Exploring the Use of Target-Language Information to Train the Part-of-Speech Tagger of Machine Translation Systems*

Felipe Sánchez-Martínez, Juan Antonio Pérez-Ortiz, and Mikel L. Forcada

Departament de Llenguatges i Sistemes Informàtics
Universitat d'Alacant
E-03071 Alacant, Spain

Abstract. When automatically translating between related languages, one of the main sources of machine translation errors is the incorrect resolution of part-of-speech (PoS) ambiguities. Hidden Markov models (HMM) are the standard statistical approach to try to properly resolve such ambiguities. The usual training algorithms collect statistics from source-language texts in order to adjust the parameters of the HMM, but if the HMM is to be embedded in a machine translation system, target-language information may also prove valuable. We study how to use a target-language model (in addition to source-language texts) to improve the tagging and translation performance of a statistical PoS tagger of an otherwise rule-based, shallow-transfer machine translation engine, although other architectures may be considered as well. The method may also be used to customize the machine translation engine to a particular target language, text type, or subject, or to statistically "retune" it after introducing new transfer rules.

1 Introduction

One of the main sources of errors in machine translation (MT) systems between related languages is the incorrect resolution of part-of-speech (PoS) ambiguities. Hidden Markov models (HMMs) [9] are the standard statistical approach [3] to automatic PoS tagging. Typically the training of this kind of taggers has been carried out from source-language (SL) untagged corpora (see below) using the Baum-Welch algorithm [9].

But target-language (TL) texts may also be taken into account in order to improve the performance of these PoS taggers, specially as to the resulting translation quality, an aspect not faced by training algorithms which use information from SL only. We propose a training method for HMMs which considers the likelihood in the TL of the translation of each of the multiple disambiguations

* Work funded by the Spanish Government through grants TIC2003-08681-C02-01 and BES-2004-4711. We thank Rafael C. Carrasco for useful comments on this work. We also thank Geoffrey Sampson (University of Sussex, England) for his Simple Good-Turing implementation.

of a source text which can be produced depending on how its PoS ambiguity is resolved. To achieve this goal, these steps are followed: first, the SL text is segmented; then, the set of all possible disambiguations for each segment is generated; after that, each disambiguation is translated into TL; next, a TL statistical model is used to compute the likelihood of each translated disambiguation of the segment; and, finally, these likelihoods are used to adjust the parameters of the SL HMM: the higher the likelihood, the higher the probability of the original SL tag sequence in the model being trained. Rules for text segmentation must be carefully chosen so that the resulting segments are treated independently by the rest of the modules in the MT system.

One of the main obstacles to overcome is the presence of *free rides*, that is, an ambiguous SL word which is translated into the same TL word for every possible disambiguation; therefore, the ambiguity remains intact in the TL and no TL information can be used for disambiguation purposes. This is specially harmful in the case of related languages, where free rides are very common.

Most current MT systems follow the *indirect* or *transfer* approach [5, ch. 4]: SL text is analysed and converted into an intermediate representation which becomes the basis for generating the corresponding TL text. Analysis modules usually include a PoS tagger for the SL. Our method for training PoS taggers may be applied, in principle, to any variant of an indirect architecture which uses or may use a HMM-based PoS tagger. In particular, a MT system using a classical *morphological transfer* architecture will be considered in the experiments.

We will refer to a text as *unambiguously tagged* or just *tagged* when each occurrence of each word (ambiguous or not) has been assigned the correct PoS tag. An *ambiguously tagged text* or *untagged* text corpus is one in which all words are assigned (using a morphological analyser) the set of possible PoS tags independently of context; in this case, ambiguous and unknown words would receive more than one PoS tag (unknown words, that is, words not found in the lexicon, are usually assigned the set of *open* categories, that is, categories which are likely to grow by new words of the language: nouns, verbs, adjectives, adverbs and proper nouns). Words receiving the same set of PoS tags are said to belong to the same *ambiguity class* [3]; for example, the words *tailor* and *book* both belong to the ambiguity class {noun, verb}.

The paper is organized as follows. Section 2 presents the basics of HMM use in disambiguation tasks and discusses existing methods for PoS tagger training; section 3 describes our proposal for HMM training and the TL model used in this paper; section 4 introduces the translation engine and shows the main results of the experiments; finally, in sections 5 and 6 we discuss the results and outline future work to be done.

2 Part-of-Speech Tagging

When a HMM is used for lexical disambiguation purposes in the *ambiguity class mode* (in which each input word is replaced by its corresponding ambiguity class) each HMM state is made to correspond to a different PoS tag and the set of observable items consists of all possible ambiguity classes [3]. Building a PoS tagger based on HMMs for the SL in a MT system usually implies:

1. *Designing or adopting a reduced tagset* (set of PoS) which groups the finer tags delivered by the morphological analyser into a smaller set of coarse tags adequate to the translation task (for example, singular feminine nouns and plural feminine nouns may be grouped under a single "noun" tag). Additionally, the number of different lexical probabilities in the HMM is usually drastically reduced by grouping words in ambiguity classes.
2. *Estimating proper HMM parameters*, that is, finding adequate values of the parameters of the model. Existing methods may be grouped according to the kind of corpus they use as input: *supervised* methods require *unambiguously tagged texts* (see the introduction); *unsupervised* methods are able to extract information from *ambiguously tagged texts*, that is, from sequences of ambiguity classes.
   On the one hand, estimating parameters from an unambiguously tagged corpus is usually the best way to improve performance, but unambiguously tagged corpora are expensive to obtain and require costly human supervision. A supervised method counts the number of occurrences in the corpus of certain tag sequences (usually bigrams) and uses this information to determine the values of the parameters of the HMM. On the other hand, for the unsupervised approach no analytical method is known, and existing methods, such as the Baum-Welch algorithm [9], are only guaranteed to reach local (not global) maxima of the expectation.
   Some estimation methods, like those using statistics from unambiguously tagged corpus, may be used in isolation or as an initialization algorithm for further reestimation via the Baum-Welch method. In other cases, simple estimation methods are exclusively considered for initialization purposes. A good initialization can significantly improve the final performance of a Baum-Welch-trained PoS tagger, although it will not completely avoid the risk of convergence at local maxima.
   The new method presented in the following section requires only ambiguously tagged SL texts and raw TL texts (needed to compute a TL model); it may, in principle, be used as a complete training method by itself, although it may as well be considered for initialization purposes.

3 Target-Language-Based Training

This section gives mathematical details on how to train a SL HMM using information from the TL.

Let S be the whole SL corpus, s be a (possibly ambiguous) segment from S, g_i a sequence of tags resulting from one of the possible disambiguation choices in s, $\tau(g_i, s)$ the translation of g_i in the TL, and $p_{\mathrm{TL}}(\tau(g_i, s))$ the likelihood of $\tau(g_i, s)$ in some TL model. We will call each g_i a *path* since it describes a unique state path in the HMM and write $g_i \in T(s)$ to show that g_i is a possible disambiguation of the words in s. Now, the likelihood of path g_i from segment s may be estimated as:

$$p(g_i|s) = \frac{p(g_i|\tau(g_i,s))\, p_{\mathrm{TL}}(\tau(g_i,s))}{\displaystyle\sum_{g_j \in T(s)} p(g_j|\tau(g_j,s))\, p_{\mathrm{TL}}(\tau(g_j,s))} \tag{1}$$

where the term $p(g_i|\tau(g_i,s))$ is the conditional probability of g_i given translation $\tau(g_i,s)$. That is, the likelihood of path g_i in source segment s is made proportional to the TL likelihood of its translation $\tau(g_i,s)$, but needs to be corrected by a weight $p(g_i|\tau(g_i,s))$, because more than one g_i may contribute to the same $\tau(g_i,s)$.

The fact that more than one path in segment s, say g_i and g_j, can produce the same translation in TL (that is, $\tau(g_i,s) = \tau(g_j,s)$ with $i \neq j$) does not imply that $p(g_i|\tau(g_i,s)) = p(g_j|\tau(g_j,s))$. Indeed, the real probabilities of paths are in principle unknown (note that their computation is the main goal of the training method). In the absence of such information, the contributions of each path will be approximated in this paper to be equally likely:

$$p(g_i|\tau(g_i,s)) = \frac{1}{\mathrm{card}\left(\{g_j \in T(s) : \tau(g_j,s) = \tau(g_i,s)\}\right)} \tag{2}$$

Now, we describe how to obtain the parameters of the HMM from the estimated likelihood of each path in each segment, $p(g_i|s)$, which will be treated as a fractional count. An estimate of tag pair occurrence frequency based on $p(g_i|s)$ is:

$$\tilde{n}(\gamma_i\gamma_j) \cong \sum_{s \in S} \sum_{g_i \in T(s)} C_{s,g_i}(\gamma_i,\gamma_j)\, p(g_i|s) \tag{3}$$

where $C_{s,g_i}(\gamma_i,\gamma_j)$ is the number of times tag γ_i is followed by tag γ_j in path g_i of segment s. Therefore, the HMM parameter $a_{\gamma_i\gamma_j}$ corresponding to the transition probability from the state associated with tag γ_i to the state associated with tag γ_j [9,3] can be computed as follows:

$$a_{\gamma_i\gamma_j} = \frac{\tilde{n}(\gamma_i\gamma_j)}{\sum_{\gamma_k \in \Gamma} \tilde{n}(\gamma_i\gamma_k)} \tag{4}$$

where Γ is the tagset, that is, the set of all PoS tags.

In order to calculate emission probabilities, the number of times an ambiguity class is emitted by a given tag is approximated by means of:

$$\tilde{n}(\sigma,\gamma) \cong \sum_{s \in S} \sum_{g_i \in T(s)} C_{s,g_i}(\sigma,\gamma)\, p(g_i|s) \tag{5}$$

where $C_{s,g_i}(\sigma,\gamma)$ is the number of times ambiguity class σ is emitted by tag γ in path g_i of segment s. Therefore, the HMM parameter $b_{\gamma_i\sigma}$ corresponding to the emission probability of ambiguity class σ from the state associated with γ_i is computed as:

$$b_{\gamma_i\sigma} = \frac{\tilde{n}(\sigma,\gamma_i)}{\sum_{\sigma' : \gamma_i \in \sigma'} \tilde{n}(\sigma',\gamma_i)} \tag{6}$$

Source language	Target language	$p(g_i \vert s)$
$s \equiv$ y la para si $\{$CNJ$\}$ $\left\{\begin{matrix}\text{ART}\\\text{PRN}\end{matrix}\right\}$ $\left\{\begin{matrix}\text{VB}\\\text{PR}\end{matrix}\right\}$ $\{$CNJ$\}$	$\tau(g_1, s) \equiv$ i la per a si $g_1 \equiv$ CNJ ART PR CNJ	0.0001
	$\tau(g_2, s) \equiv$ i la para si $g_2 \equiv$ CNJ ART VB CNJ	0.4999
	$\tau(g_3, s) \equiv$ i la per a si $g_3 \equiv$ CNJ PRN PR CNJ	0.0001
	$\tau(g_4, s) \equiv$ i la para si $g_4 \equiv$ CNJ PRN VB CNJ	0.4999

Fig. 1. Example of an ambiguous SL (Spanish) text segment, paths and translations (into Catalan) resulting from each possible disambiguation, and normalized estimated likelihood for each path translation. The second source-language word (*la*) is a free ride, as can be observed in the corresponding translation into target language.

Notice that when training from unambiguously tagged texts the expressions used to compute transition and emission probabilities are analogous to previous equations, but in this case $p(g_i|s) = 1$ in (3) and (5) as only one path is possible in a tagged corpus segment; therefore, (3) and (5) would not be approximate anymore, but exact. Figure 1 shows an example of the application of the method.

SL text segmentation must be carefully designed so that two words which get joint treatment in some stage of processing of the MT system are not associated to different segments. This would result in incorrect sequences in TL (for example, if two words involved in a word reordering rule are assigned to different segments) and, as a consequence of that, in wrong likelihood estimations. In general, HMMs can be trained by breaking the corpus into segments whose first and last word are unambiguous, since unambiguous words reveal or *unhide* the hidden state of the HMM [3, sect. 3.4]. Adequate strategies for ensuring segment independence depend on the particular translation system (we will describe in section 4 the strategy used in our experiments).

3.1 Target-Language Model

A classical trigram model of TL surface forms (SF, lexical units as they appear in original texts) is considered for this paper, although it may be worth studying some other language models. The trigram model is obtained in an unsupervised manner from a 1 822 067-word TL corpus taken from Catalan newspapers.

In order to avoid unseen trigrams to give zero probability for every text segment containing them, probabilities are smoothed via a form of deleted interpolation [6]. The smoothed trigram probabilities consist of a linear combination of trigram, bigram and unigram probabilities, and the successive linear abstraction approximation [1] is used to compute the corresponding coefficients. Since in the case of unseen words the resulting smoothed trigram probability is still zero, unigram probabilities are smoothed as well using the Good-Turing method [4].

When evaluating path likelihoods, if text segmentation is correctly performed so that segments are independent (as already mentioned), a good estimate of

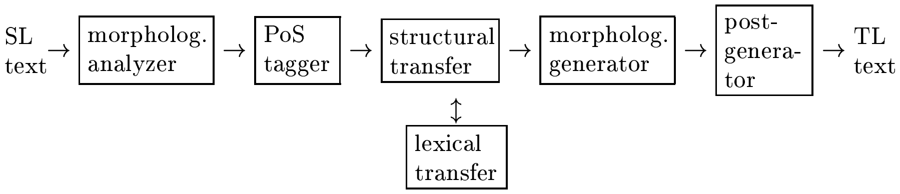

Fig. 2. Main modules of the transfer machine translation system (see section 4.1) used in the experiments.

trigram probabilities for a given path can be performed considering all possible translations of the two words preceding the current segment and the two first words of the following one. This local approach can be safely used because a complete trigram likelihood evaluation for the whole corpus would multiply the likelihood by the same factor[1] for every possible path of the segment and, therefore, it would not affect the normalized likelihood estimated for each path of a segment in eq. (1).

Finally, notice that computing likelihoods as trigram probability products causes (as in most statistical MT approaches) shorter translations to receive higher scores than larger ones.

4 Experiments

4.1 Machine Translation Engine

Since our training algorithm assumes the existence of a MT system (most likely, the system in which the resulting PoS tagger will be embedded) in order to produce texts from which statistics about TL will be collected, we briefly introduce the system used in the experiments, although almost any other architecture (with a HMM-based PoS tagger) may also be suitable for the algorithm.

We used the publicly accessible Spanish–Catalan (two related languages) MT system interNOSTRUM [2], which basically follows the (morphological) transfer architecture shown in figure 2:

- A *morphological analyser* tokenizes the text in surface forms (SF) and delivers, for each SF, one or more lexical forms (LF) consisting of *lemma*, *lexical category* and morphological inflection information.
- A *PoS tagger* (categorial disambiguator) chooses, using a hidden Markov model (HMM), one of the LFs corresponding to an ambiguous SF.
- A *lexical transfer* module reads each SL LF and delivers the corresponding TL LF.

[1] This factor results from the contribution of trigrams which do not include words in the current segment.

– A *structural transfer* module (parallel to the lexical transfer) uses a finite-state chunker to detect patterns of LFs which need to be processed for word reordering, agreement, etc.
– A *morphological generator* delivers a TL SF for each TL LF, by suitably inflecting it.
– A *postgenerator* performs some orthographical operations such as contractions.

4.2 Text Segmentation

An adequate strategy for SL text segmentation is necessary. Besides the general rules mentioned in section 3, in our setup it must be ensured that all words in every pattern transformed by the structural transfer belong to the same segment.

The strategy followed in this paper is segmenting at nonambiguous words whose PoS tag is not present in any structural transfer rule or at nonambiguous words appearing in rules not applicable in the current context. In addition, an exception is being taken into account: no segmentation is performed at words which start a multiword unit that could be processed by the postgenerator (for example, *de* followed by *los*, which usually translates as *dels* in Catalan). Unknown words are also treated as segmentation points, since the *lexical transfer* has no bilingual information for them and no *structural transfer* rule is activated at all.

4.3 Results

We study both PoS tagging performance and translation performance after training the PoS tagger for Spanish. The Spanish corpus is divided into segments as described in 4.2. For each segment, all possible translations into TL (Catalan) according to every possible combination of disambiguations are considered. The likelihoods of these translations are computed through a Catalan trigram model and then normalized and transferred to the transition matrix and emission matrix of the HMM as described in section 3. The whole process is unsupervised: no unambiguously tagged text is needed.

The tagset used by the Spanish PoS disambiguator consists of 82 coarse tags (69 single-word and 13 multi-word tags for contractions, verbs with clitic pronouns, etc.) grouping the 1 917 fine tags (386 single-word and 1 531 multiword tags) generated by the morphological analyser. The number of observed ambiguity classes is 249. In addition, a few words such as *para* (preposition or verb), *que* (conjunction or relative), *como* (preposition, relative or verb) and *más/menos* (adverb or adjective) are assigned special hidden states, and consequently special ambiguity classes, in a similar manner to that described in [8].

For comparison purposes, a HMM-based PoS tagger was trained from ambiguously tagged SL corpora following a classical approach, that is, initializing the parameters of the HMM by means of Kupiec's method [7] and using the Baum-Welch algorithm to reestimate the model; a 1 000 000-word ambiguous corpus was used for training. The resulting PoS tagger was tested after each

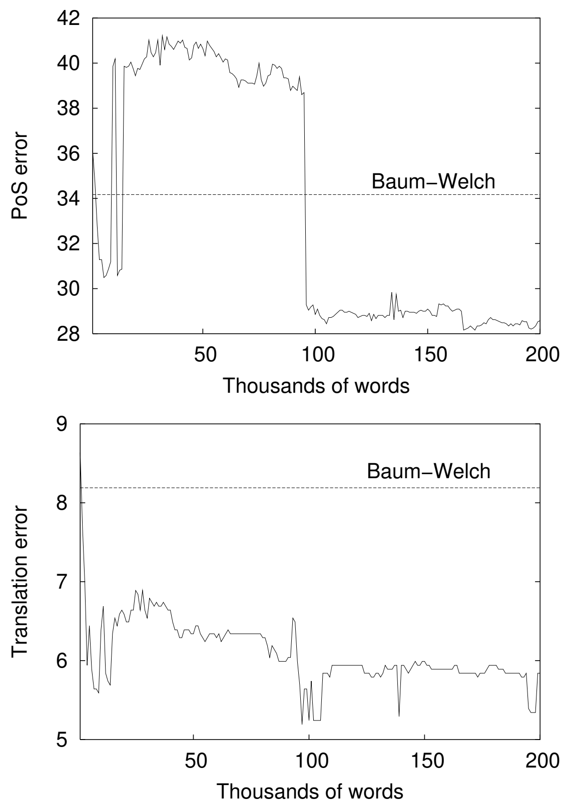

Fig. 3. PoS tagging error rate (top) and translation error rate (bottom). The Baum-Welch error rate after training with a 1 000 000-word corpus is given as well. PoS tagging error rate is expressed as the percentage of incorrect tags assigned to *ambiguous words* (including unknown words). Translation errors are expressed as the percentage of words that need to be post-edited due to mistaggings.

iteration and the one giving an error rate which did not improve in the subsequent 3 iterations was chosen for evaluation; proceeding in this way, we prevent the algorithm from stopping if a better PoS tagger can still be obtained. Moreover, another HMM was trained from an unambiguously tagged 20 000-word SL corpus and used as a reference of the best attainable results.

A set of disjoint SL corpora with 200 000 words each was considered for evaluating the proposed method and the resulting performance was recorded at every 1 000 words. Figure 3 shows the evolution of the PoS tagging error rate and the translation error rate for one representative corpus (the rest of the corpora behave in a similar way); Baum-Welch results are reported there as well. PoS tagging errors in figure 3 are expressed as the percentage of incorrect tags assigned to *ambiguous words* (including unknown words), not as the overall percentage of correct tags (over ambiguous and nonambiguous words); translation errors, however, do not consider unknown words and are expressed as the percentage of words that need to be corrected or inserted because of wrongly tagged words when post-editing the translation (in some cases, a wrongly tagged word implies correcting more than one TL word because of single words translating into multiword units or because of actions of the structural transfer or the postgenerator which would not have been performed if the word had been correctly tagged or vice versa).

The PoS tagging error is evaluated using an independent 8 031-word unambiguously tagged Spanish corpus. The percentage of ambiguous words (according to the lexicon) is 26.71% and the percentage of unknown words is 1.95%. For translation evaluation, an 8 035-word Spanish corpus and the corresponding human-corrected machine translation into Catalan are used.

The tagging error rate obtained with the PoS tagger trained from a *nonambiguous* corpus (obtained in a supervised manner) is 10.35% and the translation error rate is 2.60%; these figures can be used as a reference for the best possible results.

As can shown in figure 3, sudden oscillations causing the PoS tagging error to change around 10% occur in just one step. This behaviour is due to free rides (very common in the case of related languages like Spanish and Catalan): since free rides give the same translation regardless of disambiguation choices, the TL trigram model can not be used to distinguish among them and, consequently, paths involving free rides receive the same weight when estimating the parameters of the PoS tagger.

The most common free rides in Spanish-Catalan translation are the Spanish words *la*, *las* and *los* that belong to the same ambiguity class formed by article and proclitic pronoun. In the evaluation corpus these three words are 22.98% of the ambiguous words. Nevertheless, it may be argued that free rides should not affect the translation error rate; however, they do. This is because depending on the tag choice the *structural transfer* performs changes (gender agreement, for example) or the *postgenerator* performs contractions; in these cases, these words are not free rides, but the number of times they occur is not enough for the TL trigram model to capture their influence and make the system more stable. In the next subsection, a way of addressing this undesirable effect of free rides is explored.

4.4 Reducing the impact of free rides

The problem of free rides may be partially solved if linguistic information is used to forbid some impossible tag bigrams so that paths containing *forbidden bigrams* are ignored. The previous experiments were repeated introducing this new approach and results are discussed here.

A HMM-based PoS tagger trained via the Baum-Welch algorithm is used again for comparison purposes, but using the information from forbidden bigrams as follows: forbidden bigrams are transferred into the HMM parameters by introducing zero values in the transition matrix after Kupiec's initialization

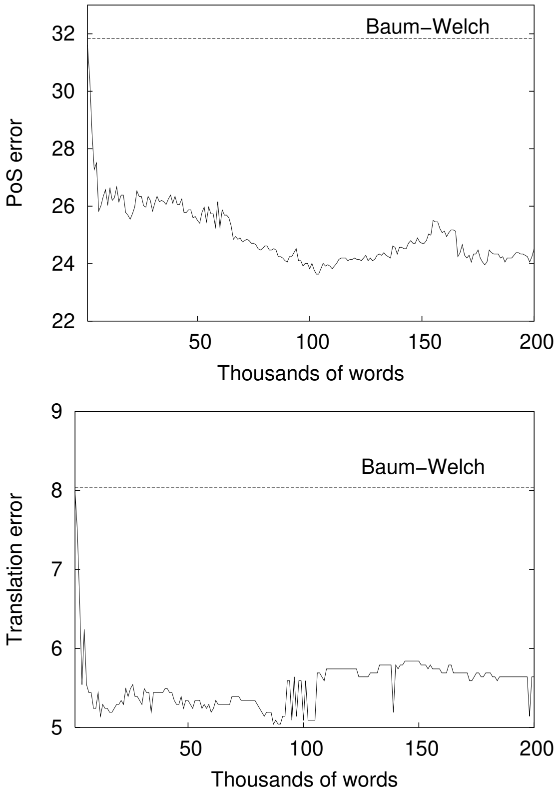

Fig. 4. PoS tagging error rate (top) and translation error rate (bottom) when rules to forbid impossible bigrams are considered (compare with figure 3). The Baum-Welch error rate after training with a 1 000 000-word corpus is given as well. PoS tagging error rate is expressed as the percentage of incorrect tags assigned to *ambiguous words* (including unknown words). Translation errors are expressed as the percentage of words that need to be corrected (post-edited) due to mistaggings.

but before training. The Baum-Welch algorithm naturally preserves these zeroes during the reestimation process.

The number of forbidden bigrams (independently collected by a linguist) is 218; the more statistically important are *article* before *verb in personal form*, *proclitic pronoun* followed by words which are not *proclitic pronouns* or *verbs in personal form*, and *article* before *proclitic pronoun*.

As can be seen in figure 4 (compare with figure 3 where the same corpus is considered), sudden oscillations decrease and the PoS tagging error rate is significantly lower than that obtained without forbidden bigrams. Smaller sudden oscillations still happen due to other free rides (for example, *que*, *conjunction* or *relative pronoun*) not in the set of forbidden bigrams but with a secondary presence in Spanish corpora.

Concerning execution time, the new method needs higher training time than the Baum-Welch algorithm because of the enormous number of translations and path likelihoods that need to be explicitly considered (remember, however, that the time necessary for processing ambiguous texts after training is independent of the algorithm being used for estimating the parameters of the HMM). With the example corpus the original algorithm takes up to 44 hours in a typical desktop computer, and around 16 hours when forbidden bigrams are introduced. The number of paths n_p and, consequently, the number of segment translations grows exponentially with segment length l and can be approximated by $n_p \approx 1.46^l$.

5 Discussion

It has been shown that training HMM-based PoS taggers using unsupervised information from TL texts is relatively easy. Moreover, both tagging and translation errors lie between those produced by classical unsupervised models using Baum-Welch estimation and those attained with a supervised solution based on nonambiguous texts.

The presence of free rides —very common when translation involves closely-related languages like Spanish and Catalan (both coming from Latin and more related than other Romance languages)— makes the algorithm behave unstably due to the kind of TL model used in the experiments (superficial form trigrams). This problem may be partially overcome by using a small amount of linguistic information (forbidden bigrams) which can be obtained in most cases much more easily than the hand-tagged corpora needed for supervised training.

Since our goal is to produce PoS taggers to be embedded in MT systems, we can focus on translation error and conclude that, despite the fact that the tagging error rate starts with higher values than the one obtained with a Baum-Welch-trained tagger, the final translation error is around 2% smaller. Our method significantly reduces the tagging error when compared with other training methods using ambiguously tagged SL texts.

The training method described in this paper produces a PoS tagger which is in tune not only with SL but also with the TL of the translation engine. This makes it suitable for training PoS taggers to be embedded in MT systems. The method may also be used to customize the MT engine to a particular text type or subject or to statistically "retune" it after introducing new transfer rules in the MT system.

6 Future Work

We plan to study other TL models. One of the most interesting alternatives in the case of a bidirectional MT system is to consider another HMM as TL model. In this way, the two HMMs used as PoS taggers in a bidirectional MT system could be trained simultaneously from scratch by initializing one of them (with Kupiec's method, for example) and using it as TL model to estimate the parameters of the other one through our training algorithm. After that, roles could be interchanged so that the last HMM being trained is now used as TL model, and so on until convergence. The process may be seen as a kind of bootstrapping: the PoS tagger for one of the languages is initialized in a simple way and both HMMs alternate either as TL model or as adjustable one.

A different line of research will study the improvement of the estimation in eq. (2). A better estimation for $p(g_i|\tau(g_i, s))$ might reduce the impact of free rides without considering linguistic information. One possible approach is to query the model currently being estimated about this probability.

Finally, another line of work will focus on time complexity reduction. On the one hand, we propose to introduce new forbidden tag bigrams to reduce even more the number of translations to be computed. On the other hand, other strategies may prove valuable like, for example, using the model being estimated to calculate approximate likelihoods which make it possible to consider only the k best paths for translation.

References

1. T. Brants and C. Samuelsson. Tagging the Teleman corpus. In *Proceedings of the 10th Nordic Conference of Computational Linguistics*, Helsinki, Finland, 1995.
2. R. Canals-Marote, et al. The Spanish-Catalan machine translation system interNOSTRUM. In *Proceedings of MT Summit VIII, Machine Translation in the Information Age*, pages 73–76, 2001.
3. D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. In *Third Conference on Applied Natural Language Processing. Association for Computational Linguistics. Proceedings of the Conference.*, pages 133–140, 1992.
4. W. Gale and G. Sampson. Good-Turing smoothing without tears. *Journal of Quantitative Linguistics*, 2(3), 1995.
5. W.J. Hutchins and H.L. Somers. *An Introduction to Machine Translation*. Academic Press, London, 1992.
6. F. Jelinek. *Statistical Methods for Speech Recognition*. The MIT Press, Cambridge, Massachusetts, 1997.
7. J. Kupiec. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6(3):225–242, 1992.
8. F. Pla and A. Molina. Improving part-of-speech tagging using lexicalized HMMs. *Journal of Natural Language Engineering*, 10(2):167–189, 2004.
9. L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.