Mixtures of IBM Model 2*

Jorge Civera, Alfons Juan

Departamento de Sistemas Informáticos y Computación.
Universitat Politècnica de València (Spain)

{jcivera,ajuan}@dsic.upv.es

Abstract

Mixture modelling is a standard pattern classification technique. However, in statistical machine translation, the use of mixture modelling is still unexplored. Two main advantages of the mixture approach are first, its flexibility to find an appropriate tradeoff between model complexity and the amount of training data available and second, its capability to learn specific probability distributions that better fit subsets of the training dataset. This latter advantage is even more important in statistical machine translation, since it is well known that most of the current translation models proposed have limited application to restricted semantic domains. In this paper, we describe a mixture extension of the IBM model 2 along with the maximum likelihood estimation of its parameters through the EM algorithm and a dynamic-programming decoding algorithm for this mixture model. Preliminary experiments carried out on the TOURIST task show that the mixture extension conveys a decrease in word-error rate of up to 15%.

1 Introduction

Mixture modelling is a popular approach for density estimation in many scientific areas (McLachlan & Peel, 2000). On the one hand, mixtures are flexible enough for finding an appropriate tradeoff between model complexity and the amount of training data available. Usually, model complexity is controlled by varying the number of mixture components while keeping the same parametric form for all components. On the other hand, maximum likelihood estimation of mixture parameters can be reliably accomplished by the well-known *Expectation-Maximisation (EM)* algorithm.

The application of mixture modelling to translation models is still an unexplored field. One of the most interesting properties of mixture modelling is its capability to learn a specific probability distribution in a multimodal dataset that better explains the general data generation process. In machine translation, these multimodal datasets are not an exception, but the general case. Indeed, it is easy to find corpora from which several topics could be drawn. These topics usually define sets of topic-specific lexicons that need to be translated taking into the sematic context in which they are found. This semantic ambiguity problem could be overcome by learning topic-dependent translation models that capture together the semantic context and the translation process.

In this paper, we describe a mixture extension of the IBM model 2 (Brown et al., 1990) along with ts corresponding EM parameter estimation. Furthermore, it was necessary to develop a specific search algorithm that finds the best translation for a given source sentence as defined by the IBM2 mixture model. A dynamic-programming search algorithm was implemented for this purpose as a mixture extension of that presented in (García-Varea, Casacuberta, & Ney, 1998; García-Varea & Casacuberta, 2001).

This paper is structured as follows. Next section reviews the IBM Model 2. Section 3

*This work has been supported by the Spanish project TIC2003-08681-C02 and the Ministerio de Educación. y Ciencia.

describes the IBM2 mixture model along with its EM parameter estimation. In Section 4, a dynamic-programming decoding algorithm for the IBM2 mixture model is presented. Finally, Section 5 will be devoted to experimental results and Section 6 will discuss some conclusions and future work.

2 IBM alignment model 2

Let (x, y) be a pair of *input-output* sentences; i.e. x is a sentence in a certain *input (source)* language and y is its corresponding translation in a different *output (target)* language. Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output vocabularies, respectively. The *IBM aligment models (1 to 5)* are parametric models for the *translation* probability $P(x\,|\,y)$; i.e., the probability that x is the source sentence from which we get a given translation y.

Without loss of generality, the IBM alignment models assume that each source word is *connected to exactly one* target word. Also, it is assumed that the target sentence has an initial "null" word to which source words with no direct translation are connected. Formally, a hidden variable $a = a_1 a_2 \cdots a_{|x|}$ is introduced to reveal, for each source word position j, the target word position $a_j \in \{0, 1, \ldots, |y|\}$ to which it is connected. Thus,

$$P(x\,|\,y) = \sum_{a \in \mathcal{A}(x,y)} P(x, a\,|\,y) \qquad (1)$$

where $\mathcal{A}(x, y)$ denotes the set of all possible alignments between x and y. The *alignment-completed* probability $P(x, a\,|\,y)$ can be decomposed in terms of individual, source position-dependent probabilities as:

$$P(x, a\,|\,y) = \prod_{j=1}^{|x|} P(x_j, a_j\,|\,a_1^{j-1}, x_1^{j-1}, y) \quad (2)$$

In the case of the IBM alignment model 2 (IBM2), it is assumed that a_j only depends on j and $|y|$, and that x_j only depends on the target word to which it is connected, y_{a_j}. Hence,

$$P(x_j, a_j | a_1^{j-1}, x_1^{j-1}, y) \stackrel{\triangle}{=} p(a_j | j, |y|)\, p(x_j | y_{a_j}) \qquad (3)$$

and the set of unknown parameters comprises the *alignment* parameters, $p(i|j, |y|)$, for all $i \in \{0, 1, \ldots, |y|\}$, $j \in \{1, \ldots, |x|\}$ and $|y|$; and the *statistical dictionary*, $p(v|w)$, for all $v \in \mathcal{X}$ and $w \in \mathcal{Y}$. Note that the alignment parameters defined here are slightly different from those defined in the original parameterisation (Brown et al., 1990), which also depend on $|x|$, $p(i|j, |x|, |y|)$.

Putting (1), (2) and (3) together, we may write the IBM2 model as follows:

$$p(x\,|\,y) = \sum_a \prod_{j=1}^{|x|} p(a_j|j, |y|)\, p(x_j|y_{a_j}) \quad (4)$$

$$= \prod_{j=1}^{|x|} \sum_{i=0}^{|y|} p(i|j, |y|)\, p(x_j|y_i) \qquad (5)$$

by some straightforward manipulations.

It is not difficult to derive an EM algorithm to perform maximum likelihood estimation of these parameters with respect to a collection of training samples $(X, Y) = \{(x_1, y_1), \ldots, (x_N, y_N)\}$. The *(incomplete)* log-likelihood function is:

$$L(\boldsymbol{\Theta}) = \sum_{n=1}^{N} \log \sum_{a_n} p(x_n, a_n | y_n) \qquad (6)$$

with

$$p(x_n, a_n|y_n) = \prod_{j=1}^{|x_n|} p(a_{nj}|j, |y_n|)\, p(x_{nj}|y_{na_{nj}})$$

$$= \prod_{j=1}^{|x_n|} \prod_{i=0}^{|y_n|} [p(i|j, |y_n|) p(x_{nj}|y_{ni})]^{a_{nji}} \quad (7)$$

where, for convenience, the alignment variable, $a_{nj} \in \{0, 1, \ldots, |y_n|\}$, has been rewritten as an indicator vector in (7), $a_{nj} = (a_{nj0}, \ldots, a_{nj|y_n|})$, with 1 in position a_{nji} and zeros elsewhere.

The so-called *complete* version of the log-likelihood function (6) assumes that the hidden (missing) alignments $a_1, \ldots, a_N$ are also known:

$$\mathcal{L}(\boldsymbol{\Theta}) = \sum_{n=1}^{N} \log p(x_n, a_n | y_n) \qquad (8)$$

The EM algorithm maximises (6) iteratively, through the application of two basic

steps in each iteration: the E(xpectation) step and the M(aximisation) step. At iteration k, the E step computes the expected value of (8) given the observed (incomplete) data, (X, Y), and a current maximum likehood estimate of the parameters, $\mathbf{\Theta}^k$. This reduces to the computation of the expected value of a_{nji}:

$$\begin{aligned} a^k_{nji} &= p(a_{nji} = 1 \,|\, x_n, y_n)^k \\ &= \frac{p(i \,|\, j, |y_n|)^k \, p(x_{nj} \,|\, y_{ni})^k}{\sum_{i'=0}^{|y_n|} p(i' \,|\, j, |y_n|)^k \, p(x_{nj} \,|\, y_{ni'})^k} \end{aligned} \quad (9)$$

Then, the M step finds a new estimate of $\mathbf{\Theta}$, $\mathbf{\Theta}^{k+1}$, by maximising (8), using (9) instead of the missing a_{nji}. This results in:

$$p(i|j,|y|)^{k+1} = \frac{\sum_{\substack{n=1 \\ j \le |x_n| \\ |y_n|=|y|}}^{N} a^k_{nji}}{\sum_{i'=0}^{|y|} \sum_{\substack{n=1 \\ j \le |x_n| \\ |y_n|=|y|}}^{N} a^k_{nji'}} \quad (10)$$

for all i, j and $|y|$; and

$$p(v|w)^{k+1} = \frac{\sum_{n=1}^{N} \sum_{\substack{j=1 \\ x_{nj}=v}}^{|x_n|} \sum_{\substack{i=0 \\ y_{ni}=w}}^{|y_n|} a^k_{nji}}{\sum_{v'} \sum_{n=1}^{N} \sum_{\substack{j=1 \\ x_{nj}=v'}}^{|x_n|} \sum_{\substack{i=0 \\ y_{ni}=w}}^{|y_n|} a^k_{nji}} \quad (11)$$

for all $v \in \mathcal{X}$ and $w \in \mathcal{Y}$.

An initial estimate for $\mathbf{\Theta}$, $\mathbf{\Theta}^0$, is required for the EM algorithm to start. This is usually done by using the solution given by the IBM alignment model 1, which is the particular case of IBM2 in which the alignment probabilities are uniformly distributed; i.e.,

$$p(i|j,|y|)^{k+1} = \frac{1}{|y|+1} \quad (12)$$

for all i, j and $|y|$.

3 IBM2 mixture models

A finite mixture model is a probability (density) function of the form:

$$p(z) = \sum_{c=1}^{C} p(c) \, p(z \,|\, c) \quad (13)$$

where C is the *number of mixture components* and, for each component c, $p(c) \in [0, 1]$ is its *prior or coefficient* and $p(z \,|\, c)$ is its *component-conditional probability (density) function*. It can be seen as a generative model that first selects the cth component with probability $p(c)$ and then generates z in accordance with $p(z \,|\, c)$. It is clear that finite mixture modelling allows generalisation of any given probabilistic model by simply using more than one component.

In this work, we are interested in modelling the translation probability $P(x \,|\, y)$ using a C-component, y-conditional mixture of IBM2 models:

$$p(x \,|\, y) = \sum_{c=1}^{C} p(c) \, p(x \,|\, y, c) \quad (14)$$

where

$$p(x|y,c) = \prod_{j=1}^{|x|} \sum_{i=0}^{|y|} p(i|j,|y|,c) \, p(x_j|y_i,c) \quad (15)$$

Note that we could have made $p(c)$ to depend on y in Eq. 14 but, for simplicity, this is left for future work. Also note that each component c has its own, component-conditional independent set of alignment parameters and statistical dictionary, say $\mathbf{\Theta}_c = (\{p(i|j,|y|,c)\}, \{p(v|w,c)\})$. Now, the entire set of unknown parameters, $\mathbf{\Theta}$, includes both the component coefficients $p(1), \ldots, p(C)$, and the component-conditional IBM2 model parameters $\mathbf{\Theta}_1, \ldots, \mathbf{\Theta}_C$.

It is easy to extend the EM algorithm developed in the previous section to the more general case of IBM2 mixtures. The log-likelihood function of $\mathbf{\Theta}$ with respect to N training samples is:

$$L(\mathbf{\Theta}) = \sum_{n=1}^{N} \log \sum_{z_n} \sum_{a_n} p(x_n, z_n, a_n | y_n) \quad (16)$$

where $z_n = (z_{n1}, \ldots, z_{nC})$ is an indicator vector for the component generating x_n, and

$$p(x_n, z_n, a_n | y_n) = \prod_{c=1}^{C} [p(c) \, p(x_n, a_n | y_n, c)]^{z_{nc}} \quad (17)$$

with

$$p(x_n,a_n|y_n,c)=\prod_{j=1}^{|x_n|}\prod_{i=0}^{|y_n|}[p(i|j,|y_n|,c)p(x_{nj}|y_{ni},c)]^{a_{nji}}$$

where, as in the previous section, $a_{nji} = 1$ means that the nth training pair has its source position j connected to target position i. Note that data completion in the mixture case includes the alignments $a_1, \ldots, a_N$ and the component labels $z_1, \ldots, z_N$ as well. Thus, the complete version of the log-likehood function (16) can be obtained by simply eliminating the sums over z_n and a_n.

At iteration k, the E step for the mixture case involves the computation of the expected value of z_{nc},

$$\begin{aligned} z_{nc}^k &= p(z_{nc}=1 \,|\, x_n, y_n)^k \\ &= \frac{p(c)^k \, p(x_n \,|\, y_n, c)^k}{\sum_{c'=1}^{C} p(c')^k \, p(x_n \,|\, y_n, c')^k} \end{aligned} \tag{18}$$

with

$$p(x_n|y_n,c)^k=\prod_{j=1}^{|x_n|}\sum_{i=0}^{|y_n|}p(i|j,|y_n|,c)^k p(x_{nj}|y_{ni},c)^k$$

and the expected value of $z_{nc}\, a_{nji}$,

$$\begin{aligned} (z_{nc}\, a_{nji})^k &= p(z_{nc}=1, a_{nji}=1 \,|\, x_n, y_n)^k \\ &= z_{nc}^k \, a_{njic}^k \end{aligned} \tag{19}$$

with

$$\begin{aligned} a_{njic}^k &= p(a_{nji}=1 \,|\, x_n, y_n, c)^k \\ &= \frac{p(i|j,|y_n|,c)^k \, p(x_{nj}|y_{ni},c)^k}{\sum_{i'=0}^{|y_n|} p(i'|j,|y_n|,c)^k \, p(x_{nj}|y_{ni'},c)^k} \end{aligned} \tag{20}$$

Note that (20) is just a component-conditional version of (9).

The M step now includes an updating rule for the mixture coefficients,

$$p(c)^{k+1} = \frac{1}{N}\sum_{n=1}^{N} z_{nc}^k \tag{21}$$

for all $c = 1, \ldots, C$; and component-conditional versions of (10) and (11):

$$p(i|j,|y|,c)^{k+1} = \frac{\sum_{\substack{n=1 \\ j\le|x_n| \\ |y_n|=|y|}}^{N} z_{nc}^k a_{njic}^k}{\sum_{i'=0}^{|y|}\sum_{\substack{n=1 \\ j\le|x_n| \\ |y_n|=|y|}}^{N} z_{nc}^k a_{nji'c}^k} \tag{22}$$

for all c, i, j and $|y|$; and

$$p(v|w,c)^{k+1}=\frac{\sum_{n=1}^{N}\sum_{j=1}^{|x_n|}\sum_{\substack{i=0 \\ x_{nj}=v \; y_{ni}=w}}^{|y_n|} z_{nc}^k a_{njic}^k}{\sum_{v'}\sum_{n=1}^{N}\sum_{j=1}^{|x_n|}\sum_{\substack{i=0 \\ x_{nj}=v' \; y_{ni}=w}}^{|y_n|} z_{nc}^k a_{njic}^k} \tag{23}$$

for all c, v and w.

The initialisation technique for the IBM2 model can be easily extended to the mixture case; i.e. by using a solution from a simpler mixture of IBM1 models.

4 Decoding algorithm

In statistical machine translation, the aim of the decoding algorithm is to search for a target sentence $\widehat{y}$ that maximises:

$$\widehat{y} = \operatorname*{argmax}_{y} \; p(y)\, p(x \,|\, y) \tag{24}$$

This search is known to be an NP-hard problem (Knight, 1999). However, several search algorithms have been proposed in the literature to solve this ill-posed problem efficiently: A^* (Brown et al., 1990), stack-decoding (Wang & Waibel, 1997; Al-Onaizan et al., 1999), integer-programming (Germann et al., 2001) and dynamic-programming (García-Varea & Casacuberta, 2001; Tillmann & Ney, 2003).

In (García-Varea & Casacuberta, 2001; García-Varea et al., 1998), a dynamic-programming search algorithm for the IBM2 model is proposed, along with some refinements for speedup. We have extended this algorithm to the case of IBM2 mixture models. The extension considers an extra dimension in the search trellis to store the translation score for each component in the mixture independently. To be more precise, the

translation model $p(x\,|\,y)$ in (24) is instantiated as the IBM2 mixture model (14) and $p(y)$, for the sake of simplicity in the notation, is assumed to be a bigram language model. Thus, the best translation associated to (24), given the source sentence x and the target-sentence length $|y|$, is computed as:

$$\widehat{y}_1^{|y|} = \underset{y_1^{|y|}}{\operatorname{argmax}} \left[\prod_{i=1}^{|y|} p(y_i|y_{i-1})\right] \times \times \sum_{c=1}^{C} p(c) \prod_{j=1}^{|x|} \sum_{i=0}^{|y|} p(i\,|\,j,|y|,c)\, p(x_j\,|\,y_i,c) \tag{25}$$

The expression in (25) can be reformulated in terms of two recursive functions T and Q:

$$\widehat{y}_1^{|y|} = \underset{y_1^{|y|}}{\operatorname{argmax}}\, T(y_{|y|},|y|) \times \times \sum_{c=1}^{C} p(c) \prod_{j=1}^{|x|} Q(y_{|y|},|y|,j) \tag{26}$$

The definition of the recursive functions T and Q for any partial hypothesis y_1^i, being $i = 1,\ldots,|y|$ and $y_i = w$, is:

$$T(w,i) = T(\widehat{w}(w,i),i\text{–}1) \times \times p(w|\widehat{w}(w,i)) \tag{27}$$

$$Q(w,i,j,c) = Q(\widehat{w}(w,i),i-1,j,c) + + p(i\,|\,j,|y|,c)\, p(x_j\,|\,w,c) \tag{28}$$

where the function $\widehat{w}(w,i)$ returns the best previous word w' given that w is going to appear next in the target sentence at position i:

$$\widehat{w}(w,i)) = \underset{w'}{\operatorname{argmax}} \Big(T(w'(w,i),i-1) \times \times p(w|w') \times \times \sum_{c=1}^{C} p(c) \prod_{j=1}^{|x|} \Big[Q(w',i-1,j,c) + + p(i\,|\,j,|y|,c)\, p(x_j\,|\,w,c) + + R(j,i+1,c)\Big]\Big) \tag{29}$$

being R, a function that estimates the cost of translating from position $i+1$ to the end of the target sentence,

$$R(j,i,c) = \sum_{k=i}^{|y|} p(k\,|\,j,|y|,c)\, p(x_j\,|\,\overline{y}_k,c) \tag{30}$$

where $\overline{y}_1^{|y|}$ is an estimation of the best translation for x.

The base case of the recursion for functions T and Q is:

$$T(w,1) = p(w|\$)$$
$$Q(w,1,j,c) = p(0\,|\,j,|y|,c)\, p(x_j\,|\,\emptyset,c) + + p(1\,|\,j,|y|,c)\, p(x_j\,|\,w,c) \tag{31}$$

for all w and $j = 1,\ldots,|x|$ and where $\emptyset$ represents the NULL word.

The estimation of the function R poses a problem when the target sentence is unknown. A mixture extension of the initial optimistic estimation of R, proposed in (García-Varea, 2003), can be calculated as:

$$R(j,i,c) = \sum_{k=i}^{|y|} \max_{w} p(k|j,|y|,c)\, p(x_j|w,c) \tag{32}$$

Once an initial translation has been computed, function R is reestimated using this translation and the complete search process is repeated. This way of proceeding defines an iterative refinement process that updates R in each iteration. This iterative translation process runs until convergence (function R remains the same between two consecutive iterations) or for a maximum number of rounds.

4.1 Decoding parameters

Furthermore, there are several parameters that need to be tuned in order, on the one hand not to eliminate the benefits of using more components in the IBM2 mixture model, and on the other hand to obtain results in a reasonable period of time. The following parameters were defined in order to control the trade-off between accuracy and response time:

- Number of translations W for each word in the source sentence, in order to constitute a translation bag-of-words to be considered during the search process.

  The definition of this bag-of-words is performed according to the inverse translation probability (Al-Onaizan et al., 1999) in order to select the most

probable translation for each source word:

$$p^{-1}(w \mid w_x) = \frac{p(w_x \mid w)\, p(w)}{\sum_{w'_x} p(w'_x \mid w)\, p(w)} \approx$$

$$\approx \frac{\sum_{c=1}^{C} \left[p(w_x \mid w, c)\, p(c)\right]\, p(w)}{\sum_{w'_x} \sum_{c=1}^{C} \left[p(w'_x \mid w, c)\, p(c)\right]\, p(w)} \quad (33)$$

where $p(w)$ is a unigram language model learnt on the training partition.

- Number of "zero-fertility" words WZ to be included in the translation bag-of-words. This set of words is defined as those target words that are least aligned to any source word in the training set according to the Viterbi alignment for the IBM2 mixture model. It is necessary to take those words into account since they are rarely included as possible direct translations of words in the source sentence, therefore they would not appear in a translation.

  A possible approximation to the Viterbi alignment for the IBM2 mixture model is:

$$\hat{\boldsymbol{a}} = \operatorname*{argmax}_{\boldsymbol{a}} \sum_{c=1}^{C} p(c) \times \times \prod_{j=1}^{|x|} \prod_{a_j=0}^{|y|} p(a_j \mid j, |y|, c)\, p(x_j \mid y_{a_j}, c) \quad (34)$$

$$\approx \operatorname*{argmax}_{\boldsymbol{a}} \max_{c=1,\ldots,C} p(c) \times \times \prod_{j=1}^{|x|} \prod_{a_j=0}^{|y|} p(a_j \mid j, |y|, c)\, p(x_j \mid y_{a_j}, c) \quad (35)$$

- Beam-search coefficient B to prune those hypotheses whose score was lower than the best score multiplied by this coefficient.

- Target-sentence length range L to be explored when translating a source sentence. The algorithm presented in Section 4 assumes that we know a priori the length of the translation. In practice, this is not true, so it is necessary to search for translations in a range of sentence lengths.

  The adopted solution considers a gaussian distribution over the target-sentence length depending on the source-sentence length. So, the range mentioned goes from $\overline{|y|}_{|x|} - L$ to $\overline{|y|}_{|x|} + L$, where $\overline{|y|}_{|x|}$ is the average length of the target sentence given the length of source sentence to be translated.

- Maximum number of search rounds D, that defines the number of times the same source sentence is going to be translated for a fixed target-sentence length while the function R needs to be recomputed.

The interested reader is referred to (García-Varea et al., 1998; García-Varea & Casacuberta, 2001) for further details on this decoding algorithm.

5 Experimental results

The Spanish-English TOURIST task (Amengual et al., 1996) was selected to assess the IBM2 mixture model proposed in this paper. In the following subsections, first, the TOURIST task is described. Then, the experimental setting is defined together with the evaluation error measures, and finally, the experimental results are presented.

5.1 Corpus

The Spanish-English TOURIST task (Amengual et al., 1996) is composed of sentence pairs corresponding to human-to-human communication situations at the front-desk of a hotel which were semi-automatically produced using a small seed corpus compiled by four people from travel guide booklets dealing with different topics. A corpus of $10,000$ random sentences pairs was selected for training purposes and a test partition was defined using $2,996$ random sentence pairs generated independently from the training partition. The basic statistics of this corpus are shown in Table 1.

Table 1: Basic statistics of the Spanish-English TOURIST task.

	Training Set		Test Set	
	Sp	En	Sp	En
num. sents	10.000		2.996	
avg. length	9	9	11	11
vocabulary	686	513	611	468
singletons	10	8	63	49
run. words	97K	99K	35K	36K
perplexity	-	-	-	3.95

This multimodal corpus defines an excellent test-bed to experiment with the IBM2 mixture model, since its simplicity will bring about the pros and cons of this model.

5.2 Evaluation error measures

Two error measures were proposed in order to evaluate the IBM2 mixture model:

- *Word Error Rate* (WER): the minimum number of substitution, insertion and deletion operations needed to convert the hypothesized translation into a given single reference translation.
- *BiLingual Evaluation Understudy* (BLEU): it is based on the n-grams of the hypothesized translation that occur in the reference translations. The BLEU metric ranges from 0.0 (worst score) to 1.0 (best score) (Papineni et al., 2002).

The reason to use the WER measure in the evaluation was to compare our results to previous works on the same corpus. Conversely, the BLEU score has become a reference error measure for the evaluation of translation quality in MT systems.

5.3 Experimental setting

Several experiments were carried out with the Spanish-English TOURIST task to analyse the evolution of the error rate as a function of the number of mixture components ($C \in \{1, 2, 5, 10, 20\}$).

The training process starts by iterating with the IBM1 mixture model from a random initialisation until convergence. Then, the parameters learnt in the IBM1 mixture model are transfered to the IBM2 mixture model that is also trained until convergence. This 2-step procedure favours smooth parameter learning from a simpler model to a more complex model leading to better translation results.

We are aware that IBM models are usually trained for a few iterations, instead of until convergence. This way of proceeding would lead to an interesting set of experiments, which we plan to carry out as a future work, in order to study the behaviour of this model as a function of the number of EM iterations.

The search parameters were fixed in order to not interfere in the study of the translation model itself, so that a large number of hypotheses were explored. The values of the search parameters, determined empirically, were the following:

- $W = 12$
- $WZ = 24$
- $B = 5000$
- $L = 4$
- $D = 3$

The language models used in these experiments were smooth bigrams and trigrams based on back-off with Witten-Bell discounting (Witten & Bell, 1991).

5.4 Experimental results

Figure 1 shows the evolution of the WER (left y axis) and BLEU score (right y axis), on the test partition of the TOURIST task, for an increasing number of mixture components (x axis). Each curve represents the progress of an evaluation measure, WER (W) or BLEU (B), when using a smooth bigram (2g) or trigram (3g) language model. Each plotted point is an average over values obtained from 10 randomised trials.

As a reference, the single-component version of these results are tightly correlated with those obtained in (García-Varea & Casacuberta, 2000). Although in that work, the translation training schema was different: $1^5H^5 3^5 4^5$, that is, 5 iterations for IBM1 model, 5 iterations for the HMM model, 5

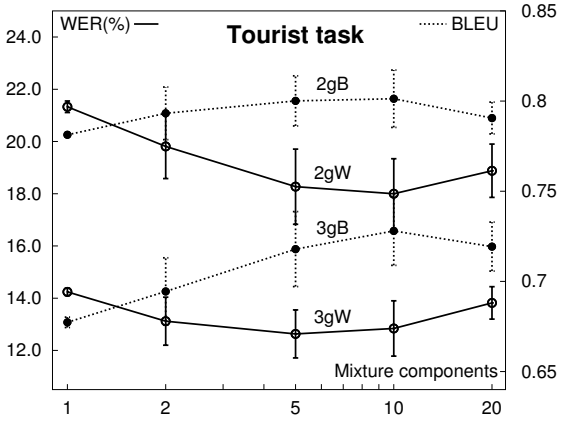

Figure 1: WER (W) and BLEU (B) curves in the test partition as a function of the number of mixture components using smooth bigram(2g) and trigram (3g) language models.

iterations for the IBM4 and 5 iterations for the IBM5. Another difference is the employment of Good-Turing discount for the language model. However, the search algorithm is based on the IBM2 model parameters.

When analysing the results in Figure 1, it is observed a systematic decrease in WER (increase in BLEU) as more components are added to the IBM2 mixture model. This positive trend reverts at different parameter settings depending on the language model we are using. In the case of bigrams that happens when the model incorporates 20 components into the mixture, while in trigrams this trend reverts when using 10 components. The reason behind this behaviour is mainly due to the fact that a trigram language model leaves less space for improvement than a simple bigram language model. So, the refinement of the translation model through the incorporation of more components produces greater benefit in a simpler bigram model, than in an already sophisticated trigram model.

A summary of baseline and best mixture results for bigram and trigram language models is shown in Table 2. These figures reflects that the IBM2 mixture model provides an average relative improvement in WER of 15% for the bigram language model and 11% for the trigram language model, with respect to the single-component IBM2 model.

However, we are aware that the results reported in this work are far from those obtained in the same corpus with state-of-the-art phrase-based models, specifically Alignment Templates (Och, 1999). However, we are mostly interested in presenting new ideas that could be extended to other translation models, like phrase-models, in order to train topic-specific translation models in large corpora.

Table 2: Baseline and best mixture results on the Spanish-English TOURIST task. The n-column indicates the n-gram order of the language model, while the C-column denotes the number of components in the IBM2 mixture model.

n	C	WER	BLEU
2	1	21.33	0.6773
	10	18.00	0.7280
3	1	14.24	0.7813
	5	12.63	0.8001

6 Conclusions and future work

In this paper a mixture extension of the IBM2 model was presented together with its maximum likelihood parameter estimation and a specific decoding algorithm.

The experiments conducted clearly indicate the benefits of the mixture approach over the single-component IBM2 model, even though these results are not competitive enough compared to those obtained by state-of-the-art phrase-based models. However, the reader should bear in mind that the work presented in this paper is a first step towards the application of mixture modelling to other translation models. Mixture modelling aims at dealing with multimodal data, which is an interesting open problem requiring further investigation in statistical machine translation.

Nevertheless, there are several challenges still ahead in the application of mixture modelling in statistical machine translation. The main problem is the increasing number of parameters to be learnt as more components are added to the model. This problem appears in the training and the decoding

process. Nonetheless, an appealing property of mixture modelling is this possibility to control the complexity of the model through the selection of an adequate number of components depending on the data available.

All in all, there are other alternative solutions to deal with the relative complexity of mixture models, both in training and decoding. The incorporation of monolingual and bilingual classes (Och, 1999) is an interesting approach to control model complexity, specifically the number of parameters in modelling component-dependent statistical dictionaries, by choosing the right number of word classes. In the decoding, there are different ways to accelerate the search process, for example, with the application of powerful heuristics as proposed in (Och, Ueffing, & Ney, 2001).

References

Al-Onaizan, Y., et al.. (1999). *Statistical Machine Translation: Final Report* (Tech. Rep.). Johns Hopkins University 1999 Summer Workshop on Language Engineering, Center for Language and Speech Processing, Baltimore, MD, USA.

Amengual, J. C., et al.. (1996). *Example-Based Understanding and Translation Systems (EuTrans)* (Final Report (part I), ESPRIT project 20268). ITI.

Brown, P. F., et al.. (1990). A Statistical Approach to Machine Translation. *Computational Linguistics*, *16*(2), 79–85.

García-Varea, I. (2003). *Traducción automática estadística: Modelos de traducción basados en máxima entropía y algoritmos de búsqueda.* Unpublished doctoral dissertation, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, Valencia, España.

García-Varea, I., & Casacuberta, F. (2000). Word categorization in statistical translation. In *Pattern recognition and applications* (pp. 212–220). IOS Press.

García-Varea, I., & Casacuberta, F. (2001). Search algorithms for statistical machine translation based on dynamic programming and pruning techniques. In *Proc. of MT Summit VIII* (pp. 115–120). Santiago de Compostela, Spain.

García-Varea, I., Casacuberta, F., & Ney, H. (1998). An iterative, DP-based search algorithm for statistical machine translation. In *Proc. of ICSLP'98* (pp. 1235–1238). Sydney, Australia.

Germann, U., et al.. (2001). Fast decoding and optimal decoding for machine translation. In *Proc. of ACL01* (pp. 228–235). Toulouse, France.

Knight, K. (1999). Decoding complexity in word-replacement translation models. *Computional Linguistics*, *25*(4), 607–615.

McLachlan, G. J., & Peel, D. (2000). *Finite Mixture Models.* Wiley.

Och, F. J. (1999). An efficient method for determining bilingual word classes. In *Proceedings of EACL'99* (pp. 71–76). Bergen, Norway.

Och, F. J., Ueffing, N., & Ney, H. (2001). An Efficient A* Search Algorithm for Statistical Machine Translation. In *Proceedings of ACL-EACL'01 Workshop on Data-Driven Machine Translation* (pp. 55–62). Toulouse, France.

Papineni, K., et al.. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL'02* (p. 311-318). Morristown, NJ, USA.

Tillmann, C., & Ney, H. (2003). Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational Linguistics*, *29*(1), 97–133.

Wang, Y.-Y., & Waibel, A. (1997). Decoding algorithm in statistical translation. In *Proc. of ACL/EACL'97* (pp. 366–372). Madrid, Spain.

Witten, I. H., & Bell, T. C. (1991). The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Trans. on Information Theory*, *37*, 1085–1094.