Machine Learning, 44, 121–141, 2001
© 2001 Kluwer Academic Publishers. Manufactured in The Netherlands.

Some Statistical-Estimation Methods for Stochastic Finite-State Transducers*

DAVID PICÓ AND FRANCISCO CASACUBERTA {dpico,fcn}@iti.upv.es
Institut Tecnològic d'Informàtica, Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, Valencia, Spain

Editors: Vasant Honavar and Colin de la Higuera

Abstract. Formal translations constitute a suitable framework for dealing with many problems in pattern recognition and computational linguistics. The application of formal transducers to these areas requires a stochastic extension for dealing with noisy, distorted patterns with high variability. In this paper, some estimation criteria are proposed and developed for the parameter estimation of regular syntax-directed translation schemata. These criteria are: maximum likelihood estimation, minimum conditional entropy estimation and conditional maximum likelihood estimation. The last two criteria were proposed in order to deal with situations when training data is sparse. These criteria take into account the possibility of ambiguity in the translations: i.e., there can be different output strings for a single input string. In this case, the final goal of the stochastic framework is to find the highest probability translation of a given input string. These criteria were tested on a translation task which has a high degree of ambiguity.

Keywords: stochastic finite-state transducers, probabilistic estimation

1. Introduction

A *translation* is a process that maps strings from a given language (input language) to strings from another language (output language). The formal devices that implement translations are known as *formal transducers* and have been thoroughly studied in the theory of formal languages (Berstel, 1979). Initially, formal translations were proposed for compiling computer programs (Aho & Ullman, 1972) and as a formal framework for the presentation of error-correction models in syntactic pattern recognition (González & Thomason, 1978).

Regular translations constitute an important class of formal translations. The solid formalism on which the theory of transducers and regular translations is built (Berstel, 1979) has allowed for a deep and rigorous study of these models. Even though regular translations are much more limited than other more general translation models, the computational costs of the algorithms that are needed to handle them are much lower.

Regular translation has recently become of great interest as a model in some practical pattern recognition problems in which the classification paradigm is not adequate, since the number of classes necessary could be large or even infinite (Vidal, Casacuberta &

*This work has been partially supported by the Spanish CICYT under contract EXTRA (TIC97/0745-C02) and by project EUTRANS (ESPRIT LTR-30268).

García, 1995). In this case, the most general paradigm of *interpretation* seems to be a better framework which can be tackled through formal translations. For example, many tasks in automatic speech recognition can be viewed as simple translations from acoustic sequences to sub-lexical or lexical sequences (acoustic-phonetic decoding), or from acoustic or lexical sequences to sequences of commands to a data-base management system or to a robot (semantic decoding). A more complex application is the translation between natural languages (e.g., English to Spanish) (Amengual et al., 1998; Vidal, 1997). Formal transducers are also used in computational linguistics for representing linguistic knowledge such as the phonetics and morphosyntactics of natural languages (Gildea & Jurafsky, 1996; Mohri, 1997; Oflazer, 1996; Roche & Schabes, 1995). On the other hand, formal transducers can be learned automatically from examples (Oncina, García &Vidal, 1993; Casacuberta, 2000). This opens a wide field of applications based on the induction of translation models from parallel corpora.

However, the application of formal transducers to syntactic pattern recognition (and also sometimes to natural language processing) needs a stochastic extension due to the noisy and distorted patterns which make the process of interpretation ambiguous (Fu, 1982). The statistical parameters of the extended models define a probability distribution over the possible translations that help determine the best translation of a given input sentence. Given a formal transducer, a common way of setting these parameters is to learn them from examples of translations.

Stochastic regular grammars are related to stochastic regular syntax-directed translation schemata (Maryanski & Thomason, 1979) and to some classes of widely-used hidden Markov models (Casacuberta, 1900). A stochastic regular syntax-directed translation schema can be viewed as a stochastic regular grammar (or as some type of hidden Markov model) together with an output function that relates transitions (or states) to the sets of possible output strings. From this point of view, stochastic regular syntax-directed translation schemata represent formalisms that are more adequate for dealing with translation than stochastic regular grammars (or hidden Markov models).

A maximum-likelihood algorithm for automatically learning the statistical parameters of stochastic regular syntax-directed translation schemata from examples has recently been proposed (Casacuberta, 1995, 1996). This algorithm estimates the parameter set by maximizing the likelihood of the training data over the model. Only the translations of a given input string that explicitely appear in the training corpus are taken into account during the estimation process, and, although the process is guaranteed to converge to the optimal model for large enough data sets, its performance is poor for limited or sparse data.

The minimum conditional entropy estimation criterion, which is presented in Section 3.2, is based on some ideas taken from maximum mutual information (Brown, 1987; Cardin, Normandin & DeMori, 1994). The conditional maximum likelihood approach was originally proposed for speech modelling in Nádas, Hahamoo and Picherny (1988). These two criteria take into consideration not only the sentences that explicitly appear in the training corpus but also take into consideration all the possible translations of these sentences that the model can produce. These two criteria can be particularly useful when the training data is sparse.

The application of these approaches in stochastic regular syntax-directed translation schemata was first proposed in Casacuberta (1996). Here, a different learning algorithm based on the maximum mutual information criterium was proposed. We have found the original proposal to be inadequate for regular translation models. A discussion about this matter is given in Section 3.2.3. Furthermore, the models and algorithms presented in this paper are more general (empty rules are allowed) than those in Casacuberta (1996). New experiments are reported here.

The rest of this paper is structured as follows. Section 2 presents the basic concepts of finite-state transducers and the main parsing algorithms. Four different methods of probabilistic estimation are developed in Section 3 and some experiments are reported in Section 4. Finally, some conclusions are given in Section 5.

2. Formal translations

A *formal translation* T is a subset of $\Sigma^\star \times \Delta^\star$, where Σ is an *input alphabet*, and Δ is an *output alphabet*. Note that naming Σ and Δ as the *input* and *output* alphabets is an arbitrary decision. We will use the terms input and output whenever it helps to make the presentation clearer.

A translation T is *left-to-right ambiguous* if there exists some $f \in \Sigma^\star$ such that the set $\{(f, g) \mid (f, g) \in T\}$ has more than one element. Similarly, a translation T is *right-to-left ambiguous* if there exists some $g \in \Delta^\star$ such that the set $\{(f, g) \mid (f, g) \in T\}$ has more than one element.

Formal translations can be characterized in a manner which is similar to formal languages. In fact, there are close relations between some types of translations and some types of languages (Aho & Ullman, 1972). (Relations of this type were also extended to stochastic translations and stochastic languages (Maryanski & Thomason, 1979)).

One important class of translations is constituted by *regular* or *rational translations*. As for formal languages, there are different models for dealing with translations of this class. One of them is the *rational transducer* (Berstel, 1979). Another one is the *regular syntax-directed translation schema* (Thomason, 1976). We have chosen the latter because it is similar to regular grammars, and there are developments for regular grammars and their corresponding stochastic extension which can be applied to translations, once they have been conveniently adapted.

2.1. Definitions

A *stochastic regular syntax-directed translation schema* (SRT) is a system $T = (N, \Sigma, \Delta, R, S, P)$, in which N is a finite set of non-terminal symbols, Σ and Δ are finite sets of input and output terminal symbols, respectively, and $S \in N$ is the initial symbol of the schema. If we represent the empty string as λ, then R is a set of rules $A \to aB, zB$ or $A \to a, z$, where $A, B \in N$, $a \in \Sigma \cup \{\lambda\}$ and $z \in \Delta^\star$ (by $\Sigma^\star$ and $\Delta^\star$ we will denote the sets of finite-length strings on Σ and Δ, respectively). Rules of the form $A \to B, B$ are called λ-rules. Although this is the definition of an SRT for the general case, in the remainder of the paper we will only consider SRTs where λ-rules do not form a loop.

The stochastic component of the schema is given by $P : R \to]0, 1]$, a probability assignment to the rules such that the sum of probabilities of all rules rewriting a non-terminal A

is equal to 1 (*proper* SRT (González & Thomason, 1978). Formally, for any non-terminal A_i, the set $\{A_i \to \omega_1,\ A_i \to \omega_2, \ldots,\ A_i \to \omega_n\}$ of all rules that rewrite A_i must satisfy the following condition of stochasticity:

$$\sum_{j=1}^{n} P(A_i \to \omega_j) = 1. \tag{1}$$

This probability assignment, P, defines a set of probabilistic parameters that will be denoted by $\Phi(T)$. On the other hand, for the sake of simplicity, in the remainder of the paper, we will denote $Pr(X = x)$ as $Pr(x)$ and $Pr(Y = y \mid X = x)$ as $Pr(y \mid x)$ where X and Y are stochastic variables and x and y are two possible values of X and Y, respectively.

A *translation form* that yields the *translation pair* $(x, y) \in \Sigma^\star \times \Delta^\star$ is a finite sequence of rules $tf = (r_1, r_2, \ldots, r_n)$ such that

$$(S, S) \overset{r_1}{\Rightarrow} (x_1 A_1, y_1 A_1) \overset{r_2}{\Rightarrow} (x_1 x_2 A_2, y_1 y_2 A_2) \cdots \overset{r_n}{\Rightarrow} (x, y).$$

If the translation form tf rewrites (S, S) as (x, y), we will denote x as *input*(tf), and y as *output*(tf).

An SRT T associates a probability to each possible translation form $tf = (r_1, r_2, \ldots, r_n)$ which is the product of the probabilities of the rules in the form:

$$Pr(tf \mid \Phi(T)) = P(r_1)P(r_2)\cdots P(r_n). \tag{2}$$

An SRT is, in general, non-deterministic. This means that a translation $(x, y) \in \Sigma^\star \times \Delta^\star$ may then be produced by more than one translation form tf (so that *input*$(tf) = x$ and *output*$(tf) = y$.) When this possibility is considered, the *probability of a translation* (x, y) must be defined as the sum of probabilities of all the corresponding translation forms:

$$Pr(x, y \mid \Phi(T)) = \sum_{\substack{\forall tf / input(tf) = x \\ \wedge output(tf) = y}} Pr(tf \mid \Phi(T)). \tag{3}$$

When there is more than one possible translation form for some pair (x, y) we will speak of a *grammatically ambiguous SRT*. When for a given input x there is more than one output string y such that $Pr(x, y|\Phi(T)) > 0$ we will speak of an *ambiguous-in-translation* SRT (or simply an *ambiguous* SRT).

We are now ready to define how the translation of a given input sentence is to be calculated. Since the SRT can be ambiguous, a single input sentence may be translated into more than one output sentence. The decision of which one of the possible different outputs is considered to be the best will be made using a statistical criterion. The *stochastic translation* of an input string $x \in \Sigma^\star$ in a SRT T is the string $y^\star \in \Delta^\star$ into which x can be translated with the highest probability:

$$y^\star = \underset{y \in \Delta^\star}{\operatorname{argmax}}\, Pr(y \mid x, \Phi(T)), \tag{4}$$

where

$$Pr(y \mid x, \Phi(T)) = \frac{Pr(x, y \mid \Phi(T))}{Pr(x \mid \Phi(T))}.$$

Given that the probability $Pr(x \mid \Phi(T))$ does not depend upon the maximization index y, we can rewrite (4) as:

$$y^\star = \operatorname*{argmax}_{y \in \Delta^\star} Pr(x, y \mid \Phi(T)) \tag{5}$$

The search of the optimal $y^\star$ in (5) using definition (3) is a difficult computational problem (Casacuberta & de la Higuera, 2000). The only possible algorithmic solution is the use of some variant of the $A^\star$ algorithm (e.g., the Stack-Decoding (Bahl, Jelinek & Mercer, 1983)), which presents exponential computational costs in the worst case and, therefore, may not be feasible for use in some real applications.

A computationally cheaper approximation to the stochastic translation can be defined. Instead of defining the probability of a translation as shown in (3), we will work with the *Viterbi probability of a translation*. This is defined as the probability of the translation form that most probably yields (x, y):

$$\hat{Pr}(x, y \mid \Phi(T)) = \max_{\substack{\forall tf / input(tf)=x \\ \wedge output(tf)=y}} Pr(tf \mid \Phi(T)) \tag{6}$$

The *approximate stochastic translation* is an approximation to (5) in which the Viterbi probability is used instead of the standard probability defined in (3):

$$y^{\star\star} = \operatorname*{argmax}_{y \in \Delta^\star} \hat{Pr}(x, y \mid \Phi(T)). \tag{7}$$

There exists a polynomial algorithm for calculating (7). Given an input string x, this algorithm searches for the maximum probability translation form *tf* so that $input(tf) = x$. More formally, we will calculate the approximate stochastic translation as:

$$y^{\star\star} = output\left(\operatorname*{argmax}_{\forall tf / input(tf)=x} Pr(tf \mid \Phi(T)) \right). \tag{8}$$

The relationship between the exact result from (4) and the approximate result from (8) is similar to the analogous relationship found in stochastic grammars (Sánchez & Benedí, 1997) and in hidden Markov models (Mergav & Ephraim, 1991). Algorithms for calculating some of the expressions defined above will be presented in the following subsection. They will also be useful in the estimation methods that are discussed in Section 3.

2.2. Algorithms

From this point on, we will use a slightly modified version of the SRT for reasons of simplicity. Now, a new non-terminal $F \notin (N \cup \Sigma)$ is added. All rules of the form $A \to a, z \in R$ are rewritten as $A \rightarrow aF, zF$, for $a \in \Sigma \cup \{\lambda\}$ and $z \in \Delta^\star$, with the same probability value. The only rule that has the non-terminal F on the left side is $F \rightarrow \lambda, \lambda$. It can be demonstrated that translation schemata with or without this modification are equivalent.

The names of the algorithms will be labeled with super-indexes I or O, depending on whether they need an input string, an output string or both as an argument. We will use the following notational convention: given a string $x = x_1 \ldots x_{|x|}$ of length $|x|$, with $x_i \in \Sigma$ for $1 \leq i \leq |x|$, and two indexes i, j such that $i \leq j$, expression $x_{i \ldots j}$ will stand for the substring $x_i \ldots x_j$. If the indexes are crossed over (i.e., if we are writing $x_{i \ldots j}$ when $i > j$), it is understood that the expression is equivalent to the empty string λ.

2.2.1. Forward and backward algorithms. Let $T = (N, \Sigma, \Delta, R, S, P)$ be a SRT, $x \in \Sigma^\star$, $A \in N \cup \{F\}$, $0 \leq i \leq |x|$, $0 \leq j \leq |y|$. We define the following two functions:

$$\alpha^{I,O}(i, j, A) = Pr((S, S) \overset{\star}{\Rightarrow} (x_{1 \ldots i} A, y_{1 \ldots j} A) \mid \Phi(T)) \tag{9}$$

$$\beta^{I,O}(i, j, A) = Pr((A, A) \overset{\star}{\Rightarrow} (x_{i+1 \ldots |x|}, y_{i+1 \ldots |y|}) \mid \Phi(T)) \tag{10}$$

These two functions represent partial sums of probability which correspond to different substrings of the input and output strings. The probability of a translation (x, y) can be obtained from any of these two functions:

$$\alpha^{I,O}(|x|, |y|, F) = \beta^{I,O}(0, 0, S) = Pr(x, y \mid \Phi(T)).$$

The so-called *forward* and *backward* algorithms are two recursive algorithms that calculate α and β. These algorithms are described in a recursive manner by means of two auxiliary functions, $\alpha_\lambda^{I,O}$ and $\beta_\lambda^{I,O}$. Assuming that the value of $\alpha^{I,O}(i, j, A)$ is known $\forall A \in N, i \geq 0$, $j \geq 0$, we will define the auxiliary function $\alpha_\lambda^{I,O}$ as follows, for $0 \leq i \leq |x|$, $0 \leq j \leq |y|$ and $k > 0$:

$$\begin{aligned}
\alpha_\lambda^{I,O}(i, j, 0, A) &= \sum_{A' \in N} \Bigg(\sum_{j' \leq j} \alpha^{I,O}(i-1, j', A') \cdot P(A' \rightarrow x_i A, y_{j'+1 \cdots j} A) \\
&\qquad + \sum_{j' < j} \alpha^{I,O}(i, j', A') \cdot P(A' \rightarrow A, y_{j'+1 \cdots j} A) \Bigg) \\
\alpha_\lambda^{I,O}(i, j, k, A) &= \sum_{A' \in N} \alpha_\lambda^{I,O}(i, j, k-1, A') \cdot P(A' \rightarrow A, A).
\end{aligned} \tag{11}$$

More informally, $\alpha_\lambda^{I,O}(i, j, k, A)$ computes the probability of parsing the pair of substrings $(x_{1 \ldots i}, y_{1 \ldots j})$ and of then producing a derivation of k λ-rules which ends up writing the non-terminal A. The value of $\alpha^{I,O}(i, j, A)$ can therefore be calculated as an infinite sum over all the probabilities of all possible derivations, which may include any number of λ-rules:

$$\alpha^{I,O}(i, j, A) = \sum_{k=0}^{\infty} \alpha_\lambda^{I,O}(i, j, k, A) \tag{12}$$

The value of this expression can be calculated in a simpler way by just making it depend only on $\alpha_\lambda^{I,O}(i, j, 0, A)$. First of all, let us expand the second equation in (11):

$$\begin{aligned}\alpha_\lambda^{I,O}(i, j, k, A) = \sum_{A_1 \in N} \alpha_\lambda^{I,O}(i, j, 0, A_1) \cdot \sum_{A_2 \in N} P(A_1 \rightarrow A_2, A_2) \cdot \ldots \\ \ldots \cdot \sum_{A_k \in N} P(A_{k-1} \rightarrow A_k, A_k) \cdot P(A_k \rightarrow A, A)\end{aligned}$$

Now we can make substitutions in (12) and reorder sums:

$$\alpha^{I,O}(i, j, A) = \sum_{k=0}^{\infty} \alpha_\lambda^{I,O}(i, j, k, A) = \sum_{A' \in N} \alpha_\lambda^{I,O}(i, j, 0, A') \cdot \sum_{k=0}^{\infty} [\mathbf{P}^k]_{A',A} \tag{13}$$

The matrix $\mathbf{P}$ of λ-rules has dimensions $|N| \times |N|$. Each one of its elements is defined as $\mathbf{P}_{A,B} = P(A \rightarrow B, B)$, for $A, B \in N$. Each element of the infinite power series of $\mathbf{P}$, $\sum_{k=0}^{\infty} [\mathbf{P}^k]_{A,B}$, is equal to the sum of probabilities of all possible derivations with λ-rules that rewrite A into B. We represent the identity matrix as $\mathbf{I}$. If matrix $\mathbf{I} - \mathbf{P}$ can be inverted, then the value of the matrix power series is known (Jelinek & Lafferty, 1991), and we can use expression (14) for calculating the exact value of $\alpha^{I,O}(i, j, A)$. Otherwise, expression (13) can be used to find an approximate value by truncating the power series.

$$\alpha^{I,O}(i, j, A) = \sum_{A' \in N} \alpha_\lambda^{I,O}(i, j, 0, A') \cdot [(\mathbf{I} - \mathbf{P})^{-1}]_{A',A} \tag{14}$$

Finally, the base case in the recursion of $\alpha^{I,O}$ can be calculated as:

$$\alpha^{I,O}(i, j, A) = \sum_{A' \in N} \delta(A', S) \cdot [(\mathbf{I} - \mathbf{P})^{-1}]_{A',A},$$

where function δ means:

$$\delta(A, B) = \begin{cases} 0 & \text{if } A \neq B, \\ 1 & \text{if } A = B. \end{cases}$$

Similar reasonings yield the following analogous expressions for $\beta^{I,O}$ and $\beta_\lambda^{I,O}$:

$$\begin{aligned}\beta^{I,O}(|x|, |y|, A) &= \sum_{A' \in N} \delta(A', F) \cdot [(\mathbf{I} - \mathbf{P})^{-1}]_{A,A'} \\ \beta^{I,O}(i, j, A) &= \sum_{A' \in N} \beta_\lambda^{I,O}(i, j, 0, A') \cdot [(\mathbf{I} - \mathbf{P})^{-1}]_{A,A'} \\ \beta_\lambda^{I,O}(i, j, 0, A) &= \sum_{A' \in N} \Bigg(\sum_{j' \geq j} P(A \rightarrow x_{i+1}A', y_{j+1\cdots j'}A') \cdot \beta^{I,O}(i+1, j', A') \\ &\qquad + \sum_{j' > j} P(A \rightarrow A', y_{j+1\cdots j'}A') \cdot \beta^{I,O}(i, j', A') \Bigg)\end{aligned}$$

$$\beta_\lambda^{I,O}(i, j, k, A) = \sum_{A' \in N} \beta_\lambda^{I,O}(i, j, k-1, A') \cdot P(A \to A', A').$$

The time complexity of the $\alpha^{I,O}$ and $\beta^{I,O}$ algorithms is $O(|x| \cdot |y| \cdot |R|)$ if dynamic programming is used.

2.2.2. Viterbi algorithm. The Viterbi probability of a translation (x, y) can be calculated by defining a function $\gamma^{I,O}$ which is very similar to $\alpha^{I,O}$ but uses maximizations instead of sums, such that:

$$\gamma^{I,O}(|x|, |y|, F) = \hat{P}r(x, y \mid \Phi(T)).$$

Let us first define an auxiliary function $\gamma_\lambda^{I,O}$:

$$\begin{aligned}\gamma_\lambda^{I,O}(i, j, 0, A) &= \max_{A' \in N} \Big(\max_{j' \leq j} \gamma^{I,O}(i-1, j', A') \cdot P(A' \to x_i A, y_{j'+1\cdots j} A) \\ &\qquad + \max_{j' < j} \gamma^{I,O}(i, j', A') \cdot P(A' \to A, y_{j'+1\cdots j} A) \Big) \\ \gamma_\lambda^{I,O}(i, j, k, A) &= \max_{A' \in N} \gamma_\lambda^{I,O}(i, j, k-1, A') \cdot P(A' \to A, A), \end{aligned} \tag{15}$$

The limits of the indexes are $0 \leq i \leq |x|$, $0 \leq j \leq |y|$ and $k > 0$.

The expression $\gamma_\lambda^{I,O}(i, j, k, A)$ computes the Viterbi probability of parsing the pair of substrings $(x_{1\ldots i}, y_{1\ldots j})$ and of then producing a derivation of k λ-rules with ends up writing the non-terminal A. The value of $\gamma^{I,O}(i, j, A)$ can therefore be calculated as an infinite application of the operator maximization over all the probabilities of all possible derivations, which may include any number of λ-rules:

$$\gamma^{I,O}(i, j, A) = \max_{k=0}^{\infty} \gamma_\lambda^{I,O}(i, j, k, A) \tag{16}$$

The value of this expression can be calculated in a simpler way by just making it depend only on $\gamma_\lambda^{I,O}(i, j, 0, A)$, as we did for $\alpha_\lambda^{I,O}$. Before doing this, however, it will be helpful to introduce some new notation for the products of matrix using maximizations.

Let $M = [a_{ij}]_{n \times m}$ and $M' = [b_{ij}]_{m \times n}$ be two matrices. Let the *max-product* of M and M', denoted as $M \otimes M'$, be a matrix where each element is calculated as follows:

$$[M \otimes M']_{i,j} = \max_{1 \leq p \leq m} a_{ip} b_{pj}.$$

The kth power of matrix $\mathbf{P}$ can be defined through the max-product instead of the standard product: $\mathbf{P}^k_{\max} = \mathbf{P}^{k-1}_{\max} \otimes \mathbf{P}$.

Now, by expanding the second equation in (15) and making substitutions, (16) can be rewritten as:

$$\gamma^{I,O}(i, j, A) = \max_{k=0}^{\infty} \gamma_\lambda^{I,O}(i, j, k, A) = \max_{A' \in N} \gamma_\lambda^{I,O}(i, j, 0, A') \max_{k=0}^{\infty} \left[\mathbf{P}^k_{\max}\right]_{A', A} \tag{17}$$

Each element of the infinite power series of $\mathbf{P}$, $\max_{k=0}^{\infty}[\mathbf{P}^k_{\max}]_{A,B}$, is equal to the maximum probability among the probabilities of all derivations with λ-rules that rewrite A into B. Note that the longest possible derivation from A into B with maximum probability is the one which rewrites each non-terminal once and only once, since the existence of any cycle would necessarily cause the probability to be lower with respect to the same derivation without the cycle. Consequently, the limit in (17) attains its maximum for values of k smaller than $|N|$, and can be calculated as:

$$\widetilde{\mathbf{P}}_{A,B} = \max_{k=0\cdots|N|} \left[\mathbf{P}^k_{\max}\right]_{A,B}$$

Finally, we can reformulate algorithm $\gamma^{I,O}$ in the following manner:

$$\begin{aligned} \gamma^{I,O}(0,0,A) &= \max_{A'\in N} \delta(A',S)\cdot \widetilde{\mathbf{P}}_{A',A} \\ \gamma^{I,O}(i,j,A) &= \max_{A'\in N} \gamma_\lambda^{I,O}(i,j,0,A')\cdot \widetilde{\mathbf{P}}_{A',A}, \end{aligned} \tag{18}$$

where $\gamma_\lambda^{I,O}(i,j,0,A)$ can be calculated as shown in (15). The time complexity of algorithm $\gamma^{I,O}$ is $O(|x|\cdot|y|\cdot|R|)$ if dynamic programming is used.

2.2.3. Viterbi translation algorithm. The maximization contained in the expression for the approximate stochastic translation (given by Eq. (8)) can be calculated with a variation of the Viterbi algorithm shown in Section 2.2.2. For any given input string x, we will define some function γ^I such that:

$$\gamma^I(|x|,F) = \max_{\forall tf/input(tf)=x} Pr(tf \mid \Phi(T)). \tag{19}$$

The algorithm for calculating the function γ^I is described in a recursive manner by means of an auxiliary function γ_λ^I which is similar to function $\gamma_\lambda^{I,O}$ in the previous section. At each step of recursion in γ^I, function γ_λ^I keeps track of all the paths that are formed only by rules of the form $A \to B, zB, z \in \Delta^\star$, and, therefore, do not consume any input symbol. Assuming that $\gamma^I(i-1,A)$ is known, the auxiliary function γ_λ^I is recursively defined as follows, $\forall A \in N, \forall i \geq 0$:

$$\begin{aligned} \gamma_\lambda^I(i,0,A) &= \max_{\substack{A'\in N\\ z\in\Delta^\star}} \gamma^I(i-1,A')\cdot P(A' \to x_i A, zA) \\ \gamma_\lambda^I(i,j,A) &= \max_{\substack{A'\in N\\ z\in\Delta^\star}} \gamma_\lambda^I(i,j-1,A')\cdot P(A' \to A, zA) \quad 1 \leq j \leq k \end{aligned} \tag{20}$$

The value of $\gamma^I(i,A)$ can be calculated as an infinite maximization over all the probabilities of all possible derivations, which may include any number of rules of the form $A \to B, zB$:

$$\gamma^I(i,A) = \max_{k=0}^{\infty} \gamma_\lambda^I(i,k,A) \tag{21}$$

If we define some matrix $\mathbf{P}'$ of dimensions $|N| \times |N|$ as $\mathbf{P}'_{A,B} = \max_{\forall z \in \Delta^\star} P(A \rightarrow B, zB)$, for $A, B \in N$, and $\tilde{\mathbf{P}}'$ is defined as in (17), then algorithm (21) can be reformulated as follows, by a reasoning similar to the one in the previous section:

$$\begin{aligned} \gamma^I(0, A) &= \max_{A' \in N} \delta(A, S) \cdot \tilde{\mathbf{P}}'_{A',A} \\ \gamma^I(i, A) &= \max_{A' \in N} \gamma^I_\lambda(i, 0, A') \cdot \tilde{\mathbf{P}}'_{A',A}. \end{aligned} \tag{22}$$

2.2.4. Forward and backward algorithms for input and output. We will finally define two more new variants of the *forward* and *backward* algorithms. Following our notation, these versions will be denoted as α^I, β^I, α^O and β^O. They correspond to the calculation of the *forward* and *backward* values of probability when the derivations are followed by keeping track only of either the input or the output string. These functions will be useful in Section 3, where we will discuss the learning of the probabilities of schemata from translation samples.

The proper definition of these functions need an additional restriction over the structure of the STRs: rules of the form $A \rightarrow B, zB'$, for $z \in \Delta^\star$, must not form a loop for the application of α^I and β^I, and rules of the form $A \rightarrow aB, B$, for $a \in \Sigma \cup \{\lambda\}$, must not form a loop for the application of α^O and β^O.

The calculation of α^I will require the definition of an auxiliary recursive function α^I_λ, which is similar to the one used in the previous development. Assuming that the value of $\alpha^I(i-1, A), \forall A \in N, \forall i \geq 0$ is known, we will define α^I_λ as:

$$\begin{aligned} \alpha^I_\lambda(i, 0, A) &= \sum_{\substack{A_0 \in N \\ z \in \Delta^\star}} \alpha^I(i-1, A_0) \cdot P(A_0 \rightarrow x_i A, zA) \\ \alpha^I_\lambda(i, j, A) &= \sum_{\substack{A_j \in N \\ z \in \Delta^\star}} \alpha^I_\lambda(i, j-1, A_j) \cdot P(A_j \rightarrow A, zA) \quad 1 \leq j \leq k \end{aligned} \tag{23}$$

We can express $\alpha^I(i, A)$ as an infinite sum of the probabilities of all possible derivations which end up generating the non-terminal A and are composed only by rules of the form $A \rightarrow B, zB$, in a way similar to the one for γ^I:

$$\alpha^I(i, A) = \sum_{k=0}^{\infty} \alpha^I_\lambda(i, k, A) = \sum_{A' \in N} \alpha^I_\lambda(i, 0, A') \cdot \sum_{k=0}^{\infty} \left[\mathbf{P}''^k\right]_{A',A} \tag{24}$$

Here, matrix $\mathbf{P}''$ registers the probabilities of rules in T of the form $A \rightarrow B, zB$, so that each element is defined as $\mathbf{P}''_{A,B} = \sum_{\forall z \in \Delta^\star} P(A \rightarrow B, zB)$. Again, if matrix $\mathbf{I} - \mathbf{P}''$ can be inverted, then the value of the matrix power series is known and we can use expression (25) for calculating the exact value of $\alpha^I(i, A)$.

$$\alpha^I(i, A) = \sum_{A' \in N} \alpha^I_\lambda(i, 0, A') \cdot [(\mathbf{I} - \mathbf{P}'')^{-1}]_{A',A} \tag{25}$$

Similar reasonings yield the following analogous expressions for the three remaining algorithms. The definition of the matrix $\mathbf{P}'''$ for expressions (27) and (28) is $\mathbf{P}'''_{A,B} = \sum_{\forall a \in \Sigma \cup \{\lambda\}} P(A \to aB, B)$.

$$\beta^I(i, A) = \sum_{A' \in N} \beta^I_\lambda(i, 0, A') \cdot [(\mathbf{I} - \mathbf{P}'')^{-1}]_{A,A'} \tag{26}$$

$$\alpha^O(j, A) = \sum_{A' \in N} \alpha^O_\lambda(j, 0, A') \cdot [(\mathbf{I} - \mathbf{P}''')^{-1}]_{A',A} \tag{27}$$

$$\beta^O(j, A) = \sum_{A' \in N} \beta^O_\lambda(j, 0, A') \cdot [(\mathbf{I} - \mathbf{P}''')^{-1}]_{A,A'} \tag{28}$$

3. Probabilistic estimation

The stochastic translation schemata discussed in the previous section are statistical models which can describe probability distributions over the universe of all possible pairs of input-output strings, $\Sigma^\star \times \Delta^\star$. These distributions depend on the structure of each particular schema, and on the set of probability parameters associated with the set of rules. In practice, building one of these schemata for adequately modeling a certain probability distribution is, therefore, a process that may be performed in two separate phases. First, a non-stochastic schema is generated, and, second, a set of probability values for the rules in the schema is chosen. The generation of the structure can be done either manually or automatically—for instance, with some techniques for inferring grammars from samples (Casacuberta, 2000; Oncina, García & Vidal, 1993). Once a schema is given, the parameter set will often need to be estimated from a representative sample of translation pairs, so as to obtain a stochastic schema that approximates the real probability distribution as closely as possible.

The problem of estimating the parameters of a model from a finite set of data has been thoroughly studied in statistics. A well-known, general-purpose method for this is the so-called *expectation-maximization* method (Dempster, Laird & Rubin, 1977). Here we will use the Baum-Welch algorithm, a more specific version of the expectation-maximization method which is suitable for estimating the probabilities of rules in a SRT. The four estimation criteria presented in this paper follow the same general procedure. First, we define a function that depends both on the statistical parameters that we want to estimate and on the training translation pairs. This function is supposed to be sensitive to the *relevant* information in the sample, so that higher values of the function correspond to better approximations of the model to reality. Then we use the Baum-Welch algorithm or some variation of it for finding a local optimal value.

The Baum-Welch algorithm (Baum & Sell, 1968) allows us to approximate the optimal value of a function by an iterative process. Functions must be polynomials with non-negative coefficients. The following theorem indicates how to build certain *growth transformations*, so that the increase of the function value is guaranteed when they are applied for reestimation.

Theorem 1. *Let $P(\Theta)$ be a homogeneous polynomial in the variables $\Theta = \{\Theta_{ij}\}$ with non-negative coefficients. Let $\theta = \{\theta_{ij}\}$ be one point of $D = \{\theta_{ij} \mid \theta_{ij} \geq 0, \sum_{j=1}^{q_i} \theta_{ij} = 1,$*

$i = 1, \ldots, p$ and $j = 1, \ldots, q_i\}$. Let $Q(\theta)$ be one point of D defined as

$$Q(\theta)_{ij} = \frac{\theta_{ij}\left(\frac{\partial P(\Theta)}{\partial \Theta_{ij}}\right)_\theta}{\sum_{k=1}^{q_i} \theta_{ik}\left(\frac{\partial P(\Theta)}{\partial \Theta_{ik}}\right)_\theta} \tag{29}$$

such that $\sum_{k=1}^{q_i} \theta_{ik}(\partial P(\Theta)/\partial \Theta_{ik})_\theta \neq 0$. Then, $P(Q(\theta)) > P(\theta)$ unless $Q(\theta) = \theta$.

The growth transformations $Q(x)$ can now be used iteratively in a straight-forward algorithm:

```
INPUT: $P(\Theta)$
PROCEDURE:
    $\theta$ := initial values
    repeat
        compute $Q(\theta)$ using $P(\Theta)$ and (29)
        $\theta := Q(\theta)$
    until convergence
OUTPUT: $\theta$
```

We will also use an extension of the previous theorem to rational functions by Gopalakrishnan et al. (1991). We need to define some concepts before we can state the theorem. The function $R(\Theta) = S_1(\Theta)/S_2(\Theta)$ is a rational function over the domain D in Theorem 1, where $S_1(\Theta)$ and $S_2(\Theta)$ are polynomials with real coefficients and variables $\Theta = \{\Theta_{ij}\}$. Function $S_2(\Theta)$ only presents positive values in D. The polynomial $P_\theta(\Theta) = S_1(\Theta) - R(\theta)S_1(\Theta)$ will be denoted as P_θ. Given a rational function $R(\Theta)$ and a real number C, let us define the following functions:

$$\Gamma_{ij}(\theta; C) = \theta_{ij}\left(\frac{\partial P_\theta(\Theta)}{\partial \Theta_{ij}}(\theta) + C\right) \tag{30}$$

and

$$\Gamma_i(\theta; C) = \sum_{j=1}^{q_i} \Gamma_{ij}(\theta; C). \tag{31}$$

We will say that C is *admissible* (for $R(\Theta)$) if for any i, j and $\theta \in D$, then $\Gamma_{ij}(\theta; C) \geq 0$, and $\Gamma_i(\theta; C) > 0$.

Theorem 2. *Let $R(\Theta)$ be a rational function in the variables $\Theta = \{\Theta_{ij}\}$ with non-negative coefficients. Let $\theta = \{\theta_{ij}\}$ be one point of $D = \{\theta_{ij} | \theta_{ij} \geq 0, \sum_{j=1}^{q_i} \theta_{ij} = 1, i = 1, \ldots, p$ and $j = 1, \ldots, q_i\}$. Let $Q(\theta)$ be one point of D defined as*

$$Q(\theta)_{ij} = \frac{\theta_{ij}\left(\left(\frac{\partial P(\Theta)}{\partial \Theta_{ij}}\right)\theta + C\right)}{\sum_{k=1}^{q_i} \theta_{ik}\left(\left(\frac{\partial P(\Theta)}{\partial \Theta_{ik}}\right)\theta + C\right)}. \tag{32}$$

Then, there exists a constant N_R such that for any $C \geq N_R$, N_R is admissible for $R(\Theta)$, and, for such C, function Q is a growth transformation of $R(\Theta)$. This is, $R(Q(\theta)) > R(\theta)$, except if $Q(\theta) = \theta$.

The demonstration of this theorem developed in Gopalakrishnan et al. (1991) provides an expression for calculating N_R. However, in practice we often find that the algorithm converges much more slowly the greater the value of C is. Furthermore, if we reduce the value of C and make it smaller than that of the theoretical N_R, convergence still holds. This property is lost for values of C which are too low, in which case the algorithms produce oscillations of the value of $R(\theta)$. In practical executions of the algorithm, the value of C will be chosen by trial and error.

3.1. Estimation through maximization of the sample likelihood

The two estimation methods presented in this section are based on the maximization of the likelihood of the training data set within the translation model. Both methods differ in the way translation probabilities are calculated. The first one uses the probability of a translation as shown in (3), while the second one approximates it through the *Viterbi* probability of a translation, defined in (6).

3.1.1. Maximum likelihood estimation. A *sample* is formally a finite collection of translation pairs with repetitions allowed—i.e., a sample is a multiset drawn from $\Sigma^\star \times \Delta^\star$. The function that will constitute the criterion for reestimation will be the so-called *maximum likelihood* of a sample *TS*:

$$R_{MLE}(\Phi(T)) = \prod_{(x,y)\in TS} Pr(x, y \mid \Phi(T)) \tag{33}$$

Likelihood $R_{MLE}(\Phi(T))$ is the probability that a sample *TS* is *accepted* by the SRT T with parameters $\Phi(T)$. The *maximum likelihood estimation* (MLE) of $\Phi(T)$ is the value $\hat{\Phi}(T)$ which maximizes $R_{MLE}(\Phi(T))$. The corresponding growth transformations are drawn from the application of the Theorem 1:

$$\begin{aligned} &Q_{TS}^{MLE}(P(A \to aB, zB)) \\ &\quad = \frac{\sum_{(x,y)\in TS} \frac{1}{Pr(x,y|\Phi(T))} \sum_i \sum_j H(x, y, (A \to aB, zB), i, j)}{\sum_{(x,y)\in TS} \frac{1}{Pr(x,y|\Phi(T))} \sum_i \sum_j L(x, y, A, i, j)} \end{aligned} \tag{34}$$

where

$$\begin{aligned} H(x, y, (A \to aB, zB), i, j) &= \delta(a, x_{i\ldots i-1+|a|}) \cdot \delta(z, y_{j+1\ldots j+|z|}) \\ &\quad \cdot \alpha^{I,O}(x_{1\ldots i}, y_{1\ldots j}, A) \cdot P(A \to aB, zB) \\ &\quad \cdot \beta^{I,O}(x_{i+|a|\ldots|x|}, y_{j+|z|+1\ldots|y|}, B), \\ L(x, y, A, i, j) &= \alpha^{I,O}(x_{1\ldots i}, y_{1\ldots j}, A) \cdot \beta^{I,O}(x_{i-1+|a|\ldots|x|}, y_{j\ldots|y|}, A). \end{aligned}$$

3.1.2. Viterbi-like estimation. This method (from now on, VLE) attempts to maximize the Viterbi probability of the translation model for a given training sample. We say that a translation form *tf generates* a sentence pair (x, y) if $input(tf) = x$ and $output(tf) = y$. The Viterbi probability of the translation model is a function of the multiset of translation forms that generate the pairs in the training sample with maximum probability. This function is not a homogeneous polynomial and, therefore, cannot be used as an objective function for the Baum-Welch algorithm. Instead, we will define the objective function for a generic set of translation forms, so the function is polynomial and growth transformations can be drawn from the application of Theorem 1. Then, we will apply the growth transformations to the particular case of the set of maximum probability translation forms.

Given a sample *TS* and a set of translation forms *TF* that compose the pairs in *TS*, the objective function that we want to maximize is:

$$R_{VLE}(\Phi(T)) = \prod_{\substack{\forall tf \in TF: \\ (input(tf), \\ output(tf)) \in TS}} Pr(tf \mid \Phi(T)). \tag{35}$$

This function accomplishes the necessary conditions for applying Theorem 1. The following growth transformations are obtained, $\forall (A \to aB, zB) \in R, A, B \in N, a \in \Sigma \cup \{\lambda\}, z \in \Delta^\star$:

$$Q_{TS}^{VLE}(P(A \to aB, zB)) = \frac{\sum_{\forall tf \in TF} N((A \to aB, zB), tf)}{\sum_{\forall tf \in TF} N(A, tf)} \tag{36}$$

In this expression, $N((A \to aA, zB), tf)$ denotes the number of times that the rule $A \to aA$, zB has been used in $tf \in TF$; $N(A, tf)$ is the number of times that the non-terminal A has been used in $tf \in TF$, and Q_{TS}^{VLE} is an application from the space $\Phi(T)$ into itself.

When these transformations are used, the value of function (35) increases. As a particular case, *TF* can be chosen to be the set of translation forms of maximum probability, so that the Viterbi-like estimation is obtained. The search for the optimal translation form can be achieved as a byproduct of the Viterbi algorithm (18) by keeping track of the rules which produce the maximum probability derivations during the execution.

3.1.3. Computational complexity. The time complexity of the transformation rules in (34) and (36) is $O(|TS| \cdot L_x \cdot L_y \cdot |R|)$ for each iteration, where $|TS|$, $|R|$, L_x and L_y are the number of pairs in the sample, the number of rules in the schema and the maximum length of the input and output strings, respectively. This upper bound is deduced from the fact that the values of $\alpha^{I,O}$ and $\beta^{I,O}$ in MLE, and $\gamma^{I,O}$ in VLE, must be calculated once for each pair in the sample. The cost of these three algorithms is $O(|x| \cdot |y| \cdot |R|)$. Note that even though VLE has the same worst-case cost as MLE, its real cost is, in general, much lower.

3.1.4. Ambiguity. Maximum likelihood estimation and Viterbi-like estimation optimize the same objective function when the SRT is not grammatically ambiguous. Moreover, in this case the optimum of the function is global and is achieved in only one iteration. This result is similar to the one with stochastic grammars (Casacuberta, 1994).

3.2. Estimation through entropy measurements

The *entropy* $H(X)$ is a measure of the number of bits that are needed to specify the outcome of a random event X (Shannon, 1948). Intuitively, entropy can be understood as a plausible measure of the level of uncertainty in the event. The definition is shown below.

$$H(X) = -\sum_{x} Pr(x) \log Pr(x)$$

Similarly, the *conditional entropy* of the random event X given the random event Y is a measure of the uncertainty in X given the outcome of Y:

$$H(X \mid Y) = -\sum_{x,y} Pr(x, y) \log Pr(x \mid y)$$

A statistical translation system can be interpreted as a bidirectional channel where two sources X and Y produce sentences in each of the languages involved, respectively, following the real distribution of probability of sentences in either language. The bidirectional channel establishes a relationship of translation between the two languages. The probability that sentences x and y from X and Y are a translation of each other is $Pr(x, y)$.

If we assume that a real translation situation between two languages can be approached as one of these statistical translation systems, then our goal is to obtain a statistical model m which is as close as possible to the real system. Therefore, let $H_m(X \mid Y)$ stand for the conditional entropy of X given Y with respect to the probability distributions in the model m. It has been demonstrated in Brown (1987) that the inequality $H_m(X \mid Y) \geq H(X \mid Y)$ always holds. Furthermore, the smaller the value of $H_m(X \mid Y)$ is, the more the distribution in model m resembles the real distribution. $H_m(X \mid Y)$ and $H(X \mid Y)$ are equal when the two distributions are the same.

This observation permits us to define two different criteria for the estimation of the probabilities in the model. As a first approximation, we would like to choose the model, m, to minimize $H_m(Y \mid X)$. On the other hand, as a second approximation, note that the model is symmetrical with respect to the direction of translation. Hence, we might also want the model to minimize $H_m(X \mid Y)$. These two simultaneous goals can be achieved by looking for a model that minimizes the sum of both values,

$$H_m(X \mid Y) + H_m(Y \mid X) = -\sum_{x,y} Pr(x, y)\ \log \frac{Pr_m^2(x, y)}{Pr_m(x) \cdot Pr_m(y)},$$

where $Pr_m(x, y)$, $Pr_m(x)$ and $Pr_m(y)$ are the probabilities yielded by the model.

3.2.1. Minimum conditional entropy estimation. The two criteria just mentioned can be used to estimate the parameter set of an SRT. For a given SRT T, with parameter set $\Phi(T)$, we will use $Pr(x, y \mid \Phi(T))$ as the probability $Pr_m(x, y)$. Since we do not know the real probability distribution, $Pr(x, y)$, we must instead assume that the pairs (x, y) in our sample

TS are representative, and choose $\Phi(T)$ to minimize

$$-\sum_{(x,y)\in TS} \log \frac{Pr^2(x, y \mid \Phi(T))}{Pr(x \mid \Phi(T)) \cdot Pr(y \mid \Phi(T))} \tag{37}$$

Therefore, the criterion function for the *minimum conditional entropy estimation* (MCEE), which must be maximized due to a sign change, will be:

$$R_{MCEE}(\Phi(T)) = \prod_{(x,y)\in TS} \frac{Pr^2(x, y \mid \Phi(T))}{Pr(x \mid \Phi(T)) \cdot Pr(y \mid \Phi(T))} \tag{38}$$

The reestimation formulae for this function will be obtained through the application Theorem 2. Let Q_{TS}^{MCEE} be a transformation from the space $\Phi(T)$ into itself. Then, $\forall(A \to aB, zB) \in R$ we have:

$$\begin{aligned} &Q_{TS}^{MCEE}(P(A \to aB, zB)) \\ &\quad = \frac{P(A \to aB, zB)\left(\frac{\partial \log R_{MCEE}(\Phi(T))}{\partial P(A\to aB,zB)} + C\right)}{\sum_{a',z',B'} P(A \to a'B', z'B')\left(\frac{\partial \log R_{MCEE}(\Phi(T))}{\partial P(A\to a'B',z'B')} + C\right)} \end{aligned} \tag{39}$$

where the numerator of the second term can be deduced as

$$\begin{aligned} &P(A \to aB, zB)\frac{\partial \log R_{MCEE}(\Phi(T))}{\partial P(A \to aB, zB)} \\ &\quad = \sum_{(x,y)\in TS} \left(\frac{2}{Pr(x, y \mid \Phi(T))} P(A \to aB, zB)\frac{\partial Pr(x, y \mid \Phi(T))}{\partial P(A \to aB, zB)}\right. \\ &\quad\quad - \frac{1}{Pr(x \mid \Phi(T))} P(A \to aB, zB)\frac{\partial Pr(x \mid \Phi(T))}{\partial P(A \to aB, zB)} \\ &\quad\quad \left. - \frac{1}{Pr(y \mid \Phi(T))} P(A \to aB, zB)\frac{\partial Pr(y \mid \Phi(T))}{\partial P(A \to aB, zB)}\right) \end{aligned} \tag{40}$$

and C is an admissible constant (Gopalakrishnan et al., 1991).

The first term in the sum is similar to the term which appears in MLE, and can be computed as in expression (34). The second term can be easily calculated by considering the *input grammar* of T, which is defined to be $G_i = (N, \Sigma, R_i, S, P_i)$, where, if $(A \to aB, zB) \in R$, then $(A \to aB) \in R_i$ and $P_i(A \to aB) = P(A \to aB, zB)$. The expressions which are obtained for MLE in stochastic grammars can be used (Casacuberta, 1996). These formulae need the use of algorithms α^I and β^O presented in Section 2. Similarly, the third term is obtained by considering the *output grammar* of $T : G_o = (N, \Sigma, R_o, S, P_o)$, where, if $(A \to aB, zB) \in R$, then $(A \to zB) \in R_o$ and $P_o(A \to aB) = P(A \to aB, zB)$. This can also be calculated as shown in (Casacuberta, 1996).

3.2.2. Conditional maximum likelihood estimation. This method is based on the minimization of the conditional entropy in only one of the directions of the translation. Conditional entropy can be calculated as:

$$H_m(Y|X) = -\sum_{x,y} Pr(x, y)\ \log \frac{Pr_m(x, y)}{Pr_m(x)}.$$

We can now reason as we did for MCEE and define an objective function to be maximized:

$$R_{CMLE}(\Phi(T)) = \prod_{(x,y)\in TS} \frac{Pr(x, y \mid \Phi(T))}{Pr(x \mid \Phi(T))} \tag{41}$$

The denomination of *conditional maximum likelihood* is due to the fact that this criterion had already been defined in Casacuberta (1996) and Cardin, Narmandin and DeMori (1994) as a maximization of the conditional probability of an output sentence given an input sentence, which happens to be equivalent:

$$\operatorname*{argmax}_{\Phi(T)} Pr(y|x, \Phi(T)) = \operatorname*{argmax}_{\Phi(T)} \frac{Pr(x, y \mid \Phi(T))}{Pr(x \mid \Phi(T))} \tag{42}$$

The growth transformations for R_{CMLE} can be obtained through the application of Theorem 2. Thus, let Q_{TS}^{CMLE} be a closed transformation in the space $\Phi(T)$. Then, $\forall (A \to aB, zB) \in R$,

$$\begin{aligned} &Q_{TS}^{CMLE}(P(A \to aB, zB)) \\ &\quad = \frac{P(A \to aB, zB)\left(\frac{\partial \log R_{CMLE}(\Phi(T))}{\partial P(A \to aB, zB)} + C\right)}{\sum_{a',z',B'} P(A \to a'B', z'B')\left(\frac{\partial \log R_{CMLE}(\Phi(T))}{\partial P(A \to a'B', z'B')} + C\right)} \end{aligned} \tag{43}$$

where the numerator of the second term can be deduced as

$$\begin{aligned} &P(A \to aB, zB) \frac{\partial \log R_{CMLE}(\Phi(T))}{\partial P(A \to aB, zB)} \\ &\quad = \sum_{(x,y)\in TS} \left(\frac{1}{Pr(x, y|\Phi(T))} P(A \to aB, zB) \frac{\partial Pr(x, y|\Phi(T))}{\partial P(A \to aB, zB)} \right. \\ &\qquad \left. - \frac{1}{Pr(x|\Phi(T))} P(A \to aB, zB) \frac{\partial Pr(x|\Phi(T))}{\partial P(A \to aB, zB)} \right) \end{aligned} \tag{44}$$

and C is an admissible constant (Gopalakrishnan, 1991).

3.2.3. Discussion. A different method based on entropy measures has been proposed in Casacuberta (1996). It was a straight-forward application of the *maximum mutual information estimation* (MMIE) by Brown (1987) to stochastic translation schemata. In Brown's

MMIE, it is claimed that minimizing the conditional entropy $H(Y \mid X)$ is equivalent to maximizing the *mutual information*, $I(X;Y)$, since

$$H(Y \mid X) = H(X) - I(X;Y).$$

$H(X)$ represents the entropy of the source X and is supposed to be determined by some known language model and, therefore, fixed. However, this approximation is not adequate if (as is our case) the language model that is being used is not independent from the translation model. When dealing with SRTs the probabilities of sources X and Y given by the model (i.e., $Pr_m(x) = Pr(x \mid \Phi(T))$ and $Pr_m(y) = Pr(y \mid \Phi(T))$) are a function of the set of parameters of the SRT, $\Phi(T)$. Therefore, they are *not* fixed during the estimation process and MMIE cannot be applied as done in Casacuberta (1996).

MMIE could be used if an independent model for the probabilities of the input and the output sentences were given. Such a model could be, for instance, a probabilistic model that represented information about the context of appearance of sentences within a line of discourse.

4. Experiments

Some experiments were carried out to compare the four estimation methods described above. The selected task was the translation of Spanish sentences into English, as defined in project EuTrans-I (Amengual et al., 1998). The semantic domain of the sentences is restricted to tourist information, consisting in sentences that a hotel guest would address to a hotel receptionist at the information desk. A parallel corpus of paired Spanish-English sentences was artificially generated.

The structure of an SRT was inferred from the corpus by means of a new method for building finite-state transducers using regular grammars and morphisms (Casacuberta, 2000). The inferred SRT contained 490 non-terminal symbols and 1438 rules.

Training was done with 5 different series of training sets. Each series was composed of 5 mutually including sets of increasing size, containing 25, 50, 100, 200 and 300 pairs, respectively. The training process was stopped when the value of the objective function did not increase significatively. A set containing 500 different translation pairs was used for testing in each one of the five series. The test sets were disjoint to all training sets in

Table 1. Word-error rate (in %) for the translation of the test set for different number of pairs of sentences in the training set and differents estimation methods.

Number of pairs	MLE	VLE	MCEE	CMLE
25	33	33	27	26
50	30	30	27	26
100	26	27	27	25
200	24	25	25	23
300	25	26	26	25

the series. All the results were averaged over the 5 series of experiments and are shown on Table 1. The performance was evaluated in terms of translation *word-error rate* (WER), which is the percentage of output words that have to be inserted, deleted and substituted in order to exactly match the corresponding expected translations.

The behaviour of the estimation methods in these experiments can be observed in the results shown in Table 1. The two methods based on entropy (CMLE and MCEE) performed significantly better for small training sets (25 and 50 training pairs). For greater training sets the results were similar for all four methods and no significant differences can be appreciated. For small training sets CMLE performed slightly better that MCEE and MLE did slightly better than Viterbi, though the differences are not very remarkable.

5. Conclusions

In this paper, we have presented a series of algorithms for obtaining some probabilistic parameters associated with stochastic regular syntax-directed translation schemata. These algorithms include the computation of the probability of a translation and the search for the highest probability translation of a given input. They have been defined so as to operate with schemata that incorporate λ-rules—i.e., rules in which the input and/or the output string is an empty string.

Four methods have been presented for the estimation of the probabilities of the stochastic syntax-directed translation schemata from translation samples. Two of them are based on the maximimization of the sample likelihood, and the other two consider the possibility of maximizing certain entropy measurements. The minimum conditional entropy estimation is a new criterion, while the other three are an adaptation of estimation methods from other fields to translation schemata. The criteria based on entropy have been proposed to deal with sparse training data.

Some experiments were carried out on a translation task between natural languages and the results show how the estimation criteria based on entropy measurements perform better than maximum likelihood estimation and Viterbi-like estimation in reducing word-error rates for small sample sizes.

Acknowledgment

The authors wish to thank the three anonymous reviewers for their criticisms and suggestions.

References

Aho, A. V. & Ullman, J. D. (1972). *The theory of parsing, translation and compiling*. vol. 1. Prentice-Hall.

Amengual, J. C., Benedí, J. B., Casacuberta, F., Castaño, A., Castellanos, A., Jiménez, V. M., Llorens, D., Marzal, A., Pastor, M., Prat, F., Vidal, E., & Vilar, J. M. (1998). *The* EUTRANS-I *speech translation system.* submitted to Machine Translation.

Bahl, L.R., Jelinek, F., & Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *5:2*, 179–196.

Baum, L. E. & Sell, G. R. (1968). Growth transformations for functions on manifolds. *Pacific Journal of Mathematics, 26:2*, 211–227.

Berstel, J. (1979). *Transductions & Context-Free Languages*. Stuttgart: B.G. Teubner.

Brown, P. F. (1987). The acoustic-modelling problem in automatic speech recognition. Ph. Dissertation. Carnegie-Mellon University.

Cardin, R., Normandin, Y., & DeMori, R. (1994). High performance connected digit recognition using maximum mutual information estimation. *IEEE Trans. on Speech & Audio Processing, 2:2*, 300–311.

Casacuberta, F. (1900). Some relations among stochastic finite state networks used in automatic speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligente, PAMI-12* (7), 691–693.

Casacuberta, F. (1994). Statistical estimation of stochastic context-free grammars using the inside-outside algorithm & a transformation on grammars. R. Carrasco & J. Oncina, (Eds.). *Grammatical inference and applications, Lecture notes in artificial intelligence* (Vol. 862), pp. 119–129, Springer-Verlag.

Casacuberta, F. (1995). Probabilistic estimation of stochastic regular syntax-directed translation schemes. *Proc. of the VI Spanish Symposium on Pattern Recognition and Image Analysis* (pp. 201–207).

Casacuberta, F. (1996). Growth transformations for probabilistic functions of stochastic grammars. *International Journal of Pattern Recognition and Artificial Intelligence, 10:3*, 183–201.

Casacuberta, F. (1996). Maximum mutual information and conditional maximum likelihood estimation of stochastic regular syntax-directed translation schemes. L. Miclet & C. de la Higuera, (Eds.). *Grammatical inference: Learning syntax from sentences*. *Lecture notes in artificial intelligence* (Vol. 1147, pp. 282–291). Springer-Verlag.

Casacuberta, F. (2000). Morphic generator translation inference, (to be submited to ICGI'2000).

Casacuberta, F. & de la Higuera, C. (2000). Computational complexity of problems on probabilistic grammars and transducers (to be published).

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society*, Ser. B, *39:1*, 1–38.

Fu, K. S. (Ed.) (1982). *Syntactic pattern recognition applications*. Englewood Cliffs, NJ: Prentice-Hall.

Gildea, D. & Jurafsky, D. (1996). Learning bias and phonological-rule induction. *Computational Linguistics, 22:4*, 497–530.

González, R. C. & Thomason, M. G. (1978). *Syntactic pattern recognition: An introduction*, Reading, MA: Addison-Wesley.

Gopalakrishnan, P. S., Kanevsky, D., Nádas, A., & Nahamoo, D. (1991). An inequality for rational functions with applications to some statistical estimation problems. *IEEE Transactions on Information Theory, 37:1*.

Jelinek, F. & Lafferty, J. D. (1991). Computation of the probability of initial substring generation by stochastic context-free grammars. *Computational Linguistics, 17:3*, 315–323.

Maryanski, F. J. & Thomason, M. G. (1979). Properties of stochastic syntax-directed translation schemata. *International Journal of Computer and Information Sciences, 8:2*, 89–110.

Merhav, N. & Ephraim, Y. (1991). Maximum likelihood hidden Markov modelling using a dominant sequence of states. *IEEE Transactions on Signal Processing, 39:9*, 2111–2115.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics, 23:2*, 269–311.

Nádas, A., Hahamoo, D., & Picherny, M. (1988). On a model-robust training method for speech recognition. *Trans. on Acoustic, Speech and Signal Processing, 36:9*, 1432–1435.

Oflazer, K. (1996). Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Computational Linguistics*, *22*(1), 73–89.

Oncina, J., García, P., & Vidal, E. (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis & Machine Intelligence, 15:5*, 448–458.

Roche, E. & Schabes, Y. (1995). Deterministic part-of-speech tagging with finite-state transducers. *Computational Linguistics, 21:2*, 227–253.

Sánchez, J. A. & Benedí, J. M. (1997). Consistency of stocastic context-free grammars from probabilistic estimation based on growth transformation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19:9*, 1052–1059.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, *27*, 379–423, (Part I); pp. 623–656 (Part II).

Thomason, M. G. (1976). Regular stochastic syntax-directed translations. Technical Report CS-76-17.

Vidal, E. (1997). Finite-state speech-to-speech translation. *Proceedings of the International Conference on Acoustic, Speech and Signal Processing* (Vol.1, pp. 111–114. Munich, Germany).

Vidal, E., Casacuberta, F., & García, P. (1995). Grammatical inference and speech recognition. *New Advances and Trends in Speech Recognition and Coding*. NATO ASI Series. (pp. 174–191), Springer-Verlag.

Received December 1, 1998
Revised March 1, 2000
Accepted June 8, 2000
Final manuscript July 28, 2000