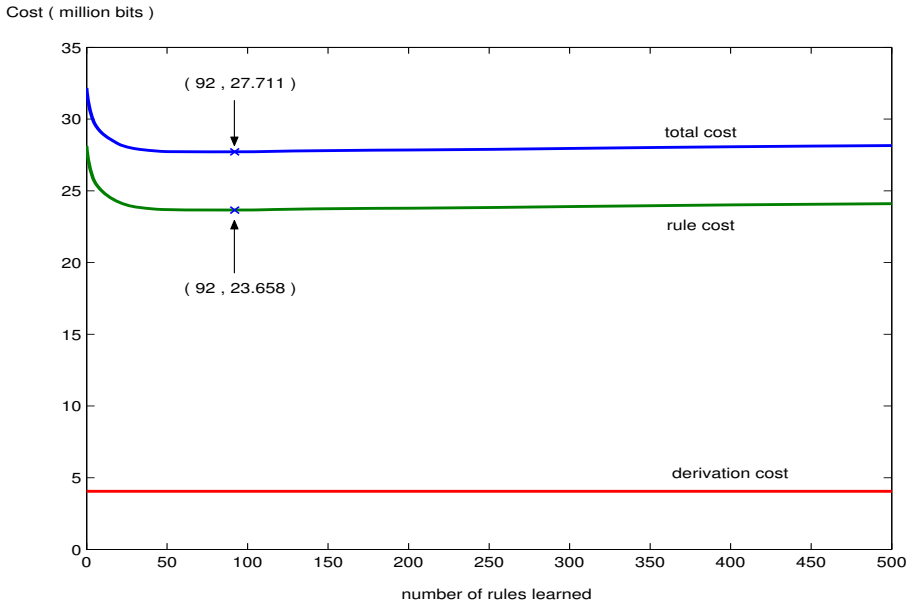

Figure 1: The cost as a function of the number of learned rules.

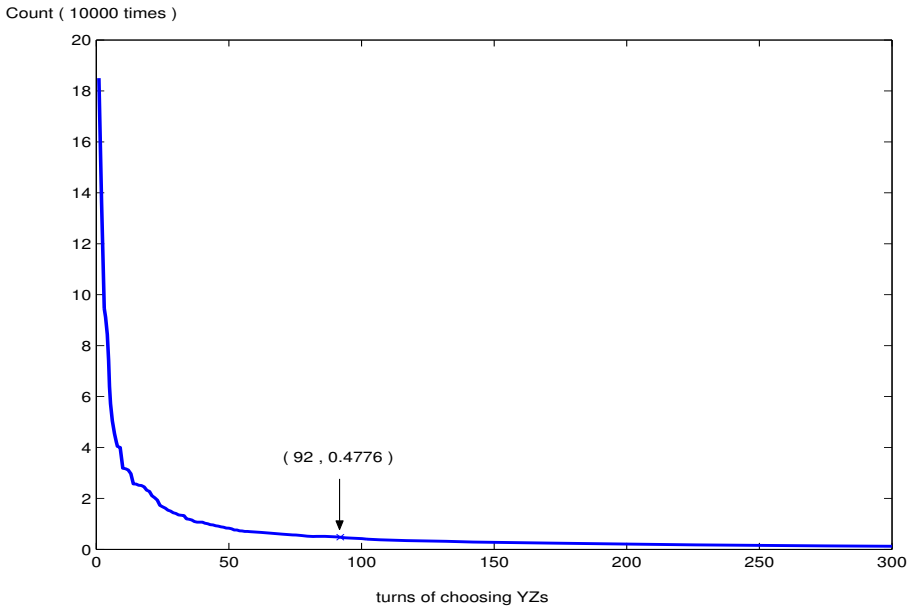

Figure 2: The maximum bigram count as a function of the number of epochs.

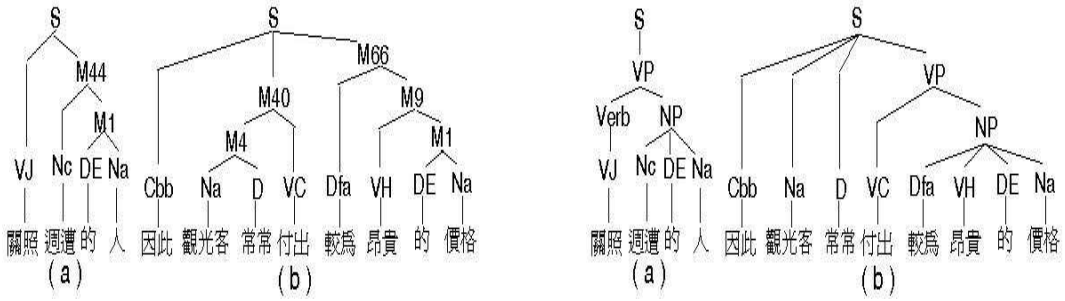

Figure 3: Examples parsed by the learned CFG (left) and parsed manually (right). Here Cbb is conjunctive and VJ is transitive verb.

References

[1] J. E. Hopcroft, R. Motwani and J. D. Ullman, “Introduction to Automata Theory, Languages and Computation”, Addison-Wesley (2001).

[2] D. Jurafsky and J. H. Martin, “Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition”, Prentice Hall (2000).

[3] John C. Kieffer and En-hui Yang, “Design of context-free grammars for lossless data compression,” Proceedings of the 1998 IEEE Information Theory Workshop, pp. 84-85.

[4] H. Lucke, “Reducing the computation complexity for inferring stochastic context-free grammar rules from example text”, Proceedings of ICASSP 1994, pp. 353-356.

[5] H. Ney, “Dynamic Programming Speech Recognition Using a Context-Free Grammar”, Proceedings of ICASSP’87, pp. 69-72.

[6] Mark van den Brand, Alex Sellink, and Chris Verhoef, “Generation of components for software renovation factories from context-free grammars”, In Working Conference on Reverse Engineering, IEEE Computer Society, WCRE97, pp. 144-153.

[7] C. Yuan and C. Wang, “Parsing model for answer extraction in Chinese question answering system”, Proceedings of IEEE NLP-KE ’05, pp. 238 - 243.

[8] M. Balakrishna, D. Moldovan, E.K. Cave, “Automatic creation and tuning of context free grammars for interactive voice response systems”, Proceedings of IEEE NLP-KE ’05, pp. 158 - 163.

[9] 中央研究院平衡語料庫的內容與說明, http://www.sinica.edu.tw/SinicaCorpus/98-04.pdf.

[10] T. Cover and J. Thomas, “Elements of Information Theory”, John Wiley and Sons (1991).