ESTIMATION OF THE PROBABILITY DISTRIBUTIONS OF STOCHASTIC CONTEXT-FREE GRAMMARS FROM THE k-BEST DERIVATIONS

Joan-Andreu Sánchez *José-Miguel Benedí*

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia (Spain)
e-mail: {jandreu,jbenedi}@dsic.upv.es

ABSTRACT

The use of the Inside-Outside (IO) algorithm for the estimation of the probability distributions of Stochastic Context-Free Grammars (SCFGs) in Natural-Language processing is restricted due to the time complexity per iteration and the large number of iterations that it needs to converge. Alternatively, an algorithm based on the Viterbi score (VS) is used. This VS algorithm converges more rapidly, but obtains less competitive models. We describe here a new algorithm that only considers the k-best derivations in the estimation process. The experimental results show that this algorithm achieves faster convergence than the IO and better models than the VS algorithm.

1. INTRODUCTION

Stochastic Context-Free Grammars (SCFGs) are an appealing alternative for Language Modeling in real tasks of Syntactic Pattern Recognition [7] and Natural Language Processing [3]. The reason for this can be found in the capability of SCFGs to express long-distance dependencies and to be potentially more compact than the n-gram models. However, the learning of SCFGs is still an important obstacle for their application to language modeling. The most widely-used method for learning SCFGs is based on the well-known Inside-Outside (IO) algorithm [2]. Unfortunately, the application of this algorithm presents important problems which are accentuated in real tasks: the time complexity per iteration and the large number of iterations that are necessary to converge.

An alternative to the IO algorithm is an estimation algorithm based on the Viterbi score (VS algorithm) [2]. The convergence of the VS algorithm is faster than the IO algorithm, since the VS algorithm only considers the information obtained from the best derivation. However, the obtained SCFGs are, in general, worse learned.

In the same way, another alternative is to consider a modification of the IO algorithm which learns SCFGs from partially bracketed corpora [8]. This algorithm only considers the possible derivations according to partial parses defined on the bracketed corpus. The results reported in [8] show that the convergence of this algorithm is significantly faster than the IO algorithm. The drawback of this proposal is the need for large corpora which are manually parsed and bracketed.

Here we describe a new algorithm for the estimation of the probability distributions of SCFGs from the k-best derivations. This algorithm considers more information than the VS algorithm, and therefore the SCFGs are, in general, better estimated. It is not necessary to incorporate deductive information to the sample, in contrast to [8]. The number of iterations increases as the value of k grows up.

In the following section, some problems which are related to the IO and VS algorithms are presented together with the notation used. Next, the new estimation algorithm of SCFGs from the k-best derivations is proposed. Finally, the experiments illustrating the behaviour of this algorithm are reported.

2. NOTATION AND DEFINITIONS

A *Context-Free Grammar* (CFG) G is a four-tuple (N, Σ, P, S), where N is a finite set of non-terminal symbols, Σ is a finite set of terminal symbols ($N \cap \Sigma = \emptyset$), P is a finite set of rules of the form $A \rightarrow \alpha$ ($A \in N$ and $\alpha \in (N \cup \Sigma)^+$) (we only consider grammars with no empty rules) and S is the initial symbol ($S \in N$). A CFG in Chomsky Normal Form is a CFG in which the rules are of the form $A \rightarrow BC$ or $A \rightarrow a$ ($A, B, C \in N$ and $a \in \Sigma$). A *left-derivation* of $x \in \Sigma^+$ in G, is a sequence of rules $d_x = (p_1, p_2 \ldots, p_m)$, $m \geq 1$ such that: $(S \overset{p_1}{\Rightarrow} \alpha_1 \overset{p_2}{\Rightarrow} \alpha_2 \ldots \overset{p_m}{\Rightarrow} x)$, where $\alpha_i \in (N \cup \Sigma)^+$, $1 \leq i \leq m-1$, and p_i rewrites the left-most non-terminal of α_{i-1}. The *language generated* by G is defined as $L(G) = \{x \in \Sigma^+ \mid S \overset{+}{\Rightarrow} x\}$.

A *Stochastic Context-Free Grammar* (SCFG) G_s is defined as a pair (G, q) where G is a CFG and $q : P \rightarrow]0, 1]$ is a probability function of rule application such that $\forall A \in N$: $\sum_{\alpha \in (N \cup \Sigma)^+} q(A \rightarrow \alpha) = 1$. Let d_x be a left-derivation (derivation from now on) of the string x, the expression $\mathrm{N}(A \rightarrow \alpha, d_x)$ represents the number of times that the rule $A \rightarrow \alpha$ has been used in the derivation d_x and $\mathrm{N}(A, d_x)$ is the number of times that the non-terminal A has been derived in d_x. We define the *probability* of the derivation d_x of the string x as: $\Pr(x, d_x \mid G_s) = \prod_{(A \rightarrow \alpha) \in P} q(A \rightarrow \alpha)^{\mathrm{N}(A \rightarrow \alpha, d_x)}$. Let Δ_x be a finite set of different derivations of the string x. We define the *probability* of the string x with respect to Δ_x as: $\Pr(x, \Delta_x \mid G_s) = \sum_{d_x \in \Delta_x} \Pr(x, d_x \mid G_s)$. When Δ_x is the set of all possible derivations then we have the *probability* of the string x and is noted as: $\Pr(x \mid G_s) = \sum_{\forall d_x} \Pr(x, d_x \mid G_s)$. We define the *probability of the best*

derivation of the string x from a set of derivations Δ_x as: $\widehat{\Pr}(x, \Delta_x \mid G_s) = \max_{d_x \in \Delta_x} \Pr(x, d_x \mid G_s)$, and we define the *best derivation*, $\widehat{d}_x$, as the argument which maximizes this function. The *language generated* by G_s is defined as $L(G_s) = \{x \in L(G) | \Pr(x \mid G_s) > 0\}$.

Given an initial SCFG G_s and a training sample Ω, the probabilities can be modified to obtain a new SCFG $G_s^{'} = (G, q')$ using the following expression $(\forall (A \to \alpha) \in P)$:

$$q'(A \to \alpha) = \frac{\sum_{x \in \Omega} \frac{1}{\Pr(x|G_s)} \sum_{\forall d_x} \mathrm{N}(A \to \alpha, d_x) \Pr(x, d_x \mid G_s)}{\sum_{x \in \Omega} \frac{1}{\Pr(x|G_s)} \sum_{\forall d_x} \mathrm{N}(A, d_x) \Pr(x, d_x \mid G_s)}. \quad (1)$$

This transformation attempts to improve the function $\Pr(\Omega \mid G_s) = \prod_{x \in \Omega} \Pr(x \mid G_s)$ guaranteeing that $\Pr(\Omega \mid G_s^{'}) \geq \Pr(\Omega \mid G_s)$ [1].

In a similar way the probabilities can be modified with the expression $(\forall (A \to \alpha) \in P)$:

$$q'(A \to \alpha) = \frac{\sum_{x \in \Omega} \mathrm{N}(A \to \alpha, \widehat{d}_x)}{\sum_{x \in \Omega} \mathrm{N}(A, \widehat{d}_x)}. \quad (2)$$

where $\widehat{d}_x$ is obtained by maximizing on all possible derivations of x. This transformation attempts to improve the function $\Pr(\Omega, \widehat{d}_\Omega \mid G_s) = \prod_{x \in \Omega} \widehat{\Pr}(x, \widehat{d}_x \mid G_s)$ guaranteeing that $\Pr(\Omega, \widehat{d}_\Omega \mid G_s^{'}) \geq \Pr(\Omega, \widehat{d}_\Omega \mid G_s)$. The estimated models are consistent, such as is demonstrated in [9].

The well-known IO algorithm consists on applying transformation (1) iteratively on an initial SCFG G_s until a local maximum of the function being maximized is achieved. Likewise, the VS algorithm consists in applying transformation (2) iteratively until a local maximum of the function being maximized is achieved. Both algorithms are habitually formulated in a different way to work with SCFGs in Chomsky Normal Form and they can be effectively computed using a Dynamic Programming scheme. These algorithms have a time complexity $O(|\Omega| n^3 |P|)$ in each iteration, where n is the length of the longest string in the sample.

In the estimation process, the VS algorithm considers only the derivation with largest probability. Only the structural information contained in the best derivations of the training sample is taken into account. The rules which do not appear in one of the best derivations take null probability and are not considered for the next iteration. The probability mass is concentrated in the best derivations in very few iterations and therefore the algorithm converges rapidly [10].

The IO algorithm proceeds essentially in a similar way to the VS algorithm, but the IO algorithm takes into account all possible derivations in the learning process. If a rule appears in at least one non-null derivation, it never takes null probability (except in practical cases due to underflow problems). As in the VS algorithm, the probability mass tends to be concentrated in very few derivations but is slower than in the VS algorithm.

This behaviour suggests that the most probable derivations greatly determine the evolution of the IO algorithm. Under this hypothesis, we propose an algorithm which only considers the k derivations with the largest probability. Ignoring the information contained in the rest of the derivations could lead to a quicker convergence, in a way similar to the VS algorithm. However, the algorithm we propose can improve the estimated models obtained by the VS algorithm.

3. ESTIMATION FROM A SET OF DERIVATIONS

Given an initial SCFG G_s, a training sample Ω and a finite set [1] $\Delta_\Omega = \bigcup_{x \in \Omega} \Delta_x$, the following function can be used to modify the probabilities $(\forall (A \to \alpha) \in P)$:

$$q'(A \to \alpha) = \frac{\sum_{x \in \Omega} \frac{1}{\Pr(x, \Delta_x|G_s)} \sum_{d_x \in \Delta_x} \mathrm{N}(A \to \alpha, d_x) \Pr(x, d_x \mid G_s)}{\sum_{x \in \Omega} \frac{1}{\Pr(x, \Delta_x|G_s)} \sum_{d_x \in \Delta_x} \mathrm{N}(A, d_x) \Pr(x, d_x \mid G_s)} \quad (3)$$

This transformation attempts to improve the function $\Pr(\Omega, \Delta_\Omega \mid G_s) = \prod_{x \in \Omega} \Pr(x, \Delta_x \mid G_s)$ guaranteeing that $\Pr(\Omega, \Delta_\Omega \mid G_s^{'}) \geq \Pr(\Omega, \Delta_\Omega \mid G_s)$. It can be proven that this transformation guarantees that the estimated models are consistent [9].

It is important to note in this expression that the function being maximized changes depending on the number of derivations which are used. It can be observed that transformation (3) becomes (1) when Δ_x has the maximum number of derivations of x, while (3) becomes (2) when Δ_x has only the best derivation over all possible derivations. It is important to point out that no relation can be established between the function being maximized when a different number of derivations is used in the transformation.

Based on transformation (3), an algorithm is proposed in which the k-best derivations is computed in each iteration following the strategy of the VS algorithm. This algorithm starts with an initial grammar and then an iterative process begins. In each iteration, the k-best derivations of each string in the sample are obtained. The counts of the numerator and denominator of (3) are accumulated for those rules appearing in these derivations. At the end of the iteration, transformation (3) is applied for each rule. This iterative process continues until a local maximum of the function being optimized is achieved.

There exists an efficient algorithm to compute the k-best derivations of a string based on a Dynamic Programming scheme [6]. The time complexity to obtain the best derivation of a string x is $O(|x|^3 |P|)$. The time complexity to obtain each new derivation is in practice *approximately* proportional to the number of rules of the previous derivation times a logarithmic factor [6].

To study the time complexity of the proposed estimation algorithm, we suppose that the SCFG is in Chomsky Normal Form.

[1] We abuse the notation and consider this union operation to be a union between multisets which maintains the result as a multiset.

The set of k-best derivations for small values of k is calculated for each string x in the sample with a time complexity of *approximately* $O(|x|^3|P|)$ [6]. Therefore the time complexity of the algorithm per iteration is $O(|\Omega|n^3|P|)$, where n is the size of the longest string in the sample.

This estimation algorithm coincides with the IO algorithm when the number of derivations is the maximum for every string, and it coincides with the VS algorithm when the number of derivations is the best one of all the derivations.

4. EXPERIMENTAL WORK

We empirically show the behaviour of the proposed algorithm, for both its convergence as well as the goodness of the estimated models. Firstly, we tried to compare the proposed algorithm with the IO and VS algorithms. Given the characteristics of the IO algorithm, a synthetic experiment was carried out. Next, results with a pseudo-natural task are presented. Finally, preliminary results with a real task are also reported.

For the synthetic experiment, the arithmetic expressions language with 5 terminal symbols was used. A training corpus of 5,000 strings was used. Moreover, an initial grammar with the maximum number of rules which can be created with 5 terminal symbols and 11 non-terminal symbols was constructed. The initial probabilities were randomly generated with severals seeds. This initial grammar was estimated using the IO algorithm, the VS algorithm and the proposed algorithm for different values of k. After each iteration, the log of the function being maximized by each one was computed and the results can be seen in Figure 1.

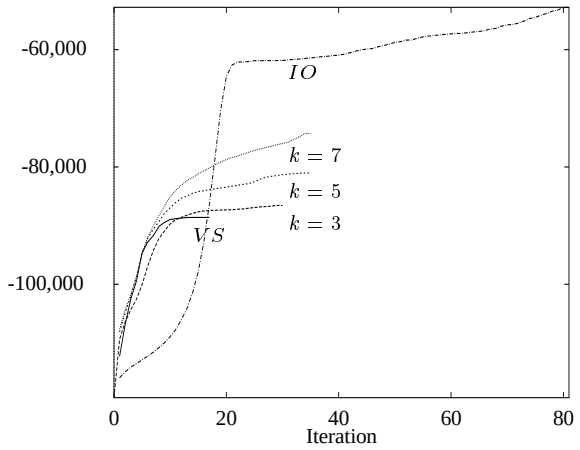

Figure 1: Evolution of the log of the function that is being maximized by each algorithm (VS, $k = 3,\ 5,\ 7$ and IO).

An important aspect to note in this figure is the number of iterations which were needed to converge depending on the number of derivations used. The IO algorithm took more iterations than the other ones. When the VS algorithm was used ($k = 1$), the convergence was very rapid because it only took into account the best derivation and a lot of information was ignored from the first iteration. When learning from the k-best derivations, the behaviour was an intermediate behaviour between the IO and VS algorithms. More information was used than in the VS algorithm and, therefore, the convergence took more iterations. However much less information was used than in the IO algorithm and, therefore, the convergence was quicker. All of the algorithms proceeded in a similar way, concentrating the probability mass of each string in a small set of derivations (see Table 1).

algorithm	iteration 0	iteration 20	convergence
VS	0.0000045 (0.5%)	0.0279 (100%)	0.0279 (100%)
$k = 3$		0.1509 (100%)	0.1710 (100%)
$k = 5$		0.1759 (100%)	0.2732 (100%)
$k = 7$		0.1870 (100%)	0.2016 (100%)
IO		0.1328 (24.0%)	0.4792 (59.8%)

Table 1: Accumulated probability mass in the sample without considering repeated strings with the initial grammar, after iteration 20 and after convergence. In parenthesis, the percentage of accumulated probability mass in the ten best derivations.

Another important aspect to note is the maxima achieved, or in other words, the goodness of the obtained model. Every algorithm improves its own function iteratively, and therefore, the likelihood of the sample was computed to be able to compare the obtained models. This is the function that is maximized by the IO algorithm and it is very interesting because it is directly related to the *perplexity* of a sample. The results can be seen in Table 2.

VS	-85,400.16		65.35%
$k = 3$	-84,939.18	0.54%	64.48%
$k = 5$	-79,974.33	6.35%	54.84%
$k = 7$	-73,854.73	13.52%	43.00%
IO	-51,648.20	39.52%	

Table 2: The second column represents the log of the likelihood of the sample. The third column represents the improvement of the log of the likelihood with respect to the VS algorithm. The fourth column represents the loss of the log of the likelihood with respect to the IO algorithm.

We can observe that the likelihood improved in absolute terms as we used more derivations in the estimation process. The improvement of the likelihood in relative terms between the VS algorithm and the IO algorithm was about 39.52% and 13.52% was obtained using only seven derivations. The likelihood decreased approximately 43% if seven derivations were used, but it decreased approximately 65% if only one derivations is used. It is important to observe that we used small values of k, and larger values could lead to better results.

In the second experiment, the proposed algorithm has been proven with an extension of a pseudo-natural task proposed in [4]. The original task consisted of descriptions of simple two-dimensional visual scenes involving a few geometric objects with different shapes, shades and sizes, and located in different relative positions. The size of the vocabulary was 19 terminal symbols and they were grouped into 9 grammatical categories. This

modification maintains most of the structural information contained in the sample.

In the experiment, 20,000 strings were used for training (2,730 different strings) and 10,000 for testing (1,723 different strings)[2]. An initial grammar with with the maximum number of rules which can be created with 9 terminal symbols and 19 non-terminal symbols was constructed and the initial probabilities of the rules were randomly generated. This initial grammar was estimated using the proposed algorithm with different number of derivations and the final model was tested using the test set. The results obtained can be seen in Table 3. We can observe that the log of the likelihood tended to increase as more derivations were used in the training process, as happened in the previous task.

algorithm	log ts. lh.	% impr.	PP.
VS	-221499.3		4.22
$k = 3$	-210422.7	5.00%	3.93
$k = 5$	-206610.2	6.72%	3.83
$k = 7$	-203826.4	7.98%	3.76

Table 3: Results of the final models for the test set. The first column corresponds to the number of derivations used for training. The second column (log ts. lh.) corresponds to the log of the likelihood of the test set. The third column corresponds to the percentage of improvement with respect to the first row. The fourth column presents the perplexity of the test set per word.

Finally, the proposed algorithm has been also proven with a part of the Wall Street Journal task processed in the Penn Treebank project [5]. The tagged part of the corpus was selected for the experiment. The text was divided into sentences. A sentence was considered a sequence of terminal symbols which finished when the terminal symbol "." was found, when a line of =’s was found or when the end of the file was found. In order to reduce the computational effort, only those sentences with length less than the average length (24 terminal symbols) were considered for training and test. Sections 00-19 (24,124 sentences) were used for training and sections 20-24 were considered for test (6,444 sentences).

An initial grammar with the maximum number of rules which can be created with 45 terminal symbols (the number of tags defined in [5] [3]) and 14 non-terminal symbols (the number of symbols used to parse the corpus [5]) was constructed. The initial probabilities were randomly generated. This initial grammar was estimated using the proposed algorithm with different number of derivations and the final model was tested using the test set.

The perplexity of the test set for the VS algorithm was 24.22 and this value was 22.73 for the proposed algorithm with $k = 3$, which means an improvement of 6.15%. More experiments are being carrying out for other values of k.

[2] A slightly modified version of the grammar proposed in [4] was used to generate these strings. The modified grammar generates an infinite language.

[3] There are 48 tags defined in [5], but three of them did not appear in the corpus.

5. CONCLUSIONS

An iterative algorithm for estimating the probabilities of a Stochastic Context-Free Grammar using the k-best derivations has been proposed. The time complexity per iteration of the algorithm is practically the same as that of the VS and IO algorithms. In general, the models obtained by this algorithm improve the models obtained by the VS algorithm for small values of k.

For future research, we propose the application of this method to estimate SCFGs in other real tasks.

6. ACKNOWLEDGEMENTS

Work partially supported by the Spanish CICYT under contract TIC95/0884-C04. The authors wish to thank Dr. Andrés Marzal for providing the software to compute the k-best derivations of a string from a SCFG.

7. REFERENCES

1. L.E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3:1–8, 1972.
2. F. Casacuberta. Growth transformations for probabilistic functions of stochastic grammars. *IJPRAI*, 10(3):183–201, 1996.
3. S.F. Chen. *Bayesian Grammar Induction for Language Modeling*. Ph. d. dissertation, Harvard University, 1996.
4. J.A. Feldman, G. Lakoff, A. Stolcke, and S.H. Weber. Miniature language acquisition: A touchstone for cognitive science. Technical Report TR-90-009, International Computer Science Institute, Berkeley, CA, USA, 1990.
5. M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.
6. A. Marzal. *Cálculo de las k mejores soluciones a problemas de programación dinámica*. Ph. d. dissertation, Universidad Politécnica de Valencia, 1994.
7. H. Ney. Stochastic grammars and pattern recognition. In P. Laface and R. De Mori, editors, *Speech Recognition and Understanding. Recent Advances*, pages 319–344. Springer-Verlag, 1992.
8. F. Pereira and Y. Schabes. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 128–135. University of Delaware, 1992.
9. A. Sánchez and J.M. Benedí. Consistency of stochastic context-free grammmars from probabilistic estimation based on growth transformation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(9):1052–1055, 1997.
10. J.A. Sánchez, J.M. Benedí, and F. Casacuberta. Comparison between the inside-outside algorithm and the viterbi algorithm for stochastic context-free grammars. In P. Perner, P. Wang, and A. Rosenfeld, editors, *Advances in Structural and Syntactical Pattern Recognition*, pages 50–59. Springer-Verlag, 1996.