Natural Language Engineering **10** (2): 167–189. © 2004 Cambridge University Press
DOI: 10.1017/S1351324904003353 Printed in the United Kingdom

Improving part-of-speech tagging using lexicalized HMMs

FERRAN PLA and ANTONIO MOLINA
Departament de Sistemes Informàtics i Computació,
Universitat Politècnica de València, Camí de Vera, s/n. 46020 València SPAIN
e-mail: fpla@dsci.upv.es

(*Received 12 April 2002; revised 20 August 2003*)

Abstract

We introduce a simple method to build Lexicalized Hidden Markov Models (L-HMMs) for improving the precision of part-of-speech tagging. This technique enriches the contextual Language Model taking into account a set of selected words empirically obtained. The evaluation was conducted with different lexicalization criteria on the *Penn Treebank* corpus using the TnT tagger. This lexicalization obtained about a 6% reduction of the tagging error, on an unseen data test, without reducing the efficiency of the system. We have also studied how the use of linguistic resources, such as dictionaries and morphological analyzers, improves the tagging performance. Furthermore, we have conducted an exhaustive experimental comparison that shows that *Lexicalized HMMs* yield results which are better than or similar to other state-of-the-art part-of-speech tagging approaches. Finally, we have applied *Lexicalized HMMs* to the Spanish corpus *LexEsp*.

1 Introduction

Over the last few years, inductive or corpus-based approaches have been widely used in nearly all the Natural Language Processing (NLP) tasks. The availability of linguistic resources such as corpora or dictionaries has made the application and development of these learning techniques possible. These methods have been successfully applied to solve different disambiguation problems such as part-of-speech tagging, shallow parsing or chunking, prepositional phrase attachment, etc. The main attractiveness of the corpus-based methods is that they can achieve satisfactory results without much human intervention. They can also be easily applied to different languages or can be fitted to different tasks with little effort.

One of the most well-known disambiguation problems is part-of-speech (POS) tagging. A POS tagger attempts to assign the corresponding POS tag to each word in a sentence, taking into account the context in which this word appears. Each POS tag is composed of the lexical category of the word (common noun, proper noun, adjective, etc.) and usually adds morphological information (number, gender, person, etc.). Normally, this set of POS tags has been previously defined by an expert human for a specific language.

A POS tagger has different applications. Generally, morpho-syntactic disambiguation is used as a preprocess in NLP systems. Thus, the use of a POS tagger simplifies the task of syntactic or semantic parsers because they do not have to manage ambiguous morphological sentences. It can be incorporated in NLP systems that have to deal with unrestricted text, such as information extraction, information retrieval, summarization, machine translation, etc. Also, speech recognition systems can reduce the number of parameters of the Language Model using lexical categories instead of words. All these applications can benefit from POS tagging to improve their performance in both accuracy and computational efficiency. Even though a great number of researchers have dedicated their efforts to developing or improving POS taggers in the last few years, the state-of-the-art performance of a single POS tagger (96–97%) still leaves a little room for improvement. For these reasons, there is a great interest in the development and improvement of POS taggers for different languages.

The most relevant corpus-based POS tagging approaches are based on Hidden Markov Models (Church 1988; Weischedel *et al.* 1993; Merialdo 1994; Brants 2000), transformation-based learning (Brill 1995a), memory-based learning (Daelemans *et al.* 1996), decision trees (Márquez *et al.* 2000; Magerman 1996), maximum entropy principle (Ratnaparkhi 1996), finite-state automata (Pla and Prieto 1998), etc. Moreover, some works combine the output of different taggers by means of some "voting" methods in order to improve their performance (Brill and Wu 1998; Van Halteren *et al.* 1998).

The comparison among these different approaches is difficult due to the multiple factors that must be considered: the language, the number and type of the tags, the size of the vocabulary, the ambiguity ratio, the difficulty of the test set, the size of the training and the test sets, etc. For English, most of the taggers have been evaluated on the Wall Street Journal (WSJ) corpus (Marcus *et al.* 1993). The results reported on the WSJ achieved a precision ratio of between 96% and 97%. Some comparison experiments have been conducted in order to rigorously contrast the different approaches (Brill and Wu 1998; Van Halteren *et al.* 1998; Zavrel and Daelemans 1999). Although some of these works report that HMM-based taggers achieved results which are lower than the results obtained for taggers based on other paradigms, it has recently been shown that HMMs perform better than or similar to other single taggers (Brants 2000; Pla and Molina 2001). Only combined methods slightly outperform the singles approaches (Brill and Wu 1998; Van Halteren *et al.* 2001).

This paper presents a way to enrich HMMs by incorporating lexical information into the contextual model. First, in section 2, we briefly review the most relevant corpus-based POS tagging approaches. Next, in section 3, we describe the use of HMMs in the POS tagging problem. In section 4, we introduce the technique used to lexicalize *HMMs*. In section 5, we describe the learning and tagging processes of the system used. The experimental work is presented in section 6. We study how the lexicalization improves the performance of a HMM-based tagger for both first and second-order HMM. We have applied and compared several criteria to determine the word set to specialize the HMM. These criteria are independent of the language and have been applied for both English and Spanish. For experiments, we used the *Penn*

Treebank corpus for English (Marcus *et al.* 1993) and the *LexEsp* corpus for Spanish. The results show that lexicalized models perform better than non-lexicalized models in all cases. Moreover, this method achieves results which are similar to or better than other current tagging approaches. We have also studied the influence on tagging precision when certain linguistic resources such as dictionaries or morphological analyzers are used. Lastly, in section 7, we present some conclusions.

2 POS tagging approaches

We will compare *Lexicalized HMMs* with the most popular current POS tagging approaches: Transformation-Based Error-Driven Learning (TBL), Memory-Based Learning (MBL) and Maximum Entropy (ME). Following, we will briefly describe each of these approximations.

2.1 Transformation-based error-driven learning

The main contribution to this approach was introduced by Brill (1992) and developed in following works (Brill 1993; Brill 1995b). It consists of the automatic learning of *transformation rules* in order to correct certain cases of morphological ambiguity. These rules are learned from a corpus using a set of patterns or rule-templates which have previously been defined. There are two kinds of rules: *lexical rules*, to predict the most likely tag for unknown words, and *contextual rules*, to improve the accuracy based on contextual cues. Lexical rules take into account the morphological information of the words (prefixes, suffixes, capitalization, and so on). Contextual rules are of the form *tag_i is changed by tag_j if P*, that is, the initial tag (tag_i) of a word, must be substituted by the new tag (tag_j), in the context P. The context is formed by the current word, the two words on the left, the current tag and the two tags on the left.

The learning process is carried out basically as follows. The training corpus is initially tagged by assigning to each word the most probable tag. The tagged corpus is compared with the reference corpus. If a word has not been correctly predicted, an instantiation of the corresponding pattern is generated. Then, the set of rules are ordered to obtain the best tagging correction. This process is repeated iteratively until the improvement in the tagging accuracy is under a certain threshold. The tagging process first assigns the most probable tag for each word. If the word is unknown, it applies the lexical rules. Then, the set of learned rules are applied following the order previously learned.

2.2 Maximum entropy

This statistical approach make use of the *Maximum Entropy* principle, that was previously applied by Rosenfeld (1996) in language modeling tasks and, afterwards, in speech recognition systems. The underlying model of this paradigm aims to maximize the entropy of a probabilistic distribution subject to certain constraints. The language model has to be consistent with the events observed in the training data and has to satisfy the constraints introduced. No knowledge about unseen events is assumed.

The most relevant application of the ME model to POS tagging was developed by Ratnaparkhi (1996). In that work, a set of "feature-templates", which takes into account the information appearing in the available context for each word, was defined in order to disambiguate the sentence. For each word w_i, the model's features take into account: the words in the context (the current word, the two previous words w_{i-1} w_{i-2}, and the two posterior words w_{i+1} w_{i+2}) and the tags in the context on the left (bigrams and trigrams). If w_i is a "rare" or uncommon word, the features also include morphological information: prefixes, suffixes, numbers, uppercase characters or special symbols. The inference of the model consists of the estimation of the parameters that combine these features thus maximizing the entropy. The tagging process consists of finding the highest probability tag sequence for a sentence. It is basically a "beam-search" algorithm that enumerates the n-tag sequence candidates with the highest probability up to a token in the sentence.

2.3 Memory-based learning

The Memory-Based Learning (MBL) approach is a kind of supervised learning based on similarity-based reasoning (Daelemans *et al.* 1996). The essential idea of MBL consists of keeping a set of examples or cases obtained from training data in memory. Each case is represented as a feature vector that defines a certain category. All the examples extracted from the training corpus are stored during the training process. The tagging process assigns the tag corresponding to the most similar case held in memory to a word.

To make this approach efficient, it is necessary to correctly define several aspects. Daelemans uses decision-trees as an appropriate data structure for both efficiently classifying all the examples and retrieving information from this data structure by means of the compressing algorithm, *IGTree.* The similarity metric used to obtain the nearest case is basic for a good performance of the tagger. The algorithm *IB-1G* takes into account the distance between the values of a certain feature and ponders them with the *information gain* of the feature. Two base-cases are distinguished in Daelemans *et al.* (1996). For known words, the features are: the ambiguity class of the focus word, the ambiguity class of the first word on the right and the disambiguated tag of the two words on the left. For unknown words, the features considered are: the three suffix letters, the first letter of the word, the tag of the first word on the left and the ambiguity class of the first word on the right.

3 POS tagging and HMMs

From the statistical point of view, POS tagging can be defined as a maximization problem. Let $\mathscr{C} = \{c_1, c_2, \dots, c_N\}$ be a set of POS tags and let $\mathscr{V} = \{w_1, w_2, \dots, w_m\}$ be the vocabulary of the application. Given an input sentence of length T, $W = w_1, \dots, w_T$, the process consists of finding the sequence of POS tags of maximum probability, that is:

$$\widehat{C} = \arg\max_{C} P(C|W) = \arg\max_{C} \left(\frac{P(C) \cdot P(W|C)}{P(W)}\right) ; C \in \mathscr{C}^T \tag{1}$$

Due to the fact that the probability $P(W)$ is a constant that can be ignored in the maximization process, the problem is reduced to maximizing the numerator of equation 1. In this equation, the contextual probabilities or language model, $P(C)$, represent the possible or probable sequences of POS tags. The lexical probabilities, $P(W|C)$, represent the relation between the vocabulary and the POS tags.

To solve equation 1, the Markov assumptions should be made in order to simplify the problem. For second-order Markov models ($n = 2$ or trigrams) and taking into account the Markov assumptions, the problem is reduced to solving the following equation:

$$\widehat{C} = \arg\max_{c_1 \ldots c_T} \left(\prod_{1 \ldots T} P(c_i|c_{i-1}, c_{i-2}) \cdot P(w_i|c_i)\right) \tag{2}$$

The parameters of this equation can be represented as a HMM. We can consider that the states of the model have pairs of POS tags associated to them. Contextual probabilities, $P(c_i|c_{i-1}, c_{i-2})$, correspond to the transition probabilities between the states (c_{i-2}, c_{i-1}) and (c_{i-1}, c_i). Lexical probabilities, $P(w_i|c_i)$, correspond to the output probabilities and, as usual, we assume that they only depend on the most recent category. The tagging process can be carried out by Dynamic Programming Decoding using the Viterbi algorithm (Viterbi 1967).

4 Lexicalized HMMs

In the HMM approach, the relationship among the words in a sentence is not directly captured by the contextual model, because it is established in terms of POS tags. This characteristic means certain relevant relations among words, or among words and tags, are not modelized. Due to the fact that a total lexicalization of the model increases the number of parameter to be estimated excessively, an alternative method is needed to introduce the words in the context. In this sense, in Kim *et al.* (1999) a selective lexicalization of a first-order HMM (bigrams) was proposed by considering a set of “uncommon” words, that is, the words whose probability distribution within a certain category is different from the rest. A new state is made (*lexicalized state*) for each “uncommon” word. The tagging accuracy improved on the *Brown* corpus using this technique from 95.79% to 95.99%.

Lexicalization techniques have also been applied to different paradigms. The Maximum Entropy (ME) model (Ratnaparkhi 1996) is refined by means of specializing some features for “difficult” words (words with a high error rate). However, this specialization had an improvement which was lower than 0.1%. Memory-Based Learning (MBL) methods can consider focus words as features, but the increasing number of parameters makes it difficult to estimate the model. For certain languages, the inclusion of the most frequent words in the feature set slightly increases the performance of the tagger (Zavrel and Daelemans 1999). Transformation-Based

Learning (TBL) (Brill 1995a) also improves performance when it introduces words into the contextual rules (from 97.0% to 97.2% for known words).

The aim of this work is to present a lexicalization technique of the underlying contextual model of a HMM (and in general, any regular model) to enrich it. This technique consists of incorporating a set of selected words (we call them *specialized words*) to the contextual model in addition to the POS tags, in order to establish new lexical-contextual constraints.

The effect of the lexicalization on the model is as follows. If we select a specialized word w_i which is emitted in a state c_i, the lexicalization process splits this state into two states: one state (w_i, c_i) that only emits the word w_i, and another state, the original state c_i, that emits all the words emitted before splitting it, except for w_i. As a lexicalized state can only emit one word, its lexical probability must be equal to one. This process specializes a word in all the categories associated to it in the training data. Therefore, the number of parameters of the model to be estimated is increased. To achieve a reliable modelization, we have to adjust the number of parameters, that is, the number of specialized words, depending on the available training data.

The specialization of certain words (which can be selected taking into account linguistic criteria or extracted automatically from a training data set) produces a better modelization and, therefore, an improvement in the performance of the tagging process as we will show below. The criteria that we have considered for selecting the specialized words are: the most frequent words in the training set, the words with highest tagging error rate and the words that belong to closed categories.

Other more selective lexicalization methods could be considered. For instance, specializing a word in only certain categories. We are not going to make reference to some preliminary experiments as they have shown that this selective lexicalization does not improve our lexicalization proposal. Although a selective lexicalization can reduce the number of parameters in the model, we think that our approach is more practical and can be easily translated to other corpora.

The lexicalization technique proposed consists of relabeling the original training data set taking the specialized words into account. This process is carried out on the training set as follows.

Given the POS tag set $\mathscr{C}$, the vocabulary of the application $\mathscr{V}$, and a training data set $\mathscr{T} \subset (\mathscr{V} \times \mathscr{C})^*$ composed by tuples of words and POS tags $(\langle w_1, c_1 \rangle, \ldots, \langle w_M, c_M \rangle)$, the goal is to get a new training set that includes lexical-contextual information.

Let $\mathscr{W}_s \subset \mathscr{V}$ be the word set to be incorporated to the contextual model. Taking this set into account, a specialization function f_s is defined over the training set $\mathscr{T}$ as follows:

$$f_s : \mathscr{T} \subset (\mathscr{V} \times \mathscr{C})^* \rightarrow \widetilde{\mathscr{T}} \subset (\mathscr{V} \times \widetilde{\mathscr{C}})^*$$

$$f_s(\langle w_i, c_i \rangle) = \begin{cases} \langle w_i, (w_i, c_i) \rangle & \text{if } w_i \in \mathscr{W}_s \\ \langle w_i, (\lambda, c_i) \rangle & \text{if } w_i \notin \mathscr{W}_s \end{cases}$$

This function produces a new training set $\widetilde{\mathscr{T}}$ in which a POS tag c_i is replaced by the new tag (w_i, c_i), if w_i is tagged with c_i and belongs to the set $\mathscr{W}_s$. If this word does not belong to $\mathscr{W}_s$, the POS tag is not changed. In this case, the POS tag has been represented as (λ, c_i)[1], where λ stands for the null string. When this function is applied, there is an extended set of POS tags $\widetilde{\mathscr{C}} \subset ((\mathscr{W}_s \cup \lambda) \times \mathscr{C})$ that encodes the desired words.

The main advantage of this specialization technique is that no change is necessary for either training or tagging processes carried out with a standard HMM approach. To confirm this, all the experimental work was conducted using the TnT[2] tagger (Brants 2000) without making any modification on it.

5 Tagger description

In a corpus-based tagging system, there are two main phases that can be distinguished: the training or learning phase and the tagging phase.

5.1 The learning phase

The learning process of the parameters in equation 2 can be carried out from labelled corpora – supervised methods – (Church 1988; Weischedel *et al.* 1993) or from an unlabelled corpus – unsupervised methods – (Cutting *et al.* 1992; Chanod and Tapanainen 1995). In the first case, the model is trained from the relative frequencies observed. In the second one, the model is learned using the Baum–Welch algorithm from an initial model which is estimated using labelled corpora (Merialdo 1994). In practice, better results are obtained when supervised methods are used (Elworthy 1994).

TnT tagger uses a supervised method that estimates the parameters of the model by Maximum Likelihood from annotated data. Lexical probabilities $P(w_i|c_i)$ are calculated by dividing the frequency of the pair $\langle w_i, c_i\rangle$ by the frequency of the category c_i. Contextual probabilities for trigrams are estimated by dividing the frequency of the sequence (c_i, c_{i-1}, c_{i-2}) by the frequency of the sequence (c_{i-1}, c_{i-2}).

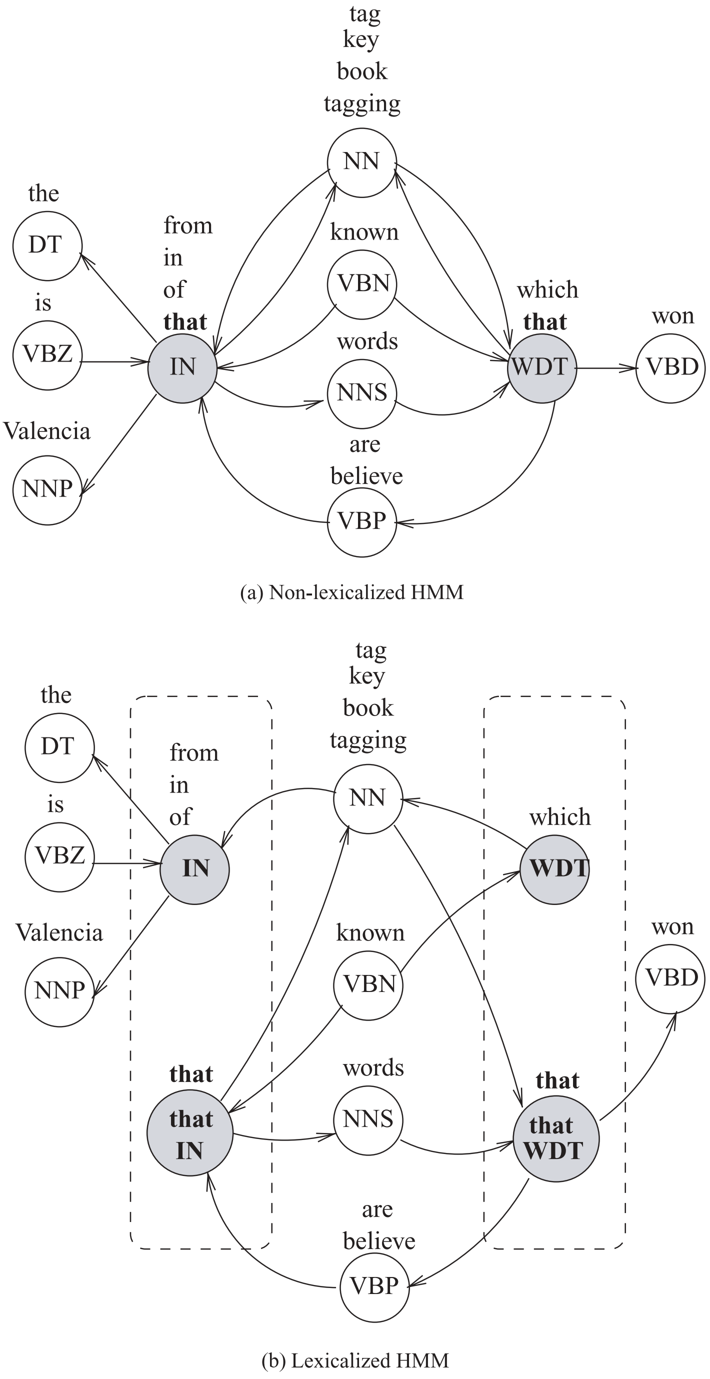
To solve sparse data problems, TnT applies a linear interpolation technique that combines unigram, bigram and trigram information to smooth contextual probabilities. To handle unknown words, it assigns lexical probabilities based on a word suffix probabilistic analysis. A detailed description of this tagger can be consulted in Brants (2000).

We now present an example that clarifies the specialization process on the training set and the differences between the non-lexicalized and the lexicalized models learnt. For simplicity, we will show the result of the specialization on a first-order HMM (see figure 1). This example shows the different contexts where the word “**that**” can appear, that is, as a subordinating conjunction (IN[3]) or as a wh-determiner (WDT).

[1] For simplicity, these POS tags are represented as c_i in the rest of this paper.

[2] TnT is available on the WWW at `http://www.coli.uni-sb.de/~thorsten/tnt`.

[3] The tag “IN” is used in Penn Treebank corpus to label both subordinating conjunctions and prepositions.



(a) Non-lexicalized HMM

(b) Lexicalized HMM

Fig. 1. An example of the result of lexicalization on a first-order HMM.

To do this, the word “**that**” is included in the specialized word set ($\mathscr{W}_s = \{\texttt{that}\}$). In this example, the training set $\mathscr{T}$, annotated with Penn Treebank POS tags, consists of the following sentences:

We/PRP *believe/VBP* **that**/*IN tagging/NN* is/VBZ difficult/JJ.
It/PRP is/VBZ *known/VBN* **that**/*IN words/NNS* are/VBP ambiguous/JJ.
Words/NNS **that**/*WDT are/VBP* ambiguous/JJ are/VBP difficult/JJ to/TO tag/VB.
I/PRP read/VBP the/DT *book/NN* **that**/*WDT won/VBD* the/DT award/NN.
This/DT is/VBZ the/DT *key/NN of/IN the/DT* house/NN.
He/PRP *is/VBZ in/IN the/DT* house/NN.
She/PRP *is/VBZ from/IN Valencia/NNP.*
The/DT tagger/NN has/VBZ *known/VBN which/WDT tag/NN* to/TO assign/VB.

In this case, the application of the specialization function f_s on $\mathscr{T}$ with the set $\mathscr{W}_s$ produces a specialized training set ($\widetilde{\mathscr{T}}$) where only the pair **that**/IN has been replaced by **that**/(that,IN), and the pair **that**/WDT by **that**/(that,WDT).

Figure 1(a) shows a sample of the first-order HMM (only for the training sequences marked in italics in the sentences of the example) obtained from the training set $\mathscr{T}$. Filled states correspond to the categories associated with the specialized word "**that**" in the training data $\mathscr{T}$. Figure 1(b) shows how this model is modified when the new training data $\widetilde{\mathscr{T}}$ is considered. The state `(IN)` is split into two states: a lexicalized state, `(that,IN)`, that only emits the word "**that**" with lexical probability equal to one, and the original state `(IN)` that now does not emit the word "**that**". In a similar way, the state `(WDT)` is split into two. Thus, the lexicalized model can distinguish among the different local contexts where the word "**that**" appears, for example, between `VBP (that,IN) NN` and `NN (that,WDT) VBP`; in the first context, "**that**" will be tagged as IN, and, in the second one, as WDT. Therefore, it can be seen that a *Lexicalized HMM* represents a more specific modelization for certain contexts in which the selected words are involved. That is, this technique attempts to reduce the overgeneration produced by the n-gram models.

5.2 *The tagging phase*

TnT carries out the tagging process by using the well-known Viterbi algorithm, which finds the highest probability state sequence for the input sentence. It uses a "beam search" technique that increases the speed of the tagging process. A correct choice of the beam-pruning threshold does not significantly affect the tagging accuracy. We chose a beam-default value of TnT, which has been empirically tested in Brants (2000). No change is needed in the tagging process for *Lexicalized HMMs*. You simply have to apply a function that undoes the specialization previously defined. This function, f_d, directly maps the sequence of output POS tags (which belong to $\widetilde{\mathscr{C}}$) to the original POS tag set $\mathscr{C}$.

$$f_d : \hat{\mathscr{C}} \to \mathscr{C}$$
$$f_d(\langle w_i, c_i \rangle) = c_i \quad \text{where} \quad w_i \in (\mathscr{W}_s \cup \lambda)$$

6 Experimental work

In this section, we present the evaluation of tagging performance using the models described above. Our first goal was to contrast HMMs against *Lexicalized HMMs*. We considered first-order HMMs (bigrams) and second-order HMMs (trigrams). In this sense, we defined different lexicalization criteria which are independent of the language, and we tested them on English and Spanish corpora. We also conducted an experimental comparison with the most relevant approaches described in section 2. Finally, we studied the influence on tagging precision when certain linguistic resources such as dictionaries (for English) or morphological analyzers (for Spanish) are used. The experimental work was conducted using the TnT tagger with default options and without making any modification on its source code.

6.1 Tagging the WSJ Corpus

We used the part of the Wall Street Journal which had been processed in the *Penn Treebank*, release 2. This corpus was automatically labelled with POS tags and manually checked as described in (Marcus *et al.* 1993). The POS tag set was composed of 45 different tags.

6.1.1 Lexicalization criteria

To test the different lexicalization criteria proposed, we used sections 00 to 19 of the *Penn Treebank* corpus. We divided this data set into two partitions: 90% for training and 10% for tuning (development set).

We defined three criteria to determine the set of words to be used to specialize the models. The first one is based on the frequency of the words in the training set (SWF). The second one only takes into account the words in the training set that belong to closed categories (SCC). The third one takes into account the words with tagging error frequency (SEF) calculated on a development data set.

For SWF criterion, we chose the words whose frequency in the training set was higher than a certain *threshold* (some words such as proper nouns, punctuation signs or numbers were not considered[4]). We specialized the training set and learned the corresponding lexicalized bigram (BIG-SWF) and trigram (TRI-SWF) models using these words.

To determine which *threshold* maximized the performance of the model (that is, the best set of words to specialize the model), we tuned it on a development partition with word sets of different sizes.

In figure 2, we show the results obtained with these specialized models on the development set. The result for zero words corresponds to non-lexicalized models (96.13% for bigrams and 96.44% for trigrams). The accuracy for BIG-SWF and TRI-SWF was better than BIG and TRI, respectively. The best result for BIG-SWF was 96.43% using 286 words (those words whose frequency was higher than 250). The best precision for TRI-SWF was 96.66% using 31 words (with a frequency higher than 2000). It can be observed that with a few words (around 30 words), lexicalized models obtain improvements on the development data set (a 6.2% reduction of the tagging error using trigrams). On the other hand, the use of more words in the models reduces the tagging accuracy. We think this is because the number of

[4] We also performed some experiments that included these “excluded” words, but this increased the number of parameters of the models without improving the tagging results. Another reason to exclude proper nouns and numbers is that we were trying to look for “common” words that could appear in other corpora.

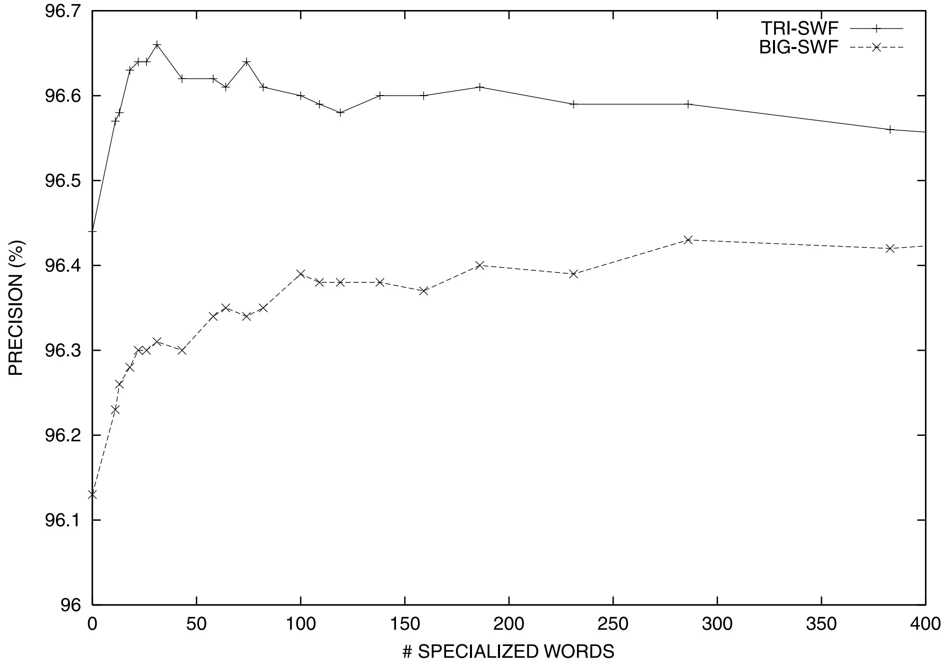


Fig. 2. Performance of *L-HMM* using SWF criterion for different word-set sizes on the development set.

parameters of the models is larger and thus more poorly estimated for the same training set. Moreover, due to the fact that the selection of words was only based on their frequency in the training set, there could be words that do not improve the precision or that do not appear with sufficient frequency on the development set to be able to observe some improvement.

The main drawback of this criterion is that the *threshold* is very dependent on the training and development sets defined, and it can only be known if a tuning experiment is carried out in advance.

Therefore, we defined the SCC criterion which is based on more general properties. In particular, this criterion takes into account only the words from the training set that belong to closed categories[5]. The number of these words was 200 and the precision obtained was 96.34% for bigrams (BIG-SCC) and 96.60% for trigrams (TRI-SCC). These results were slightly lower than those obtained using the SWF criterion (96.43% for BIG-SWF and 96.66% for TRI-SWF), but the SCC criterion is more general and it can be established in advance.

For the third criterion (SEF), we considered the words in the development set whose error frequency was greater than a certain value. The best results (96.68% for trigrams and 96.38% for bigrams) were obtained by specializing the words whose error frequency was greater than 5 (98 words) on the development set using a non-specialized HMM model.

[5] The closed categories considered are: *CC, DT, MD, POS, PP$, RP, TO, WDT, WP$, EX, IN, PDT, PRP, WP, WRB.*

Table 1. *Precision results of the ten-fold cross validation for the different criteria using bigram models on WSJ corpus (sections 00 to 19)*

	BIG	BIG-SWF	BIG-SCC	BIG-SEF
PART_1	96.17%	96.42%	96.36%	96.37%
PART_2	95.91%	96.26%	96.19%	96.17%
PART_3	95.96%	96.26%	96.12%	96.17%
PART_4	96.13%	96.42%	96.40%	96.35%
PART_5	96.27%	96.47%	96.36%	96.38%
PART_6	95.96%	96.32%	96.18%	96.19%
PART_7	96.24%	96.58%	96.45%	96.47%
PART_8	95.73%	96.07%	95.92%	95.96%
PART_9	95.89%	96.22%	96.15%	96.15%
PART_10	96.14%	96.44%	96.35%	96.39%
AVERAGE	96.04% ± 0.11	96.35% ± 0.09	96.25% ± 0.10	96.26% ± 0.10

Table 2. *Precision results of the ten-fold cross validation for the different criteria using trigram models on WSJ corpus (sections 00 to 19)*

	TRI	TRI-SWF	TRI-SCC	TRI-SEF
PART_1	96.45%	96.63%	96.62%	96.63%
PART_2	96.25%	96.49%	96.45%	96.45%
PART_3	96.29%	96.51%	96.46%	96.50%
PART_4	96.50%	96.68%	96.68%	96.63%
PART_5	96.60%	96.76%	96.68%	96.62%
PART_6	96.33%	96.54%	96.49%	96.44%
PART_7	96.49%	96.69%	96.69%	96.68%
PART_8	96.08%	96.33%	96.28%	96.23%
PART_9	96.32%	96.52%	96.51%	96.43%
PART_10	96.45%	96.65%	96.60%	96.68%
AVERAGE	96.38% ± 0.09	96.58% ± 0.08	96.55% ± 0.08	96.53% ± 0.09

To better contrast the different criteria, we conducted a ten-fold cross validation experiment using the entire data set (sections 00 to 19). Each experimental partition consisted of 90% of the data set for training and 10% for the test set. The data test sets were completely different in the different partitions, and so the entire data set was used as a test set. We chose the specialized word sets that were selected in the experiments reported above, without tuning the models in every experiment. We made this decision in order to test the behaviour of these specialized word sets in a more extensive data test.

Tables 1 and 2 show the cross validation results for bigrams and trigrams, respectively. In both cases, the best model (SWF) achieved significant differences with respect to the non-lexicalized one at 95% confidence level[6] (see figure 3).

[6] We calculated the confidence interval by using the formula $\overline{P} \pm 1.96\sqrt{\frac{s^2}{10}}$, where $\overline{P}$ is the average precision and s^2 is the variance.

Table 3. *Total word precision and unknown word precision for HMM and L-HMM models on WSJ corpus (Training set: sections 00 to 19; Test set: sections 23 and 24)*

Model	Total words	Known words	Unknown words	$\lvert\widetilde{\mathscr{C}}\rvert$	$\lvert\mathscr{W}_s\rvert$
BIG	96.28%	96.60%	84.67%	45	0
BIG-SCC	96.52%	96.86%	83.92%	535	200
BIG-SWF	**96.71%**	**97.04%**	**84.46%**	831	286
BIF-SEF	96.50%	96.83%	84.42%	342	98
TRI	96.58%	96.90%	85.17%	45	0
TRI-SCC	96.77%	97.09%	85.29%	535	200
TRI-SWF	**96.80%**	**97.10%**	**85.38%**	144	31
TRI-SEF	96.74%	97.06%	85.21%	342	98

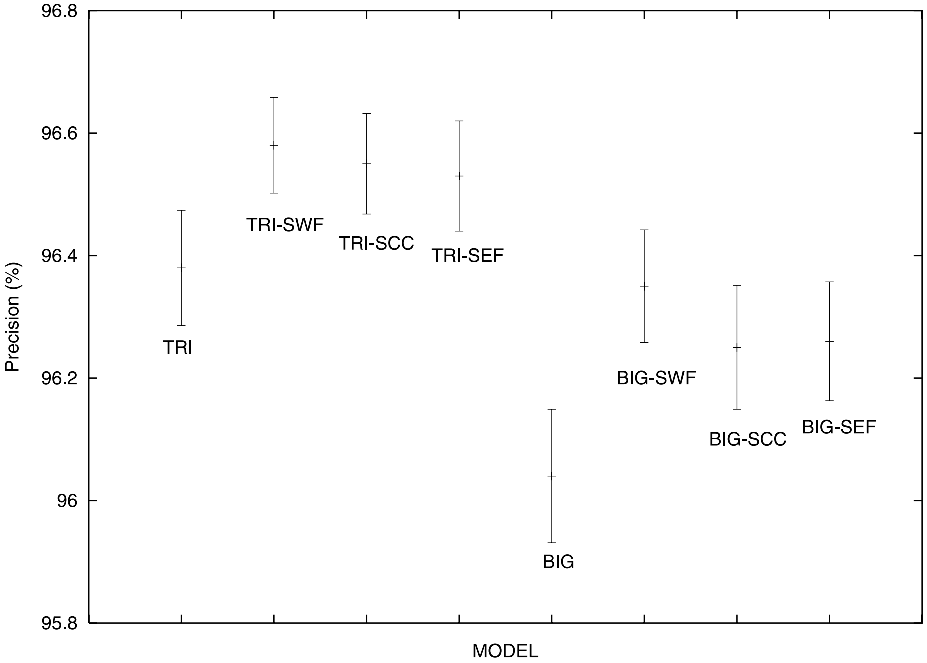


Fig. 3. Confidence intervals for the compared models using 00-19 sections of *WSJ* corpus.

To show that lexicalized criteria behave in a similar way on a new data set, we defined a training set consisting of sections 00 to 19 (956,549 words) and a test set that included sections 23 and 24 (89,529 words). We chose the same specialized word sets that were selected in the tuning experiments reported above. We learned both bigram (BIG) and trigram (TRI) models from this training set. With these models, TnT achieved an accuracy of 96.28% (BIG) and 96.58% (TRI). These results were considered as the baseline system to contrast with the lexicalized models. Table 3 shows the total precision results, distinguishing between known and unknown word precision. In addition, it shows the size of the output tag set ($\lvert\widetilde{\mathscr{C}}\rvert$) and the size of the selected word set ($|\mathcal{W}_s|$). It can be observed that lexicalization improved tagging precision in all cases. The best results were obtained again using SWF models obtaining an error tagging reduction of 11.6% for **BIG-SWF** and an error tagging reduction of 6.4% for **TRI-SWF**.

This experimental result shows that, in some cases, lower-order HMMs performed as well as higher-order HMMs (for instance, 96.71% for BIG-SWF vs. 96.58% for TRI on this test set, or 96.35% for BIG-SWF vs. 96.38% for TRI on the cross-validation). Although the number of parameters for a lexicalized bigram is much bigger than for a non-lexicalized trigram, the lexicalized model includes more features (selected words) in the context. This shows the importance of the selection of these features. We think that this behaviour should be studied more in detail in future works. This could provide a selective method that allows us to include more features without overly increasing the size of the models.

For the bigram model, the error tagging reduction (11.6%) was higher than the result presented in (Kim *et al.* 1999) (about 5% of error reduction). Kim's approach uses a more sophisticated lexicalization method based on the computation of transition vectors for each state. Although the amount of data used was similar, these results are not directly comparable because Kim tested his approach on a different corpus (the Brown corpus).

Finally, we conducted additional experiments combining the different specialized word sets defined above. We combined them using the set operations *union* and *intersection*. None of these experiments achieved significant improvements. The best result for trigrams on the test set was 96.85% using the specialized word set defined as $(\mathcal{W}_{SCC} \cap \mathcal{W}_{SEF}) \cup \mathcal{W}_{SWF}$. This set is composed by the following 64 words. (The words that belong to $\mathcal{W}_{SWF}$ are typed in bold.)

a, *about, ago, all, along,* **an**, **and**, **are**, **as**, **at**, *back,* **be**, *because, both,* **by**, **company**, *do, down, either, enough,* **for**, **from**, *further, had, half,* **has**, **have**, **he**, **in**, **is**, **it**, **its**, *later, left, less, long,* **million**, *more, most, much, next, no,* **of**, *off,* **on**, *one, only,* **or**, *out, over, plus,* **said**, *so,* **that**, **the**, *there,* **to**, *up,* **was**, *what,* **which**, **will**, **with**, **year**

Unfortunately, when this combination was applied to different data sets, e.g. in the development data set, no improvement was achieved. We think that a more extensive study (from the linguistic point of view) of the words that might be significant to the lexicalization process could be done. In particular, it would be important to determine how words belonging to open categories influence this process and which words included in closed categories are more relevant.

6.2 Comparison with other approaches

The results presented in section 6.1 are in line with the best tagging results reported in the literature on the WSJ corpus. However, these results cannot be reliably interpreted because the experimental conditions were different. Therefore, we performed some experiments in order to compare our system to other current tagging approaches under the same experimental conditions. We used the training

Table 4. *Comparison among different taggers on WSJ corpus (Training set: sections 00 to 19; Test set: sections 23 and 24)*

Tagger	Total words	Known words	Unknown words	Training time	Testing speed
TRI-SWF	96.80%	97.10%	**85.38%**	**20 sec.**	**18,000 w/s**
ME	**96.92%**	**97.24%**	85.29%	1 day	70 w/s
TBL	96.47%	96.84%	83.12%	9 days	750 w/s
MBL	96.45%	96.82%	83.18%	4.5 min.	11,200 w/s

and test set defined in section 6.1. The parameters of all taggers were set to optimize the tagging accuracy, but not the training and test time. The experiments for TRI-SWF, TBL[7] and ME[8] were run on a Pentium 266 Mhz with 256MB of RAM. The results for MBL were provided by Walter Daelemans on the same data sets.

Table 4 shows the results of the comparison among these different taggers. We calculated tagging precision (for all words, known words and unknown words), training time and tagging speed (words per second) including file I/O. It can be observed that lexicalized models (TRI-SWF) performed better than TBL and MBL achieving significant differences at the 95% level of confidence[9]. Only ME achieved a precision (96.92% ± 0.11%) which was slightly better than TRI-SWF (96.80% ± 0.11%), but it is not significant at 95% level of confidence (see figure 4). It can also be observed that ME achieved a higher precision than TRI-SWF for known words (97.24% against 97.10%), but that TRI-SWF achieved a precision which was slightly better than ME for unknown words (85.38% against 85.29%). On the other hand, the training time and testing time for ME were much higher than TRI-SWF. This is an important aspect to be taken into account when we plan to incorporate a tagger to a NLP system, because the tagging speed must be very fast in order to construct efficient on-line applications.

6.3 Study of the lexicalization for difficult words

We also studied the effect of the lexicalization on the highest error-rate words in the test set. We compared the absolute tagging error number for these words, for the different tagging approaches compared.

The words whose error rates were improved by the *Lexicalized HMM* are listed in table 5, and the words whose error rates were not improved are listed in table 6. For both tables, the three first columns show the words and their corresponding frequencies in the training and test sets. The rest of the columns correspond to

[7] Available at ftp://ftp.cs.jhu.edu/pub/brill/Programs/. A TBL toolkit (*fnTBL*), developed by the NLP group at Johns Hopkins University, has recently become available at *http://nlp.cs.jhu.edu/˜tbl–toolkit.html*

[8] Available at ftp://ftp.cis.upenn.edu/pub/adwait/jmx/

[9] If the experiment is run only one time, the confidence interval is estimated by using the formula $P \pm 1.96\sqrt{\frac{P(1-P)}{N}}$, where P is the precision and N is the number of samples in the test data set.

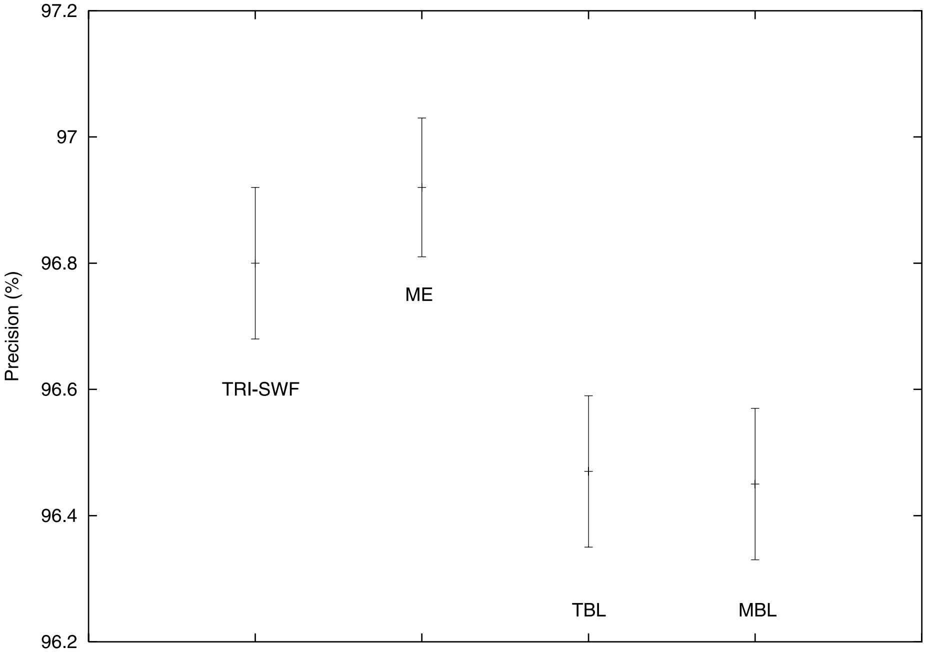


Fig. 4. Confidence intervals for the compared approaches.

the absolute tagging error produced by the different taggers. With respect to the specialized words, the words *that* and *as* decreased the error rate and no improvements were obtained for the words *do* and *on*. Other difficult words, such as *about, ago, as, out, up,* etc., were more accurately disambiguated with respect to the non-lexicalized model. The improvement of the accuracy using TRI-SWF with respect to TRI models was mainly due to the improvements obtained in the words presented in table 5. In general, the number of errors produced by the lexicalized models are in line with or better than the other approaches compared.

6.4 A finer comparison between ME and HMM

Having seen the results presented above, we decided to perform a finer comparison between the approaches that achieved the best results, that is, ME and TRI-SWF. We conducted a ten-fold cross validation experiment in order to better contrast the systems. Again, we used sections 00 to 19 as data set and each experimental partition consisted of 90% of this data set as the training data and 10% for the test set. As we have seen in figure 3, the differences among the tested criteria were not significant and we used the set of words obtained with the best criterion (SWF). This set of words was the same in every partition.

Table 7 show the results[10] of this experiment. From these results we can draw the following conclusions. First, the TRI-SWF model reduced the error tagging about

[10] We also tested TRI-SCC and TRI-SEF criteria obtaining less improvements than TRI-SWF.

Table 5. *Comparison of the tagging error number for the highest error-rate words on the test set, for different tagger approaches. Only the words whose absolute error number has been improved by TRI-SWF model with respect to TRI model are shown*

Words	Ftrain	Ftest	TRI	TRI-SWF	TBL	ME	MBT
’s	9341	903	16	12	17	14	15
about	2063	258	104	93	104	90	98
ago	452	55	17	14	17	15	16
airlines	32	31	20	19	19	16	18
as	4242	461	48	17	15	28	24
back	439	50	23	20	14	10	16
both	375	48	14	11	10	13	10
called	277	35	11	10	6	8	8
capital-gains	75	17	15	14	2	3	4
close	307	43	11	10	6	8	10
communications	50	18	11	9	6	8	7
down	697	94	51	46	52	39	51
ended	299	26	11	8	8	10	11
estimated	201	19	8	3	3	6	2
firm	457	50	9	7	8	6	5
further	244	28	10	9	12	13	10
had	1749	175	8	6	9	7	8
late	297	27	14	10	6	7	6
no	722	79	9	6	8	4	7
off	508	62	14	12	21	14	15
one	1410	149	11	8	10	8	13
out	1011	98	39	21	16	19	29
proposed	211	20	11	10	6	9	7
right	250	28	9	7	4	8	4
securities	418	73	18	17	13	14	15
selling	245	43	9	6	10	7	10
so	628	79	20	11	12	14	11
that	8063	831	88	60	62	94	73
trading	1065	97	20	16	16	14	15
up	1696	146	46	33	36	51	38

6% with respect to the non-lexicalized model TRI. Second, the difference in the total tagging precision between TRI-SWF (96.58%) and ME (96.63%) was insignificant (only an 0.05%). This difference was similar for both known and unknown words.

Finally, we performed an additional experiment in order to test our lexicalized approach in ME. We performed the ten-fold cross validation obtaining a tagging precision of 96.68% (an improvement only of 0.05%). We think this is because ME already includes words in the definition of certain features. Nevertheless, we think this is an open question and an area to be studied more in depth.

6.5 Using the lexicon

It is known that the use of dictionaries improves the accuracy of the taggers. In this sense, we have incorporated a supervised lexicon into the system. We used a POS tag dictionary extracted from the full Penn Tree Bank. Due to errors in corpus annotation, this lexicon was filtered in (Márquez *et al.* 2000) by manually checking the entries for the most frequent 200 words in the corpus (these words cover over half of it). Although in most cases the checking was done by filtering out wrong

Table 6. *Comparison of the tagging error number for the highest error-rate words on the test set, for different tagger approaches. Only the words whose absolute error number has not been improved by TRI-SWF model with respect to TRI model are shown*

Words	Ftrain	Ftest	TRI	TRI-SWF	TBL	ME	MBT
all	1065	113	13	17	13	14	14
around	259	36	8	9	9	13	9
chief	537	58	16	16	15	15	18
deficit-reduction	19	13	8	8	0	6	4
do	940	104	9	11	8	8	8
executive	539	60	13	17	17	13	13
farmers	60	17	14	14	14	13	10
half	253	24	11	13	11	11	11
in	15379	1641	24	24	24	18	22
less	365	39	8	11	10	13	11
machines	93	25	9	9	9	7	8
markets	379	61	10	10	6	6	4
more	1905	198	33	35	35	43	42
most	792	104	13	13	13	15	7
much	662	75	19	21	22	23	20
only	930	86	8	8	8	8	5
on	5162	507	13	17	11	11	18
over	879	82	8	9	5	5	8
p.m.	43	15	9	9	9	4	8
sell	466	55	8	8	6	7	7

Table 7. *Precision results of the ten-fold cross validation*

	TRI	TRI-SWF	ME
PART_1	96.45%	96.63%	96.74%
PART_2	96.25%	96.49%	96.54%
PART_3	96.29%	96.51%	96.57%
PART_4	96.50%	96.68%	96.63%
PART_5	96.60%	96.76%	96.82%
PART_6	96.33%	96.54%	96.58%
PART_7	96.49%	96.69%	96.73%
PART_8	96.08%	96.33%	96.28%
PART_9	96.32%	96.52%	96.61%
PART_10	96.45%	96.65%	96.77%
AVERAGE	96.38%	96.58%	96.63%
AVG-known-words	96.75%	96.96%	97.00%
AVG-unknown-words	85.22%	85.48%	85.53%

tags, there are some entries for which some missing readings were also added. The use of this lexicon is equivalent to having a morphological analyzer that provides the tagger with all the possible tags for every known word. This assumption, which significantly improves the tagging performance, has been used in different works (Márquez *et al.* 2000; Pla *et al.* 2000).

This dictionary did not include statistics from the corpus, that is, frequency of words, of categories, of words per category, etc. This information was only extracted

Table 8. *Precision results for HMM and Lexicalized HMM models, using the Lexicon, on WSJ corpus (Training set: sections 00 to 19; Test set: sections 23 and 24)*

Model	Total words	Known words	Unknown words
Using the lexicon only for KNOWN words			
BIG	96.47%	96.80%	84.79%
BIG-SCC	96.72%	97.08%	83.88%
BIG-SWF	**96.93%**	**97.27%**	**84.33%**
BIF-SEF	96.69%	97.03%	84.38%
TRI	96.79%	97.11%	85.29%
TRI-SCC	96.98%	97.31%	85.33%
TRI-SWF	**96.99%**	**97.31%**	**85.38%**
TRI-SEF	96.95%	97.27%	85.33%
Using the lexicon both for KNOWN and UNKNOWN words			
BIG-SWF	**97.33%**	**97.28%**	**99.21%**
TRI-SWF	**97.38%**	**97.32%**	**99.25%**

from the training set. We took into account the lexicon to modify these statistics as follows[11]:

- If the word was unknown in the training set, TnT assigned a lexical probability for every possible POS given by the lexicon using a smoothing method based on the suffix of the words.
- If the word was known in the training set, we modified the lexical model estimated by TnT as follows:
  1. If the lexicon gave a POS which was not seen in the training set for the word, we incorporated an entry to the lexical model. This entry associated the POS with the word setting the frequency equal to 1, and we added 1 to the total frequency of the word.
  2. If the lexical model gave a POS that did not appear in the lexicon, we removed the POS for this word from the lexical model, and we subtracted its frequency from the total frequency of the word.

Table 8 shows how the use of this lexicon outperformed the tagging precision in all the experiments conducted. This improvement was about 0.2% in all the models. We only used this lexicon for known words on the different lexicalized models presented in section 6.1. We did this as the supervised lexicon is not a complete dictionary. In general, it only provides the POS seen in the corpus for every word. For this reason, in the experiments conducted, unknown words had a low ambiguity ratio and, in general, they were correctly disambiguated, which is not a real situation. The use

[11] We used this simple approach to avoid to modify the source code of TnT tagger.

Table 9. *Precision results on LexEsp corpus using HMM and Lexicalized HMM with SCC criterion*

	TnT			TnT+MACO		
Model	Total words	Known words	Unknown words	Total words	Known words	Unknown words
BIG	95.2%	97.4%	84.0%	96.8%	97.3%	94.2%
BIG-SCC	95.3%	97.5%	84.3%	96.9%	97.5%	94.0%
TRI	95.4%	97.4%	85.4%	96.9%	97.4%	94.7%
TRI-SCC	95.5%	97.5%	85.1%	97.0%	97.4%	94.8%

of this lexicon for both known and unknown words improved the performance by 0.4% as can be see in table 8.

6.6 Tagging the *LexEsp* corpus

For Spanish, we used the *LexEsp* corpus. It contains 5.5 million words of written material, including general news, sports news, literature, etc. This corpus only has 96,000 manually disambiguated words. The tag set contains 62 tags. The percentage of ambiguous words is 39.26% and the average ambiguity ratio is 2.63 tags/word for the ambiguous words (1.64 overall). In this experiment, we chose the SCC lexicalization criterion presented above, that is, the words that belong to closed categories from the training set (45 words). We did not choose the SWF criterion because we did not have enough data to tune the lexicalized model.

We conducted a ten-fold cross validation and the results are summarized in table 9. We show the total and unknown word tagging precision obtained using TnT tagger for different order HMMs. The total tagging precision was lower than the precision obtained for English. This is mainly due to the high error rate for unknown words. The method used by TnT to handle unknown words is based on word-suffix analysis. This method does not seem to work well for languages such as Spanish, which presents a morphology which is more complex than English. To solve this inconvenience, we used the Spanish morphological analyzer MACO+(Carmona *et al.* 1998) as guesser. We can see the improvements obtained in table 9. Lexical model was modified following the method described in section 6.5.

It can also be observed that lexicalized models outperformed non-lexicalized ones, but the differences were not significant at 95% confidence level, maybe because the training data were insufficient.

7 Conclusions

We have presented a method to build *Lexicalized HMMs* by incorporating a set of words to the contextual model. We have followed three different criteria: the most frequent words in the training set (SWF), the words that belong to closed categories (SCC) and the words with the highest error frequency (SEF). These criteria can be applied automatically to any training set, independently of the language or the tag set used.

In all the experiments conducted, the *Lexicalized HMM* outperformed the standard HMM tagger. For English, the lexicalization obtained about 6% reduction of the tagging error using the SWF criterion. This increment in the tagging precision is better than the results presented in other works which use more sophisticated lexicalization methods. We think that a more extensive study of the words that might be significant to the lexicalization could improve the performance of these models. In particular, it would be important to determine how words belonging to open categories influence this process and which words included in closed categories are more relevant.

It would also be interesting to continue research on more selective lexicalization methods that would allow us to include more features without overly increasing the size of the models. This is an important aspect of study in the field of Language Modeling. More work is necessary to contrast our approach using other corpora for the tagging problem or in other tasks that need a language model, such as speech recognition, machine translation, etc.

The exhaustive experimental comparison conducted shows that our approach (*Lexicalized HMMs*) outperforms other current approaches (MBL and TBL) and yields results of precision which are comparable to the ME approach. Our approach also clearly outperforms the training and testing time, so that the incorporation of the tagger to help other tasks such as parsing, machine translation, information retrieval, etc., does not seriously affect the efficiency of these systems.

Although the direct application of this technique in other tagging approaches (such as ME) did not significantly improve the tagging precision, we think that it would be interesting to study the way to adapt this technique to other paradigms. The definition of other lexicalization criteria from a linguistic point of view may enrich this technique. Finally, this technique can also be applied to other NLP tasks which can be treated as classification problems. For example, an adaptation of this technique (Molina and Pla 2002) has been successfully applied to the chunking problem.

Acknowledgments

We would like to thank Thorsten Brants for the use of the TnT tagger. We also would like to thank Walter Daelemans for his help in the comparison with the MBL approach, and the anonymous reviewers for their helpful and valuable comments. This work has been supported by the Spanish research projects CICYT TIC2003–07158–C04–03 and TIC2003–08681–C02–02.

References

Brants, T. (2000) TnT – a statistical part-of-speech tagger. *Proceedings of the Sixth Applied Natural Language Processing (ANLP-2000)*, Seattle, WA.

Brill, E. (1992) A Simple Rule-Based Part-of-speech Tagger. *Proceedings 3rd Conference on Applied Natural Language Processing, ANLP*, pp. 152–155. ACL.

Brill, E. (1993) Automatic Grammar Induction and Parsing Free Text: A Transformation-based Approach. *Proceedings 31st Annual Meeting of the Association for Computational Linguistics.*

Brill, E. (1995a) Transformation–based error–driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics* **21**(4): 543–565.

Brill, E. (1995b) Unsupervised learning of disambiguation rules for part-of-speech tagging. *Proceedings 3rd Workshop on Very Large Corpora*, pp. 1–13. Massachusetts. http://www.cs.jhu.edu/~brill/acadpubs.html.

Brill, E. and Wu, J. (1998) Classifier Combination for Improved Lexical Disambiguation. *Proceedings Joint 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics, COLING-ACL*, pp. 191–195. Montréal, Canada.

Carmona, J., Cervell, S., Màrquez, L., Martí, M. A., Padró, L., Placer, R., Rodríguez, H., Taulé, M. and Turmo, J. (1998) An environment for morphosyntactic processing of unrestricted Spanish text. *Proceedings 1st International Conference on Language Resources and Evaluation, LREC*, pp. 915–922. Granada, Spain, May.

Chanod, J.-P. and Tapanainen, P. (1995) Tagging French – Comparing a Statistical and a Constraint-Based Method. *Proceedings 7th Conference of the European Chapter of the Association for Computational Linguistics, EACL*, pp. 149–156. Dublin, Ireland.

Church, K. W. (1988) A Stochastic parts program and noun phrase parser for unrestricted text. *Proceedings 1st Conference on Applied Natural Language Processing, ANLP*, pp. 136–143. ACL.

Cutting, D., Kupiec, J., Pederson, J. and Sibun, P. (1992) A practical part-of-speech tagger. *Proceedings 3rd Conference on Applied Natural Language Processing, ANLP*, pp. 133–140. ACL.

Daelemans, W., Zavrel, J., Berck, P. and Gillis, S. (1996) MBT: A memory-based part-of-speech tagger generator. *Proceedings 4th Workshop on Very Large Corpora*, pp. 14–27. Copenhagen, Denmark.

Elworthy, D. (1994) Does Baum–Welch Re–estimation Help Taggers? In *Proceedings of the 4th Conference on Applied Natural Language Processing, ANLP*, pages 53–58. ACL.

Kim, J. D., Lee, S. Z. and Rim, H. C. (1999) HMM Specialization with selective lexicalization. *Proceedings Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP-VLC-99).*

Magerman, D. M. (1996) Learning grammatical structure using statistical decision-trees. *Proceedings 3rd International Colloquium on Grammatical Inference, ICGI*, pp. 1–21. *Lecture Notes Series in Artificial Intelligence 1147*. Springer-Verlag.

Marcus, M. P., Marcinkiewicz, M. A. and Santorini, B. (1993) Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* **19**(2).

Márquez, L., Padró, L. and Rodríguez, H. (2000) A machine learning approach to POS tagging. *Machine Learning* **39**(1): 59–91.

Merialdo, B. (1994) Tagging English text with a probabilistic model. *Computational Linguistics* **20**(2): 155–171.

Molina, A. and Pla, F. (2002) Shallow parsing using specialized HMMs. *Journal of Machine Learning Research* **2**: 595–613.

Pla, F. and Molina, A. (2001) Part-of-speech tagging with lexicalized HMM. *Proceedings International Conference on Recent Advances in Natural Language Processing (RANLP2001)*, Tzigov Chark, Bulgaria.

Pla, F. and Prieto, N. (1998) Using grammatical inference methods for automatic part-of-speech tagging. *Proceedings 1st International Conference on Language Resources and Evaluation, LREC*, Granada, Spain.

Pla, F., Molina, A. and Prieto, N. (2000) Tagging and chunking with bigrams. *Proceedings COLING–2000*, Saarbrücken, Germany.

Ratnaparkhi, A. (1996) A maximum entropy part-of-speech tagger. *Proceedings 1st Conference on Empirical Methods in Natural Language Processing, EMNLP.*

Rosenfeld, R. (1996) A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language* **10**: 187–228.

Van Halteren, H., Zavrel, J. and Daelemans, W. (1998) Improving data driven wordclass tagging by system combination. *Proceedings Joint 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics, COLING-ACL*, pp. 491–497. Montréal, Canada.

Van Halteren, H., Zavrel, J. and Daelemans, W. (2001) Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics* **27**(2): 199–229.

Viterbi, A. J. (1967) Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory* 260–269, April.

Weischedel, R., Schwartz, R., Palmucci, J., Meteer, M. and Ramshaw, L. (1993) Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics* **19**(2): 260–269.

Zavrel, J. and Daelemans, W. (1999) Recent advances in memory-based part-of-speech tagging. *Proceedings VI Simposio Internacional de Comunicacion Social*, Santiago de Cuba, Cuba.