A Recursive Statistical Translation Model[*]

Juan Miguel Vilar
Dpto. de Lenguajes y Sistemas Informáticos
Universitat Jaume I
Castellón (Spain)
jvilar@lsi.uji.es

Enrique Vidal
Dpto. de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Instituto Tecnológico de Informática
Valencia (Spain)
evidal@iti.upv.es

Abstract

A new model for statistical translation is presented. A novel feature of this model is that the alignments it produces are hierarchically arranged. The generative process begins by splitting the input sentence in two parts. Each of the parts is translated by a recursive application of the model and the resulting translation are then concatenated. If the sentence is small enough, a simpler model (in our case IBM's model 1) is applied.

The training of the model is explained. Finally, the model is evaluated using the corpora from a large vocabulary shared task.

1 Introduction

Suppose you were to find an English translation for a Spanish sentence. One possible approach is to assume that every English sentence is a candidate but that different English sentences have different probabilities of being the correct translation. Then, the translation task can be divided in two parts: define an adequate probability distribution that answers to the question "given this English sentence, which is the probability that it is a good translation of that Spanish sentence?"; and use that distribution in order to find the most likely translation of your input sentence.

[*]Work partially supported by Bancaixa through the project "Sistemas Inductivos, Estadísticos y Estructurales, para la Traducción Automática (Siesta)".

This approach is referred to as the statistical approach to machine translation. The usual approach is to define an statistical model and train its parameters from a training corpus consisting in pairs of sentences that are known to be translation of each other. Different models have been presented in the literature, see for instance (Brown et al., 1993; Och and Ney, 2004; Vidal et al., 1993; Vogel et al., 1996). Most of them rely on the concept of alignment: a mapping from words or groups of words in a sentence into words or groups in the other (in the case of (Vidal et al., 1993) the mapping goes from rules in a grammar for a language into rules of a grammar for the other language). This concept of alignment has been also used for tasks like authomatic vocabulary derivation and corpus alignment (Dagan et al., 1993).

A new statistical model is proposed in this paper, which was initially introduced in (Vilar Torres, 1998). This model is designed so that the alignment between two sentences can be seen in an structured manner: each sentence is divided in two parts and they are put in correspondence; then each of those parts is similarly divided and related to its translation. This way, the alignment can be seen as a tree structure which aligns progressively smaller segments of the sentences. This recursive procedure gives its name to the model: MAR, which comes from "Modelo de Alineamiento Recursivo", which is Spanish for "Recursive Alignment Model".

The rest of the paper is structured as follows: after a comment on previous works, we introduce the notation that we will use throughout the paper, then we briefly explain the model 1 from IBM, next we introduce our model, then we explain the process of parameter estimation, and how to use the model to translate new test sentences. Finally, we present some experiments and results, together with conclusions.

2 Previous works

The initial formulation of the proposed model, including the training procedures, was presented in (Vilar Torres, 1998), along with preliminary experiments in a small translation task which provided encouraging results.

This model shares some similarities with the stochastic inversion transduction grammars (SITG) presented by Wu in (Wu, 1997). The main point in common is the type of possible alignments considered in both models. Some of the properties of these alignments are studied in (Zens and Ney, 2003). However, the parametrizations of SITGs and the MAR are completely different. The generative process of SITGs produces simultaneously the input and output sentences and the parameters of the model refer to the rules of the nonterminals. This provides a symmetry to both input and output sentences. In contrast, our model clearly distinguishes the input and output sentences and the parameters are based on observable properties of the strings (their lengths and the words composing them). On the other hand, the MAR idea of splitting the sentences until a simple structure is found, also appears in the Divisive Clustering approach presented in (Deng et al., 2004). Again, the main difference lies in the probabilistic modeling of the alignments. In Divisive Clustering a uniform distribution on the alignments is assumed while MAR uses a explicit parametrization.

3 Some notation

In the rest of the paper, we use the following notation. Sentences are taken as concatenations of symbols (words) and represented using a letter and a small bar, like in $\bar{x}$. The individual words are designed by the name of the sentence and a subindex indicating the position, so $\bar{x} = x_1 x_2 \dots x_n$. The length of a sentence is indicated by $|\bar{x}|$. Segments of a sentence are denoted by $\bar{x}_i^j = x_i \dots x_j$. For the substrings of the form $\bar{x}_i^{|\bar{x}|}$ we use the notation $\bar{x}_{i.}$.

Consistently, $\bar{x}$ denotes the input sentence and $\bar{y}$ its translation and both are assumed to have at least one word. The input and output vocabularies are $\mathcal{X}$ and $\mathcal{Y}$, respectively. Finally, we assume that we are presentend a set $\mathcal{M}$ for training our models. The elements of this set are pairs $(\bar{x}, \bar{y})$ where $\bar{y}$ is a possible translation for $\bar{x}$.

4 IBM's model 1

IBM's model 1 is the simplest of a hierarchy of five statistical models introduced in (Brown et al., 1993). Each model of the hierarchy can be seen as a refinement of the previous ones. Although model 1, which we study here, relies on the concept of alignment, its formulation allows an interpretation of it as a relationship between multisets of words (the order of the words is irrelevant in the final formula).

A word of warning is in order here. The model we are going to present has an important difference with the original: we do not use the empty word. This is a virtual word which does not belong to the vocabulary of the task and that is added to the beginning of each sentence in order to allow words in the output that cannot be justified by the words in the input. We have decided not to incorporate it because of the use we are going to make of the model. As we will see, model 1 is going to be used repeatedly over different substrings of the input sentence in order to analyze their contribution to the total translation. This means that we would have an empty word in each of these substrings. We have decided to avoid this "proliferation" of empty words. Future work may introduce the concept in a more appropriate way.

The model 1 makes two assumptions. That a *stochastic dictionary* can be employed to model the probability that word y is the translation of word x and that all the words in the input sentence have the same weight in producing a word in the output. This leads to:

$$p_I(\bar{y} \mid \bar{x}) = \frac{\varepsilon(|\bar{x}|, |\bar{y}|)}{|\bar{x}|^{|\bar{y}|}} \prod_{j=1}^{|\bar{y}|} \sum_{i=1}^{|\bar{x}|} t(y_j \mid x_i). \qquad (1)$$

Where t is the stochastic dictionary and ε represents a table that relates the length of the alignment with the length of the input sentence (we assume that there is a finite range of possible lengths). This explicit relations between the lengths is not present in the original formulation of the model, but we prefer to include it so that the probabilities are adequately normalized.

Clearly, this model is not adequate to describe complex translations in which complicated patterns and word order changes may appear. Nevertheless, this model can do a good job to describe the translation of short segments of texts. For example, it can be adequate to model the translation of the Spanish “gracias” into the English “thank you”.

5 A Recursive Alignment Model

To overcome that limitation of the model we will take the following approach: if the sentence is complex enough, it will be divided in two and the two halves will be translated independently and joined later; if the sentence is simple, the model 1 will be used.

Let us formalize this intuition for the generative model. We are given an input sentence $\bar{x}$ and the first decission is whether $\bar{x}$ is going to be translated by IBM’s model 1 or it is complex enough to be translated by MAR. In the second case, three steps are taken: a cut point of $\bar{x}$ is defined, each of the resulting parts are translated, and the corresponding translations are concatenated. For the translation of the second step, the same process is *recursively* applied. The concatenation of the third step can be done in a “direct” way (the translation of the first part and then the translation of the second) or in an “inverse” way (the translation of the second part and then the translation of the first). The aim of this choice is to allow for the differences in word order between the input and ouput languages.

So, we are proposing an alignment model in which IBM’s model 1 will account for translation of elementary segments or individual words while translation of larger and more complex segments or whole sentences will rely on a hierarchical alignment pattern in which model 1 alignments will be on the lowest level of the hierarchy.

Following this discussion, the model can be formally described through a series of four random experiments:

- The first is the selection of the model. It has two possible outcomes: IBM and MAR, with obvious meanings.
- The second is the choice of b, a cut point of $\bar{x}$. The segment $\bar{x}_1^b$ will be used to generate one of the parts of the translation, the segment $\bar{x}_{b+1}^{\cdot}$ will generate the other. It takes values from 1 to $|\bar{x}| - 1$.
- The third is the decision about the order of the concatenation. It has two possible outcomes: D (for direct) and I (for inverse).
- The fourth is the translation of each of the halves of $\bar{x}$. They take values in $\mathcal{Y}^+$.

The translation probability can be approximated as follows:

$$p_T(\bar{y} \mid \bar{x}) = \Pr(M = \text{IBM} \mid \bar{x}) p_I(\bar{y} \mid \bar{x}) \\ + \Pr(M = \text{MAR} \mid \bar{x}) p_M(\bar{y} \mid \bar{x}).$$

The value of $p_I(\bar{y} \mid \bar{x})$ corresponds to IBM’s model 1 (Equation 1). To derive $p_M(\bar{y} \mid \bar{x})$, we observe that:

$$p_M(\bar{y} \mid \bar{x}) = \\ \sum_{b=1}^{|\bar{x}|-1} \Pr(b \mid \bar{x}) \\ \sum_{d \in \{D,I\}} \Pr(d \mid b, \bar{x}) \\ \sum_{\bar{y}_1 \in \mathcal{Y}^+} \Pr(\bar{y}_1 \mid b, d, \bar{x}) \\ \sum_{\bar{y}_2 \in \mathcal{Y}^+} \Pr(\bar{y}_2 \mid b, d, \bar{x}, \bar{y}_1) \Pr(\bar{y} \mid d, b, \bar{x}, \bar{y}_1, \bar{y}_2).$$

Note that the probability that $\bar{y}$ is generated from a pair $(\bar{y}_1, \bar{y}_2)$ is 0 if $\bar{y} \neq \bar{y}_1 \bar{y}_2$ and 1 if $\bar{y} = \bar{y}_1 \bar{y}_2$, so the last two lines can be rewritten as:

$$\sum_{\bar{y}_1 \in \mathcal{Y}^+} \Pr(\bar{y}_1 \mid b, d, \bar{x}) \\ \sum_{\bar{y}_2 \in \mathcal{Y}^+} \Pr(\bar{y}_2 \mid b, d, \bar{x}, \bar{y}_1) \Pr(\bar{y} \mid b, d, \bar{x}, \bar{y}_1, \bar{y}_2) \\ = \sum_{\substack{\bar{y}_1, \bar{y}_2 \in \mathcal{Y}^+ \\ \bar{y} = \bar{y}_1 \bar{y}_2}} \Pr(\bar{y}_1 \mid b, d, \bar{x}) \Pr(\bar{y}_2 \mid b, d, \bar{x}, \bar{y}_1) \\ = \sum_{\bar{y}_1 \,\in\, \text{pref}(\bar{y}) - \bar{y}} \Pr(\bar{y}_1 \mid b, d, \bar{x}) \Pr(\bar{y}_1^{-1} \bar{y} \mid b, d, \bar{x}, \bar{y}_1) \\ = \sum_{c=1}^{|\bar{y}|-1} \Pr(\bar{y}_1^c \mid b, d, \bar{x}) \Pr(\bar{y}_{c+1}^{\cdot} \mid b, d, \bar{x}, \bar{y}_1^c),$$

where $\mathrm{pref}(\bar{y})$ is the set of *prefixes* of $\bar{y}$. And finally:

$$p_M(\bar{y} \mid \bar{x}) = \sum_{b=1}^{|\bar{x}|-1} \Pr(b \mid \bar{x}) \sum_{d \in \{D,I\}} \Pr(d \mid b, \bar{x}) \sum_{c=1}^{|\bar{y}|-1} \Pr(\bar{y}_1^c \mid b, d, \bar{x}) \Pr(\bar{y}_{c+1}^{\cdot} \mid b, d, \bar{x}, \bar{y}_1^c). \tag{2}$$

The number of parameters of this model is very large, so it is necessary to introduce some simplifications in it. The first one relates to the decision of the *translation model*: we assume that it can be done just on the basis of the length of the input sentence. That is, we cat set up two tables, $\mathcal{M}_I$ and $\mathcal{M}_M$, so that

$$\Pr(M = \text{IBM} \mid \bar{x}) \approx \mathcal{M}_I(|\bar{x}|),$$
$$\Pr(M = \text{MAR} \mid \bar{x}) \approx \mathcal{M}_M(|\bar{x}|).$$

Obviously, for any $\bar{x} \in \mathcal{X}^+$, we will have $\mathcal{M}_I(|\bar{x}|) + \mathcal{M}_M(|\bar{x}|) = 1$. On the other hand, since it is not possible to break a one word sentence, we define $\mathcal{M}_I(1) = 1$. This restriction comes in the line mentioned before: the translation of longer sentences will be structured whereas shorter ones can be translated directly.

In order to decide the *cut point*, we will assume that the probability of cutting the input sentence at a given position b is most influenced by the words around it: x_b and x_{b+1}. We use a table $\mathcal{B}$ such that:

$$\Pr(b \mid \bar{x}) \approx \frac{\mathcal{B}(x_b, x_{b+1})}{\sum_{i=1}^{|\bar{x}|-1} \mathcal{B}(x_i, x_{i+1})}.$$

This can be interpreted as having a weight for each pair of words and normalizing these weights in each sentence in order to obtaing a proper probability distribution.

Two more tables, $\mathcal{D}_D$ and $\mathcal{D}_I$, are used to store the probabilities that the *alignment be direct or inverse*. As before, we assume that the decission can be made on the basis of the symbols around the cut point:

$$\Pr(d = D \mid b, \bar{x}) = \mathcal{D}_D(x_b, x_{b+1}),$$
$$\Pr(d = I \mid b, \bar{x}) = \mathcal{D}_I(x_b, x_{b+1}).$$

Again, we have $\mathcal{D}_D(x_b, x_{b+1}) + \mathcal{D}_I(x_b, x_{b+1}) = 1$ for every pair of words (x_b, x_{b+1}).

Finally, a probability must be assigned to the translation of the two halves. Assuming that they are independent we can apply the model in a recursive manner:

$$\Pr(\bar{y}_1^c \mid b, d, \bar{x}) \approx \begin{cases} p_T(\bar{y}_1^c \mid \bar{x}_1^b) & \text{if } d = D, \\ p_T(\bar{y}_1^c \mid \bar{x}_{b+1}^{\cdot}) & \text{if } d = I, \end{cases}$$

$$\Pr(\bar{y}_{c+1}^{\cdot} \mid b, d, \bar{x}, \bar{y}_1^c) \approx \begin{cases} p_T(\bar{y}_{c+1}^{\cdot} \mid \bar{x}_{b+1}^{\cdot}) & \text{if } d = D, \\ p_T(\bar{y}_{c+1}^{\cdot} \mid \bar{x}_1^b) & \text{if } d = I. \end{cases}$$

Finally, we can rewrite (2) as:

$$p_M(\bar{y} \mid \bar{x}) = \sum_{b=1}^{|\bar{x}|-1} \frac{\mathcal{B}(x_b, x_{b+1})}{\sum_{i=1}^{|\bar{x}|-1} \mathcal{B}(x_i, x_{i+1})} \cdot \left(\mathcal{D}_D(x_b, x_{b+1}) \sum_{c=1}^{|\bar{y}|-1} p_T(\bar{y}_1^c \mid \bar{x}_1^b) p_T(\bar{y}_{c+1}^{\cdot} \mid \bar{x}_{b+1}^{\cdot}) + \mathcal{D}_I(x_b, x_{b+1}) \sum_{c=1}^{|\bar{y}|-1} p_T(\bar{y}_{c+1}^{\cdot} \mid \bar{x}_1^b) p_T(\bar{y}_1^c \mid \bar{x}_{b+1}^{\cdot}) \right).$$

The final form of the complete model is then:

$$p_T(\bar{y} \mid \bar{x}) = \mathcal{M}_I(|\bar{x}|) p_I(\bar{y} \mid \bar{x}) + \mathcal{M}_M(|\bar{x}|) \sum_{b=1}^{|\bar{x}|-1} \frac{\mathcal{B}(x_b, x_{b+1})}{\sum_{i=1}^{|\bar{x}|-1} \mathcal{B}(x_i, x_{i+1})} \cdot \left(\mathcal{D}_D(x_b, x_{b+1}) \sum_{c=1}^{|\bar{y}|-1} p_T(\bar{y}_1^c \mid \bar{x}_1^b) p_T(\bar{y}_{c+1}^{\cdot} \mid \bar{x}_{b+1}^{\cdot}) + \mathcal{D}_I(x_b, x_{b+1}) \sum_{c=1}^{|\bar{y}|-1} p_T(\bar{y}_{c+1}^{\cdot} \mid \bar{x}_1^b) p_T(\bar{y}_1^c \mid \bar{x}_{b+1}^{\cdot}) \right). \tag{3}$$

6 Parameter estimation

Once the model is defined, it is necessary to find a way of estimating its parameters given a training corpus $\mathcal{M}$. We will use maximun likelihood estimation. In our case, the likelihood of the sample corpus is:

$$V = \prod_{(\bar{x},\bar{y}) \in \mathcal{M}} p_T(\bar{y} \mid \bar{x}).$$

In order to maximize V, initial values are given to the parameters and they are reestimated using repeatedly Baum-Eagon's (Baum and Eagon, 1967) and Gopalakrishnan's (Gopalakrishnan et al., 1991) inequalities. Let P be a parameter of the model (except for those in $\mathcal{B}$) and let $\mathcal{F}(P)$ be its "family" (i.e. the set of parameters such that $\sum_{Q \in \mathcal{F}(P)} Q = 1$). Then, a new value of P can be computed as follows:

$$
\begin{aligned}
\mathcal{N}(P) &= \frac{P\dfrac{\partial V}{\partial P}}{\displaystyle\sum_{Q \in \mathcal{F}(P)} Q\frac{\partial V}{\partial Q}} \\
&= \frac{\displaystyle\sum_{(\bar{x},\bar{y}) \in \mathcal{M}} \frac{P}{p_T(\bar{y} \mid \bar{x})} \frac{\partial p_T(\bar{y} \mid \bar{x})}{\partial P}}{\displaystyle\sum_{Q \in \mathcal{F}(P)} \sum_{(\bar{x},\bar{y}) \in \mathcal{M}} \frac{Q}{p_T(\bar{y} \mid \bar{x})} \frac{\partial p_T(\bar{y} \mid \bar{x})}{\partial Q}} \\
&= \frac{\mathcal{C}(P)}{\displaystyle\sum_{Q \in \mathcal{F}(P)} \mathcal{C}(Q)},
\end{aligned}
\tag{4}
$$

where

$$
\mathcal{C}(P) = \sum_{(\bar{x},\bar{y}) \in \mathcal{M}} \frac{P}{p_T(\bar{y} \mid \bar{x})} \frac{\partial p_T(\bar{y} \mid \bar{x})}{\partial P}, \tag{5}
$$

are the "counts" of parameter P. This is correct as long as V is a polynomial in P. However, we have a problem for $\mathcal{B}$ since V is a rational function of these parameters. We can solve it by assuming, without lose of generality, that $\sum_{x_1,x_2 \in \mathcal{X}} \mathcal{B}(x_1, x_2) = 1$. Then Gopalakrishnan's inequality can be applied similarly and we get:

$$
\mathcal{N}(P) = \frac{C + \mathcal{C}(P)}{\displaystyle\sum_{Q \in \mathcal{F}(P)} C + \mathcal{C}(Q)}, \tag{6}
$$

where C is an adequate constant. Now it is easy to design a reestimation algorithm. The algorithm gives arbitrary initial values to the parameters (typically those corresponding to uniform probabilities), computes the counts of the parameters for the corpus and, using either (4) or (6), gets new values for the parameters. This cycle is repeated until a stopping criterion (in our case a prefixed number of iterations) is met. This algorithm can be seen in Figure 1

7 Some notes on efficiency

Estimating the parameters as discussed above entails high computational costs: computing $p_T(\bar{y} \mid \bar{x})$ requires $\mathcal{O}(mn)$ arithmetic operations involving the values of $p_T(\bar{y}_i^j \mid \bar{x}_k^l)$ for every possible value of i, j, k and l, which are $\mathcal{O}(m^2n^2)$. This results in a global cost of $\mathcal{O}(m^3n^3)$. On the other hand, computing $\frac{\partial p_T}{\partial P}$ costs as much as computing p_T. So it is interesting to keep the number of computed derivatives low.

7.1 Reduction of the parameters to train

In the experiments we have followed some heuristics in order not to reestimate certain parameters:

- The values of M_I —and, consequently, of M_M— for lengths higher than a threshold are assumed to be 0 and therefore there is no need to estimate them.
- As a consequence, the values of ε for lengths above the same threshold, need not be reestimated.
- The values of t for pairs of words with counts under a certain threshold are not reestimated.

Furthermore, during the computation of counts, the recursion is cut on those substring pairs where the value of the probability for the translation is very small.

7.2 Efficient computation of model 1

Other source of optimization is the realization that for computing $p_T(\bar{y} \mid \bar{x})$, it is necessary to compute the value of p_I for each possible pair $(\bar{x}_{ib}^{ie}, \bar{y}_{ob}^{oe})$ (where ib, ie, ob and oe stand for *input begin*, *input end*, *output begin* and *output end*, respectively). Fortunately, it is possible to accelerate this computations. First, define:

$$
\begin{aligned}
I(ib, ie, ob, oe) &= \frac{p_I(\bar{x}_{ib}^{ie}, \bar{y}_{ob}^{oe})}{\varepsilon(ie - ib + 1, oe - ob + 1)} \\
&= \frac{1}{(ie - ib + 1)^{oe-ob+1}} \prod_{j=ob}^{oe} \sum_{i=ib}^{ie} t(\bar{y}_j \mid \bar{x}_i).
\end{aligned}
$$

Now let

$$
S(ib, ie, j) = \sum_{i=ib}^{ie} t(\bar{y}_j \mid \bar{x}_i).
$$

```
Algorithm Maximum likelihood estimation
    give initial values to the parameters;
    repeat
        initialize the counts to 0;
        for each $(\bar{x}, \bar{y}) \in \mathcal{M}$ do
            compute $p_T(\bar{y} \mid \bar{x})$;
            for each parameter $P$ involved in the alignment of $(\bar{x}, \bar{y})$ do
                $\mathcal{C}_P := \mathcal{C}_P + \frac{P}{p_T(\bar{y} \mid \bar{x})} \frac{\partial\, p_T(\bar{y} \mid \bar{x})}{\partial\, P}$;
            endfor
        endfor
        for each parameter $P$ do
            reestimate $P$ using (4) or (6);
        endfor
    until the stopping criterion is met;
End Maximum likelihood estimation
```

Figure 1: Algorithm for maximum likelihood estimation of the parameters of MAR

This leads to

$$I(ib, ie, ob, oe) = S(ib, ie, ob),$$

if $ob = oe$, and to

$$I(ib, ie, ob, oe) = \frac{I(ib, ie, ob, oe - 1) S(ib, ie, ob)}{(ie - ib + 1)},$$

if $ob \neq oe$.

So we can compute all values of I with the algorithm in Figure 2.

7.3 Splitting the corpora

Another way of reducing the costs of training has been the use of a heuristic to split long sentences into smaller parts with a length less than l words.

Suppose we are to split sentences $\bar{x}$ and $\bar{y}$. We begin by aligning each word in $\bar{y}$ to a word in $\bar{x}$. Then, a score and a translation is assigned to each substring $\bar{x}_i^j$ with a length below l. The translation is produced by looking for the substring of $\bar{y}$ which has a length below l and which has the largest number of words aligned to positions between i and j. The pair so obtained is given a score equal to sum of: (a) the square of the length of $\bar{x}_i^j$; (b) the square of the number of words in the output aligned to the input; and (c) minus ten times the sum of the square of the number of words aligned to a nonempty position out of $\bar{x}_i^j$ and the number of words outside the segment chosen that are aligned to $\bar{x}_i^j$.

After the segments of $\bar{x}$ are so scored, the partition of $\bar{x}$ that maximizes the sum of scores is computed by dynamic programming.

8 Translating the test sentences

The MAR model can be used to obtain adequate bilingual templates which can be used to translate new test sentences using an appropriate template-based translation system. Here we have adopted the `pharaoh` program (Koehn, 2004).

8.1 Finding the templates

The parameters of the MAR were trained using the algorithm above: first ten IBM model 1 iterations were used for giving initial values to the dictionary probabilities and then five more iterations for retraining the dictionary together with the rest of the parameters.

The alignment of a pair has the form of a tree similar to the one in Figure 3 (this is one of the sentences from the Spanish-English part of the training corpus). Each interior node has two children corresponding to the translation of the two parts in which the input sentence is divided. The leaves of the tree correspond to those segments that were translated by model 1. The templates generated were those defined by the leaves. Further templates were obtained by interpreting each pair of words in the dictionary as a template.

```
Algorithm all IBM
    for $ob := 1$ to $|\bar{y}|$ do
        for $oe := ob$ to $|\bar{y}|$ do
            for $ib := 1$ to $|\bar{x}|$ do
                $S := 0$;
                for $ie := ib$ to $|\bar{x}|$ do
                    $S := S + t(y_{oe} \mid x_{ie})$;
                    $I(ib, ie, ob, oe) := \begin{cases} S/(ie - ib + 1) & \text{if } ob = oe, \\ I(ib, ie, ob, oe - 1) \times S/(ie - ib + 1) & \text{otherwise;} \end{cases}$
End all IBM
```

Figure 2: Efficient computation of different values of IBM's model 1.

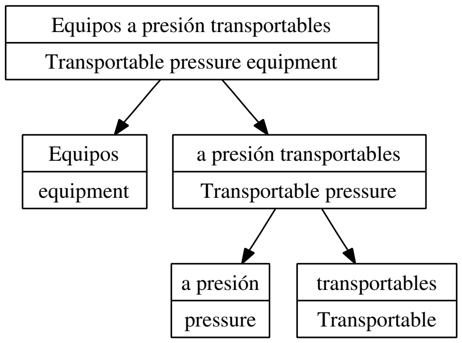

Figure 3: A sample alignment represented as a tree.

Each template was assigned four weights[1] in order to use the pharaoh program. For the templates obtained from the alignments, the first weight was the probability assigned to it by MAR, the second weight was the count for the template, i.e., the number of times that template was found in the corpus, the third weight was the normalized count, i.e., the number of times the template appeared in the corpus divided by the number of times the input part was present in the corpus, finally, the fourth weight was a small constant (10^{-30}). The intention of this last weight was to ease the combination with the templates from the dictionary. For these, the first three weights were assigned the same small constant and the fourth was the probability of the translation of the pair obtained from the stochastic dictionary. This weighting schema allowed to separate the influence of the dictionary in smoothing the templates.

[1]They should have been probabilities, but in two of the cases there was no normalization and in one they were even greater than one!

Table 1: Statistics of the training corpora. The languages are German (De), English (En), Spanish (Es), Finnish (Fi) and French (Fr).

Languages	Sentences	Words (input/output)
De-En	751 088	15 257 871 / 16 052 702
Es-En	730 740	15 725 136 / 15 222 505
Fi-En	716 960	11 318 863 / 15 493 334
Fr-En	688 031	15 599 184 / 13 808 505

9 Experiments

In order to test the model, we have decided to participate in the shared task for this workshop.

9.1 The task

The aim of the task was to translate a set of 2,000 sentences from German, Spanish, Finnish and French into English. Those sentences were extracted from the Europarl corpus (Koehn, Unpublished). As training material, four different corpora were provided, one for each language pair, comprising around 700 000 sentence pairs each. Some details about these corpora can be seen in Table 1. An automatic alignment for each corpus was also provided.

The original sentence pairs were splitted using the techniques discussed in section 7.3. The total number of sentences after the split is presented in Table 2. Two different alignments were used: (a) the one provided in the definition of the task and (b) one obtained using GIZA++ (Och and Ney, 2003) to train an IBM's model 4. As it can be seen, the number of parts is very similar in both cases. The

Table 2: Number of training pairs after splitting to a maximum length of ten. "Provided" refers to the alignment provided in the task, "GIZA++" to those obtained with GIZA++.

	Sentence pairs	
Languages	Provided	GIZA++
De-En	2 351 121	2 282 316
Es-En	2 160 039	2 137 301
Fi-En	2 099 634	2 017 130
Fr-En	2 112 931	2 080 200

Table 3: Number of templates for each language pair: "Alignment" shows the number of templates derived from the alignments; "dictionary", those obtained from the dictionary; and "total" is the sum.

(a) Using the alignments provided with the task.

Lang.	Alignment	Dictionary	Total
De-En	2 660 745	1 840 582	4 501 327
Es-En	2 241 344	1 385 086	3 626 430
Fi-En	2 830 433	2 852 583	5 683 016
Fr-En	2 178 890	1 222 266	3 401 156

(b) Using GIZA++.

Lang.	Alignment	Dictionary	Total
De-En	2 672 079	1 796 887	4 468 966
Es-En	2 220 533	1 350 526	3 571 059
Fi-En	2 823 769	2 769 929	5 593 698
Fr-En	2 140 041	1 181 990	3 322 031

number of pairs after splitting is roughly three times the original.

Templates were extracted as described in section 8.1. The number of templates we obtained can be seen in Table 3. Again, the influence of the type of alignment was small. Except for Finnish, the number of dictionary templates was roughly two thirds of the templates extracted from the alignments.

9.2 Obtaining the translations

Once the templates were obtained, the development corpora were used to search for adequate values of the weights that `pharaoh` uses for each template (these are the weights passed to option `weight-t`, the other weights were not changed as an initial exploration seemed to indicate that they had little impact). As expected, the best weights differed between language pairs. The values can be seen in table 4.

It is interesting to note that the probabilities assigned by the model to the templates seemed to be better not taken into account. The most important feature was the counts of the templates, which sometimes were helped by the use of the dictionary, although that effect was small. Normalization of counts also had little impact.

Table 4: Best weights for each language pair. The columns are for the probability given by the model, the counts of the templates, the normalized counts and the weight given to the dictionary.

(a) Using the alignments provided with the task.

Languages	Model	Count	Norm	Dict
De-En	0.0	3.0	0.0	0.3
Es-En	0.0	2.9	0.0	0.4
Fi-En	0.0	7.0	0.0	0.0
Fr-En	0.0	7.0	1.0	1.0

(b) Using GIZA++.

Languages	Model	Count	Norm	Dict
De-En	0.0	3.0	0.0	0.0
Es-En	0.0	2.9	0.0	0.4
Fi-En	0.0	3.0	1.5	0.0
Fr-En	0.0	3.0	1.0	0.4

Table 5: BLEU scores of the translations.

	BLEU	
Languages	Provided	GIZA++
De-En	18.08	18.89
Es-En	21.65	21.48
Fi-En	13.31	13.79
Fr-En	21.25	19.86

10 Results and discussion

The results over the test sets can be seen in Table 5. It can be seen that, except for French, the influence of the initial alignment is very small. Also, the best results are obtained for Spanish and French, which are more similar to English that German or Finnish.

There are still many open questions that deserve more experimentation. The first is the influence of the split of the original corpora. Although the similarity of results seem to indicate that it has little influence, this has to be tested. Two more relevant aspects are whether the weighting schema is the best for the decoder. In particular, it is surprising that the normalization of counts had so little effect.

Finally, the average number of words per template is below two, which probably is too low. It is interesting to find alternate ways of obtaining the templates, for instance using internal nodes up to a given height or covering portions of the sentences up to a predefined number of words.

11 Conclusions

A new translation model has been presented. This model produces translations in a recursive way: the input sentence is divided in two parts, each is translated using the same procedure recursively and the translations are concatenated. The model has been used for finding the templates in a large vocabulary translation task. This involved using several heuristics to improve training time, including a method for splitting the input before training the models. Finally, the influence of using a stochastic dictionary together with the templates as a means of smoothing has been explored.

References

Leonard E. Baum and J. A. Eagon. 1967. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73:360–363.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, June.

Ido Dagan, Kenneth W. Church, and William A. Gale. 1993. Robust bilingual word alignment for machine aided translation. In *Proceedings of the Workshop on Very Large Corpora*, Columbus, Ohio (USA). ACL.

Yonggang Deng, Shankar Kumar, and William Byrne. 2004. Bitext chunk alignment for statistical machine translation. Research Note 50, CLSP Johns Hopkins University, April.

P. S. Gopalakrishnan, Dimitri Kanevsky, Arthur Nádas, and David Nahamoo. 1991. An inequality for rational functions with applications to some statistical problems. *IEEE Transactions on Information Theory*, 37(1):107–113, January.

Philipp Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124.

Philipp Koehn. Unpublished. Europarl: A multilingual corpus for evaluation of machine translation. Draft.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Joseph Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449, December.

Enrique Vidal, Roberto Pieraccini, and Esther Levin. 1993. Learning associations between grammars: A new approach to natural language understanding. In *Proceedings of the EuroSpeech'93*, pages 1187–1190, Berlin (Germany).

Juan Miguel Vilar Torres. 1998. *Aprendizaje de Traductores Subsecuenciales para su empleo en tareas de dominio restringido*. Ph.D. thesis, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Valencia (Spain). (in Spanish).

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the COLING'96*, pages 836–841, Copenhagen (Denmark), August.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 144–151, Sapporo (Japan), July. Association for Computational Lingustics.