Accurate computation of the relative entropy between stochastic regular grammars

Rafael C. Carrasco
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, E-03071 Alicante
E-mail: carrasco@dlsi.ua.es

Running head: Relative entropy between grammars.
Keywords: stochastic languages, relative entropy, grammatical inference.

Abstract

Works dealing with grammatical inference of stochastic grammars often evaluate the relative entropy between the model and the true grammar by means of large test sets generated with the true distribution. In this paper, an iterative procedure to compute the relative entropy between two stochastic deterministic regular grammars is proposed.

Resumé

Les travails sur l'inférence de grammaires stochastiques évaluent l'entropie relative entre le modèle et la vraie grammaire en utilisant grands ensembles de test générés avec la distribution correcte. Dans cet article, on propose une procédure itérative pour calculer l'entropie relative entre deux grammaires.

1 Introduction

Stochastic models have been widely used in computer science, especially in those tasks dealing with noisy data or random sources such as pattern recognition, natural language modeling, etc. A stochastic model predicts a probability distribution for the events in the class under consideration and one of the most popular measures of the success in the prediction is the so-called relative entropy or Kullback-Leibler distance (see, for instance, ref. [2]). On the other hand, a number of algorithms [6, 4, 1] have been proposed within the grammatical inference approach that identify stochastic regular grammars from examples. Regular grammars define a rather small subset of languages, in particular those whose that can be processed and recognized by finite-state automata. However, they present the important advantage that the identification problem is well defined and some algorithms, as the one by Carrasco & Oncina [1] have been proved to converge in the limit to the correct grammar. Usually, instead of the relative entropy between the known grammar and the proposed model, the relative entropy between a large test set (a collection of examples generated with the target grammar) and the hypothesis is evaluated to check the different techniques. However, an accurate estimation requires huge test sets to be generated, and sometimes convergence can be very slow. Therefore, an algorithm providing the exact distance without generating large test sets is of interest.

2 Preliminaries

Let $\mathcal{A} = \{a, b, ...\}$ be a finite alphabet, $\mathcal{A}^*$ the set of strings generated by $\mathcal{A}$ and λ the empty string. For every string $x \in \mathcal{A}^*$, the expression $x\mathcal{A}^*$ denotes the set of strings that contain x as a prefix. A *stochastic language* L is defined by a probability density function $p(x|L)$ for the strings $x \in \mathcal{A}^*$. The probability of any subset $X \subset \mathcal{A}^*$ is

$$p(X|L) = \sum_{x \in X} p(x|L). \tag{1}$$

A *stochastic regular grammar* (SRG), $G = (\mathcal{A}, V, S, R, p_G)$, consists of a finite alphabet $\mathcal{A}$, a finite set of variables V —one of which, S, is referred to as the starting symbol—, a finite set of derivation rules R with either of the following structures

$$\begin{array}{rcl} X & \to & aY \\ X & \to & \lambda \end{array} \tag{2}$$

where $a \in \mathcal{A}$, $X, Y \in V$, and a real function $p_G : R \rightarrow [0,1]$ giving the probability of each derivation. Obviously, the sum of the probabilities for all derivations from a given variable X must be equal to one. The definition (2), although formally different, is equivalent to other definitions used in the literature, as the one in [3]. A stochastic grammar G is said to be *deterministic* if for all $X \in V$ and for all $a \in \mathcal{A}$ there is at most one $Y \in V$ such that $p_G(X \rightarrow aY) \neq 0$.

Every stochastic deterministic regular grammar G defines a *stochastic deterministic regular language* (SDRL) through the probabilities $p(w|G) = p_G(S \Rightarrow w)$. The probability $p_G(S \Rightarrow w)$ that the grammar G generates the string $w \in \mathcal{A}^*$ is defined in a recursive way:

$$\begin{array}{rcl} p_G(X \Rightarrow \lambda) & = & p_G(X \rightarrow \lambda) \\ p_G(X \Rightarrow aw) & = & p_G(X \rightarrow aY) p_G(Y \Rightarrow w) \end{array} \tag{3}$$

where Y is the only variable satisfying $p_G(X \rightarrow aY) \neq 0$. Provided that the SDRG contains no *useless symbols* [5], the probabilities of all strings sum up to 1:

$$p(\mathcal{A}^*|G) = \sum_{w \in \mathcal{A}^*} p(w|G) = 1 \tag{4}$$

A *stochastic deterministic finite automaton* (SDFA), $A = (Q, \mathcal{A}, \delta, q_I, p_A)$, consists of an alphabet $\mathcal{A}$, a finite set of nodes $Q = \{q_1, q_2, \dots q_n\}$, with $q_I \in Q$ the initial node, a transition function $\delta : Q \times \mathcal{A} \rightarrow Q$ and a probability function $p_A : Q \times \mathcal{A} \rightarrow [0,1]$. The probability $p_A(q_i, \lambda)$, defined for every node q_i as

$$p_A(q_i, \lambda) = 1 - \sum_{a \in \mathcal{A}} p_A(q_i, a)\,, \tag{5}$$

represents the probability that the string ends at q_i. Every SDFA generates a SDRL through the probabilities $p(w|A) = \pi(q_I, w)$, defined recursively as

$$\begin{array}{rcl} \pi(q_i, \lambda) & = & p_A(q_i, \lambda) \\ \pi(q_i, aw) & = & p_A(q_i, a) \pi(\delta(q_i, a), w) \end{array} \tag{6}$$

The comparison of equations (3) and (6) directly suggests the way of building a SDFA that generates the same language as a given grammar G. Indeed, it suffices to take $Q = V$, $q_I = S$ and for all $a \in \mathcal{A}$ and $X, Y \in V$

$$\begin{array}{rcl} \delta(X, a) = Y & \text{iff} & X \Rightarrow aY \in R \\ p_A(X, a) & = & p_G(X \Rightarrow aY) \end{array} . \tag{7}$$

3 Entropy of a SRDL

The entropy of the stochastic language L is defined [2] as

$$H(L) = -\sum_{x \in \mathcal{A}^*} p(x|L) \log p(x|L) \tag{8}$$

with the convention $0 \log 0 = 0$. When the logarithm is binary, the result is expressed in bits. The entropy is always a positive number related to the average length of the strings in a minimal coding of the language and to the average number of yes/no questions (with an optimal interrogation strategy) necessary in order to identify the result of a random extraction of a word in L.

Two stochastic languages having the same entropy are not necessarily identical. However, the magnitude

$$H(L_1, L_2) = \sum_{x \in \mathcal{A}^*} p(x|L_1) \log \frac{p(x|L_1)}{p(x|L_2)} \tag{9}$$

has the property that $H(L_1, L_2) = 0$ if and only if $p(x|L_1) = p(x|L_2)$ for all strings x in $\mathcal{A}^*$. This magnitude is known as *relative entropy* or *Kullback-Leibler distance*, although it is not a true distance (even if it can be easily symmeterized, it does not satisfy the triangular inequality). The relative entropy indicates the penalty (in bits) for using a wrong distribution instead of the true one when coding a word or when predicting the result of a random experiment.

In the following, $p_L(a|x)$ will denote the conditioned probability that symbol a follows the prefix x in L:

$$p_L(a|x) = \frac{p(xa\mathcal{A}^*|L)}{p(x\mathcal{A}^*|L)} \tag{10}$$

Consistently with eq. (5), we will denote with $p_L(\lambda|x)$ the probability that the "end of string" is observed after the prefix x, i.e., that x is not followed by any other symbol. In other words,

$$p_L(\lambda|x) = \frac{p(x|L)}{p(x\mathcal{A}^*|L)} \tag{11}$$

With these conventions, for instance, the probability $p(ab|L)$ for the string ab in the language L satisfies:

$$\log p(ab|L) = \log p_L(a|\lambda) + \log p_L(b|a) + \log p_L(\lambda|ab) \tag{12}$$

Thus, when evaluating the entropy as defined in eq. (8), the term $\log p_L(b|a)$ will appear for every string containing ab as a prefix. In general, a factor $\log p_L(a|x)$, will appear for every string in the subset $xa\mathcal{A}^*$, while the factor $\log p_L(\lambda|x)$ will only multiply $p(x|L)$. Therefore,

$$H(L) = -\sum_{x\in\mathcal{A}^*}\sum_{a\in\mathcal{A}} p(xa\mathcal{A}^*|L)\log p_L(a|x) - \sum_{x\in\mathcal{A}^*} p(x|L)\log p_L(\lambda|x). \quad (13)$$

By using eqs. (10) and (11), one can rewrite the above equation in a simpler form:

$$H(L) = -\sum_{x\in\mathcal{A}^*}\sum_{a\in\mathcal{A}'} p(x\mathcal{A}^*|L)p_L(a|x)\log p_L(a|x) \quad (14)$$

where $\mathcal{A}' = \mathcal{A}\cup\{\lambda\}$.

If L is generated by a SRG, there is an associated automaton $A = (Q, \mathcal{A}, \delta, q_i, p_A)$ generating L, and $p_L(a|x)$ may take only a finite number of different values. Indeed, for all x satisfying $\delta(q_I, x) = q_i$ and for all $a \in \mathcal{A}'$ one gets $p_L(a|x) = p_A(q_i, a)$. The different subsets $L_i = \{x \in \mathcal{A}^* : \delta(q_I, x) = q_i\}$ define a partition in L and, if one defines

$$c_i = \sum_{x\in L_i} p(x\mathcal{A}^*|L), \quad (15)$$

the entropy becomes

$$H(L) = -\sum_{q_i\in Q}\sum_{a\in\mathcal{A}'} c_i\, p_A(q_i, a)\log p_A(q_i, a) \quad (16)$$

This sum can be computed straightforwardly once the coefficients c_i are known.

Note that always $\lambda \in L_I$ (being I the index of the initial state) and recall that $p(\mathcal{A}^*|L) = 1$. This allows us to deal separately with the special case $x = \lambda$ and write

$$c_i = \delta_{iI} + \sum_{x\in\mathcal{A}^*}\sum_{\substack{a\in\mathcal{A}\\ xa\in L_i}} p_A(xa\mathcal{A}^*|L) \quad (17)$$

where δ_{ij} is Kronecker's delta. As $L = \bigcup_j L_j$ and for every $x \in L_j$, $p(xa\mathcal{A}^*|L) = p(x\mathcal{A}^*|L)p_A(q_j, a)$, one gets

$$c_i = \delta_{iI} + \sum_{j=1}^{|Q|}\sum_{x\in L_j}\sum_{\substack{a\in\mathcal{A}\\ \delta(q_j,a)=q_i}} p(x\mathcal{A}^*|L)p(q_j, a) \quad (18)$$

Therefore, the coefficients c_i can be obtained through an iterative method:

$$c_i^{[t+1]} = \sum_{j=1}^{|Q|} A_{ij} c_j^{[t]} + \delta_{iI} \tag{19}$$

with

$$A_{ij} = \sum_{\substack{a \in \mathcal{A} \\ \delta(q_j, a) = q_i}} p_A(q_j, a). \tag{20}$$

and $c_i^{[0]} = 0$. It is easy to prove by induction in t that $c_i^{[t+1]} \geq c_i^{[t]}$ but $c_i^{[t]} \leq c_i$, and therefore, the iterative calculation converges rapidly to the correct value.

4 The relative entropy between SDRL

An analogous procedure can be applied to the relative entropy between two stochastic regular languages L and L', generated by M and M' respectively. In this case,

$$H(L, L') = \sum_{q_i \in Q} \sum_{q'_j \in Q'} \sum_{a \in \mathcal{A}'} c_{ij} \, p_M(q_i, a) \log \frac{p_M(q_i, a)}{p_{M'}(q'_j, a)} \tag{21}$$

with the coefficients

$$c_{ij} = \sum_{x \in L_{ij}} p(x\mathcal{A}^* | L), \tag{22}$$

where

$$L_{ij} = \{x \in \mathcal{A}^* : \delta(q_I, x) = q_i \wedge \delta'(q'_{I'}, x) = q'_j\}. \tag{23}$$

The coefficients c_{ij} are evaluated through the relation:

$$c_{ij}^{[t+1]} = \delta_{iI} \delta_{I'j} + \sum_{k=1}^{|Q|} \sum_{l=1}^{|Q'|} A_{ijkl} \, c_{kl}^{[t]} \tag{24}$$

where

$$A_{ijkl} = \sum_{\substack{a \in \mathcal{A} \\ \delta(q_k, a) = q_i \\ \delta'(q'_l, a) = q'_j}} p_M(q_k, a). \tag{25}$$

The above expression for $c_ij^{[t+1]}$ is straightforward to prove following the same steps of former section and noting that $\lambda \in L_{II'}$.

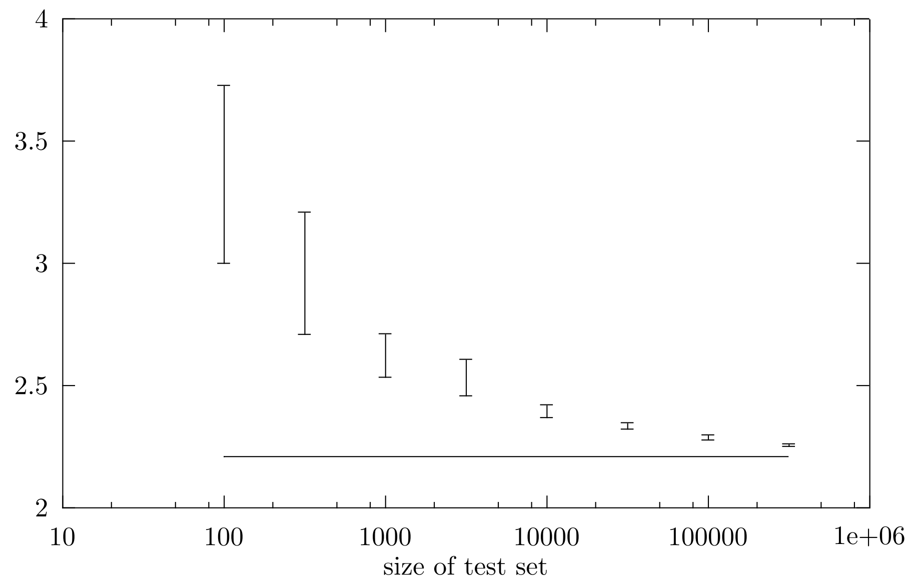

Figure 1: Relative entropy (in bits) between two randomly generated grammars of size 10. Solid line: exact computation. Dots: estimation using samples.

5 Results and Conclusion

The Figure 1 shows the relative entropy between two randomly generated grammars, each with 10 variables and 30 rules, both working with the alphabet $\mathcal{A} = \{0, 1\}$. The solid line is the result of the algorithm, while the dots represent the results and deviations of the relative entropy with random test sets of increasing size. It can be seen that even for relatively simple grammars as these, the convergence to the true value is rather slow, and huge samples are needed in order to get a good estimate of the relative entropy between the languages. Therefore, the procedure described in this paper can be used for a more accurate testing of grammatical inference methods.

Acknowledgements

This work was partially supported by the Spanish CICYT under grant TIC-97-0941. I also thank Dr. Mikel Forcada for his suggestions.

References

[1] Carrasco, R.C. and Oncina, J. (1994): Learning stochastic regular grammars by means of a state merging method, in *Grammatical Inference and Applications* (R.C. Carrasco and J. Oncina, Eds.). Lecture Notes in Artificial Intelligence **862**, Springer-Verlag, Berlin.

[2] Cover, T.M and Thomas, J.A. (1991): *Elements of Information Theory.* John Wiley and Sons, New York.

[3] Fu, K.S. (1982): *Syntactic Pattern Recognition and Applications.* Prentice Hall, Englewood Cliffs, New Jersey.

[4] Stolcke A. and Omohundro, S. (1993): Hidden Markov Model Induction by Bayesian Model Merging, in *Advances in Neural Information Processing Systems 5* (C.L. Giles, S.J. Hanson and J.D. Cowan Eds.), Morgan Kaufman, Menlo Park, California.

[5] Hopcroft, J.E. and Ullman, J.D. (1979): *Introduction to automata theory, languages and computation.* Addison Wesley, Reading, Massachusetts.

[6] van der Mude, A. and Walker, A. (1978): On the Inference of Stochastic Regular Grammars. *Information and Control* **38**, 310–329.