Part–of–Speech Tagging Using Decision Trees*

LLUÍS MÀRQUEZ and HORACIO RODRÍGUEZ

Dep. Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya
c/ Jordi Girona 1–3. Barcelona 08034, Catalonia
{lmarquez,horacio}@lsi.upc.es

Abstract. We have applied inductive learning of statistical decision trees to the Natural Language Processing (NLP) task of morphosyntactic disambiguation (Part Of Speech Tagging). Previous work showed that the acquired language models are independent enough to be easily incorporated, as a statistical core of rules, in any flexible tagger. They are also complete enough to be directly used as sets of POS disambiguation rules. We have implemented a quite simple and fast tagger that has been tested and evaluated on the Wall Street Journal (WSJ) corpus with a remarkable accuracy. In this paper we basically address the problem of tagging when only small training material is available, which is crucial in any process of constructing, from scratch, an annotated corpus. We show that quite high accuracy can be achieved with our system in this situation. In addition we also face the problem of dealing with *unknown* words under the same conditions of lacking training examples. In this case some comparative results and comments about close related work are reported.

1 Introduction and State of the Art

POS Tagging is a very well known NLP problem which consists of assigning to each word of a text the proper morphosyntactic tag in its context of appearance. Figure 1 shows the correct part of speech assignment to the words of a sentence, together with the list of valid labels for each word taken in isolation[1]. The base of POS tagging is that being most words ambiguous regarding their POS, they can be almost completely disambiguated taking into account an adequate context.

Starting with the pioneer tagger TAGGIT (Greene & Rubin 71), used for an initial tagging of the Brown Corpus (BC), a lot of efforts have been devoted to improve the quality of the tagging process in terms of accuracy and efficiency. Existing taggers can be classified into three main groups according to the kind of knowledge they use: linguistic, statistic and machine–learning family. Of course

* This research has been partially funded by the Spanish Research Department (CICYT's ITEM project TIC96–1243–C03–02), by the EU Commission (EuroWordNet LE4003) and by the Catalan Research Department (CIRIT's quality research group 1995SGR 00566).

[1] All tags appearing in the paper are from the Penn Treebank tag set. They are described in figure 2. For a complete description see for instance (Marcus et al.93).

The_DT first_JJ time_NN he_PRP was_VBD shot_VBN in_IN the_DT hand_NN as_IN he_PRP chased_VBD the_DT robbers_NNS outside_RB ...

first	time	shot	in	hand	as	chased	outside
JJ	NN	NN	IN	NN	IN	JJ	IN
RB	VB	VBD	RB	VB	RB	VBD	JJ
		VBN	RP			VBN	NN
							RB

Fig. 1. A sentence and its POS ambiguity

DT: Determiner	RB: Adverb	VBD: Verb, past tense
IN: Preposition	RP: Particle	VBN: Verb, past participle
JJ: Adjective	TO: *to*	
NN: Noun, singular	VB: Verb, base form	

Fig. 2. A subset of the Penn Treebank tag set

some taggers are difficult to classify into these classes and hybrid approaches must be considered.

Within the linguistic approach most systems codify the involved knowledge as a set of rules (or constraints) manually written by linguists (usually around a thousand rules). The work of the TOSCA group (Oostdijk 91) and more recently the development of Constraint Grammars (Karlsson et al. 95) can be considered the most important in this direction.

The most extended approach nowadays is the statistical family (obviously due to the limited amount of human effort involved). Basically it consists of building a statistical model of the language and using this model to disambiguate a word sequence. There are different approaches in the estimation of the parameters of the model, i.e. the lexical and transition probabilities. The form of the model and the way of determining the sequence to be modeled can be approached too in several ways. Many systems reduce the model to unigrams, bi–grams, tri–grams, or to a combination of them. Hidden Markov Models have been widely used too. The seminal work in this direction is the CLAWS system (Garside et al. 87), which was the probabilistic version of TAGGIT. (Church 88), (DeRose 88) or (Cutting et al. 92) are notable examples of statistic taggers. (Merialdo 94) presents an excellent overview.

Other works that can be placed in this class are those of (Schmid 94a) which performs energy–function optimization using neural nets and (Rosenfeld 94) which has applied a Maximum Entropy Approach to POS tagging. A comparison between linguistic and statistic taggers can be found in (Samuelsson & Voutilainen 97).

Although the statistic approach involves some kind of learning, either supervised or unsupervised, of the parameters of the model from a training corpus, we place in the machine-learning family only those systems that include more sophisticated information than a n–gram model. (Brill 92) and (Brill 95) automat-

ically learn a set of transformation rules which best repair the errors commited by a most–likely–tag tagger, (Samuelsson et al. 96) acquires Constraint Grammar rules from tagged corpora and (Daelemans et al. 96) or the work presented here learn decision trees from tagged corpora.

An example of hybrid approach is (Padró 97) that applies relaxation techniques over a set of constraints involving statistical, linguistic and machine–learning obtained information.

The accuracy reported by most statistic taggers overcomes 96–97% while Constraint Grammars overcome 99% allowing a residual ambiguity of 1.026 tags per word. Taking these figures into account one may think that POS tagging is a solved and closed problem being this accuracy perfectly acceptable for most NLP systems. So why wasting time in designing yet another tagger? What does an increasing of 0.3% in accuracy really mean? We think that there are several reasons for thinking that there is still work to do in the field of automatic POS tagging.

Common sentences in running texts have an average lenght of around 30 words. If we admit an error rate of 3–4% then it follows that, on average, each sentence contains one error. Since POS tagging is a very basic task in most NLP understanding systems, starting with an error in each sentence could be a severe drawback, specially considering that the propagation of this errors could grow more than linearly. Other NLP tasks that are very sensitive to POS disambiguation errors can be found in the domain of Word Sense Disambiguation (Wilks and Stevenson 97) and Information Retrieval (Krovetz 97).

Another issue refers to the need of adapting and tuning taggers that have acquired (or learned) their parameters from an specific corpus onto another one trying to minimizing the cost of transportation. No serious attempts have been performed to test the reported accuracy of taggers (usually measured against reference corpora as Wall Street Journal corpus –WSJ– or BC) on different, perhaps domain-specific, corpora.

Finally, some specific problems must be addressed when applying taggers to other languages than English. Beside the problems derived from the richer morphology of the particular language, there is a more general problem consisting of the lack of large, manually annotated corpora for training.

Although a *bootstrapping* approach can be carried out, using a low-accurate tagger, for producing annotated text that could be used then for learning a more accurate model, the results of such approach are dubious. So there is a real need for methods achieving high accuracy, both on known and unknown words, learning from small high-quality corpora.

In this direction, we are involved in a project for tagging Spanish and Catalan corpora (over 5M words) with limited linguistic resources, that is, departing from a manually tagged core of a size not greater than 50000 words.

For the sake of comparability the experiments reported here are performed over a reference corpus of English. However we fairly believe that qualitative results could be extrapolated to Spanish or Catalan.

The paper is organized as follows: In section 2 we describe the domain of application, in section 3 we describe the language model acquisition and the tagger implementation and in section 4 we describe the whole set of experiments together with a comparative analysis of the obtained results. Finally, the main conclusions and an overview of the future work can be found in section 5.

2 Domain of Application

Choosing, from a set of possible tags, the proper syntactic tag for a word in a particular context can be seen as a problem of classification. In this case, classes are identified with the tags. Decision trees, recently used in several NLP basic tasks, such as tagging and parsing (Schmid 94b), (McCarthy & Lehnert 95), (Daelemans et al. 96), (Magerman 96), are suitable for performing this task.

Ambiguity classes

It is possible to group all the words appearing in the corpus according to the set of their possible tags (i.e. *adjective–noun*, *adjective–noun–verb*, *adverb–preposition*, etc.). We will call this sets *ambiguity classes*. It is obvious that there exists an inclusion relation between these classes (i.e. all the words that can be *adjective*, *noun* and *verb*, can be, in particular, *adjective* and *noun*), so the whole set of ambiguity classes is viewed as a taxonomy with a DAG structure. In this way we split the general POS tagging problem into one classification problem for each ambiguity class.

Description of the corpus

We have used a portion of 1,170 Kwords of the WSJ, tagged according to the Penn Treebank tag set, to train and test the system. The tag set contains 45 different tags[2]. About 36.5% of the words in the corpus are ambiguous, with an ambiguity ratio of 2.44 tags/word over the ambiguous words, 1.52 overall. The corpus contains 243 different ambiguity classes, but they are not all equally important. In fact, only the 40 most frequent ambiguity classes cover 83.95% of the occurrences in the corpus, while the 194 most frequent cover almost all of them (>99.50%).

The training corpus has been used also to create a word form lexicon with the associated lexical probabilities for each word. These probabilities are simply estimated by counting the number of times each word appears in the corpus with each different tag.

Statistical decision trees

We identify some particular features in our domain, comparing with common

[2] The size of tag sets differ greatly from one domain to another. Depending on the contents, complexity and level of annotation they are moving from 30–40 to several hundreds of different tags. Of course, these differences have important effects in the performance rates reported by different systems and imply difficulties when comparing them. See (Krenn & Samuelsson 96) for a more detailed discussion.

classification domains in Machine Learning field. Firstly, there is a so high number of training examples: up to 60000 examples for a single tree. Secondly, there is quite significant noise in the training/test data: WSJ corpus contains about 2–3% of mistagged words.

The main consequence of the above characteristics, together with the fact that simple context conditions cannot explain all ambiguities (Karlsson et al. 95)), is that it is not possible to obtain trees for completely classify the training examples. Instead, we aspire to obtain more adjusted probability distributions of the words over their possible tags, conditioned to the particular contexts of appearance. So we will use *Statistical* decision trees, instead of common decision trees, for representing this information.

3 Brief Description of the Tree-Based Tagger

3.1 Language Model Acquisition

The algorithm we used for constructing the statistical decision trees is a non–incremental supervised learning–from–examples algorithm of the TDIDT (Top Down Induction of Decision Trees) family. It constructs the trees in a top–down way, guided by the distributional information of the examples (Quinlan 93).

Training Set

For each class of POS ambiguity the initial example set is built by selecting from the training corpus all the occurrences of the words belonging to this ambiguity class. For most of the experiments reported in section 4, the set of attributes that describe each example consists of the part–of–speech tags of the neighbour words, and the information about the word itself: orthography and the proper tag in its context. The window considered is 3 words to the left and 2 to the right. The following are two real examples from the training set for the words that can be preposition and adverb at the same time (`IN-RB` class).

```
VB DT NN <"as",IN> DT JJ
NN IN NN <"once",RB> VBN TO
```

Attributes with many values (for instance the *word–form* and other attributes used when dealing with unknown words) are treated by dynamically adjusting the number of values to the N most frequent and joining the rest in a new *default* value. The maximum number of values is fixed to 45 (the number of different tags) in order to have homogeneous attributes.

Attribute Selection Function

After testing several attribute selection functions, with no significant differences between them, we used an attribute selection function due to López de Mántaras (López de Mántaras 91), belonging to the information–based family, which showed a slightly higher stability than the others. Roughly speaking, it defines a distance measure between partitions and selects for branching the attribute that generates the closest partition to the *correct partition*, namely the one that joins together all the examples of the same class.

Branching Strategy

Usual TDIDT algorithms consider a branch for each value of the selected attribute. However other solutions are possible. For instance, some systems perform a previous recasting of the attributes in order to have binary–valued attributes (Magerman 96). The motivation could be efficiency (dealing only with binary trees has certain advantages), and avoiding excessive data fragmentation (when there is a large number of values). Although this transformation of attributes is always possible, the resulting attributes lose their intuition and direct interpretation, and explode in number. We have chosen a mixed approach which consists of splitting for all values and afterwards joining the resulting subsets into groups for which we have not enough statistical evidence of being different distributions. This statistical evidence is tested with a χ^2 test at a 5% level of significance, with a previous smoothing of data in order to avoid zero probabilities.

Pruning the Tree

In order to decrease the effect of *over-fitting*, we have implemented a post pruning technique. In a first step the tree is completely expanded and afterwards is pruned following a minimal cost–complexity criterion (Breiman et al. 84), using a comparatively small fresh part of the training set. The alternative of smoothing the conditional probability distributions of the leaves using fresh corpus (Magerman 96) has been left out because we also wanted to reduce the size of the trees. Experimental tests have shown that in our domain the pruning process reduces tree sizes up to 50% and improves their accuracy in a 2–5%.

3.2 Tagging Algorithm

We have implemented a *reductionistic* tagger in the sense of constraint grammars (Karlsson et al. 95). In a initial step a word-form frequency dictionary constructed from the training corpus provides each input word with all possible tags with their associated lexical probability. After that, an iterative process reduces the ambiguity (discarding low probable tags) at each step until a certain stopping criterion is satisfied. The whole process is represented in figure 3.

More particularly, at each step and for each ambiguous word the work to be done is: 1) Classify the word using the corresponding decision tree[3]. 2) Use the resulting probability distribution to update the probability distribution of the word[4]. 3) Discard the tags with *almost* zero probability, that is, those with probabilities lower than a certain *discard boundary* parameter.

After the stopping criterion is satisfied some words could still remain ambiguous. Then there are two possibilities: 1) Choose the most-likely tag for each

[3] Ambiguity of the context during classification may generate multiple answers for the questions of the nodes. In this case, all the paths are followed and the result is taken as a weighted average of the results of all possible paths.

[4] The updating of the probabilities is done by simply multiplying previous probabilities per new evidences.

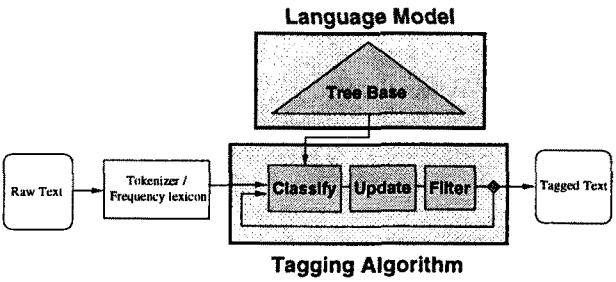

Fig. 3. The tagging process

still ambiguous word to completely disambiguate the text. 2) Accept the residual ambiguity (perhaps for treating it in successive stages).

Note that a unique iteration forcing the complete disambiguation is equivalent to use directly the trees as classifiers and results in a very efficient tagger, while performing several steps reduces progressively the efficiency but takes advantage of the statistical nature of the trees.

Another important point is to determine an appropriate stopping criterion. First experiments seem to indicate than the performance increases up to a unique maximum and then softly decreases as the number of iterations increases. For the experiments reported in section 4, the number of iterations was fixed to 3.

4 Experiments

We report here the results of four experiments. The first two summarize the work done for testing the system when all the training material is available. These two experiments were reported in previous papers (Màrquez & Rodríguez 95), (Màrquez & Rodríguez 97), (Màrquez & Padró 97). Third and fourth experiments are devoted to the testing of the tagger when using small training sets. We treat separately the cases of dealing either with known or unknown words. In both cases we give comparative results with similar work by (Daelemans et al. 96).

4.1 First Experiment

We divided the WSJ corpus in two parts: 1,120 Kw were used as a training/pruning set, and 50 Kw as a fresh test set. We used a lexicon derived from training corpus, containing all possible tags for each word, as well as their lexical probabilities. The noise in the lexicon was filtered by manually checking the lexicon entries for the most frequent 200 words in order to eliminate the tags due to errors in the training set. Note that the 200 most frequent words in the corpus represent over half of it. For the words in the test corpus not appearing in

the train set, we stored all possible tags, but no lexical probability (i.e. assuming uniform distribution)[5].

From the 243 ambiguity classes the acquisition algorithm learned a base of 194 trees covering 99.5% of the ambiguous words and requiring about 500 Kb of storage. The tagging algorithm, running on a SUN UltraSparc2, processed the test set at a speed of >300 words/sec, obtaining the following global results: when forcing a complete disambiguation the resulting accuracy was 97.29%, while accepting residual ambiguity the accuracy rate increased up to 98.22%, with an ambiguity ratio of 1.08 tags/word over the ambiguous words and 1.026 tags/word overall[6]. In (Màrquez & Rodríguez 97) it is shown that these results are, at least, as good as the results of a number of the non linguistically motivated state-of-the-art taggers.

4.2 Second Experiment

Another test of the appropriateness of the tree model was done in a previous experiment with another tagger. The group of the 44 most representative trees (covering 83.95% of the examples) were translated into a set of weighted context constraints and used to feed a relaxation-labelling-based tagger, together with bi/tri-gram information. The usual way of expressing trees as a set of rules was used to construct the context constraints. For instance the two following constraints, extracted from a real tree branch for the **IN-RB** (preposition-adverb) ambiguity class,

```
-5.81 <["as" "As"],IN> ([RB]) ([IN]);
2.366 <["as" "As"],RB> ([RB]) ([IN]);
```

express the compatibility (either positive or negative) of the word-tag pair in angle brackets with the given context. The compatibility value for each constraint is calculated as the *mutual information* (Cover & Thomas 91) between the tag and the context.

Reported results, 97.09% accuracy when using the tree model alone and 97.39% when combining with a trigram model, showed that the addition of the automatically acquired context constraints led to an improvement in the accuracy of the tagger[7].

4.3 Small Training Sets

We present in figure 4 the performance achieved by our tagger with increasing sizes of the training corpus. Results in accuracy are taken over all words. The same figure includes most-likely results, which can be seen as a lower bound.

[5] That is, we assumed a morphological analyzer that provides all possible tags for unknown words.

[6] In other words, 2.75% of the words remained ambiguous, retaining only 2 tags for over 96% of them.

[7] Overcoming the bi/tri-gram models and properly cooperating with them and with a small set of linguistically motivated hand-written constraints.

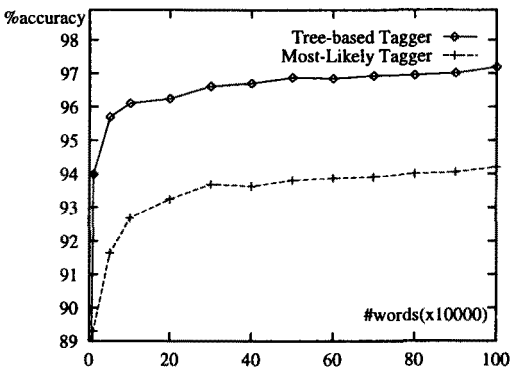

Fig. 4. Performance of the tagger related to the training set size

Following the intuition, we see that performance grows as the training set size grows. The maximum is at 97.29%, as reported in the first experiment. For the case of our interest we can see that using only 50000 training examples, the accuracy rate is 95.69%. This figure has been calculated as the average of the results obtained by repeating the experiment ten times on completely different training material. This mean value has a confidence interval of $\pm 0.17\%$, at a 95% confidence rate.

We think this result is quite accurate. In order to corroborate this statement we can compare our result on a training set of 100K examples (an accuracy of 96.16%) with the figures reported in (Daelemans et al. 96) for the IGTree[8] system (96.0%). So our results can be considered, at least, as good as theirs.

4.4 Unknown Words

Unknown words are those words not present in the lexicon[9]. In the previous experiments we have not considered the possibility of unknown words. Instead we have assumed a morphological analyzer providing the set of possible tags with a uniform probability distribution. However, there exist several approaches to deal with real unknown words. On the one hand one can assume that unknown words may potentially take any tag, excluding those tags corresponding to closed categories (preposition, determiner, etc.), and try to the disambiguate between them. On the other hand, other approaches includes a pre–process that tries to guess the set of candidate tags for each unknown word to feed the tagger with this information. See (Padró 97) for a detailed explanation of the methods.

[8] IGTree system is a Memory–Based POS tagger which uses a tree representation of the training set of examples. The main difference from our solution is that, in IGTree, the order of application of attributes is predetermined in the construction of the trees.

[9] That is, in our case, the words not present in the training corpus.

In our case we consider unknown words as words belonging to the ambiguity class containing all possible tags corresponding to opened categories (i.e. noun, proper noun, verb, adjective, adverb, cardinal, etc.). The number of candidate tags sum to 20, so we state a classification problem with 20 different classes. We have estimated the proportion of each of these tags appearing naturally in the WSJ as unknown words and we have collected the examples from the training corpus according to these proportions. The most frequent tag, **NNP** (proper noun), represents almost 30% of the sample. This fact establishes a lower bound for accuracy of 30% in this domain (i.e. the performance that a *most-likely-tag* tagger would obtain).

We have used very simple information about the orthography and the context of unknown words in order to improve these results. In particular, from an initial set of 17 potential attributes, we have empirically decided the most relevant, which turned out to be the following ten: 1) On the word form: the first letter, the last three letters, and four binary-valued attributes more, accounting for capitalization, whether the word is a multi-word or not, and for the existence of some numeric characters in the word. 2) On the context: just the preceding and the following POS tags.

Table 1 shows the generalization performance of the trees learned from training sets of increasing sizes up to 50000 words. In order to compare these figures again with the results of IGTree, we have implemented IGTree algorithms and we have tested its performance exactly under the same condition as ours. These results are also shown in table 1.

#exs.		Tree-based Tagger	IGTree
2000	accuracy	77.53%	70.364%
	#nodes	224	627
5000	accuracy	80.90%	76.33%
	#nodes	520	1438
10000	accuracy	83.300%	79.18%
	#nodes	1112	2664
20000	accuracy	85.82%	82.30%
	#nodes	1644	4783
30000	accuracy	87.32%	85.11%
	#nodes	2476	6477
40000	accuracy	88.00%	86.78%
	#nodes	2735	8086
50000	accuracy	88.12%	87.145%
	#nodes	4056	9554

Table 1. Generalization performance of the trees for unknown words

Note that our system produces better quality trees than those of IGTree. We measure this quality in terms of generalization performance (how well these trees fit new examples) and size (number of nodes). Of course, this conclusion has to be taken in the domain of small training sets. Using big corpora for training might improve performance significantly. For instance, (Daelemans et al. 96) report an accuracy rate of 90.6% on unknown words when training with 2 million words of the WSJ.

5 Conclusions and Future Work

We have applied a classical supervised algorithm of the machine learning field, in order to automatically acquire a language model for POS tagging based on statistical decision trees. This learning algorithm uses more complex contextual information than usual n–gram models and it can easily accept other kinds of information. We have used this model for developing a fast and simple tagger tested on the WSJ corpus with a remarkable accuracy. In addition we have shown the independence of the acquired language model from the particular tagging algorithm, by translating the trees into a set of context constraints to feed a flexible relaxation–labelling–based tagger. Results obtained with this tagger are fairly good. Finally, we have tested the appropriateness of our system when dealing with small training corpora, as a previous step for applying the tagger to the Spanish and Catalan. We have obtained encouraging results both in tagging known and unknown words.

However, further work is still to be done in several directions. Referring to the language model learning algorithm, we are interested in testing more informed attribute selection functions, considering more complex questions in the nodes and finding a good smoothing procedure for dealing with very small ambiguity classes. See (Màrquez & Rodríguez 97) for a first approach.

About the information that this algorithm uses, we want to explore the inclusion of more morphological and semantic information, as well as more complex context features, such as non–limited distance or barrier rules in the style of (Samuelsson et al. 96).

Regarding the current work, we are beginning to apply our tagger to Spanish and Catalan languages. In this direction we are interested in testing our system —alone and in cooperation with other taggers, as (Padró 97)— in order to verify the hypothesis stated in this paper.

We conclude saying that we have done first attempts in using the same techniques to tackle another classification problem in NLP area, namely Word Sense Disambiguation (WSD). We believe, as other authors do, that we can take profit of treating both problems jointly.

References

Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J.: *Classification and Regression Trees.* Wadsworth International Group, Belmont, California, 1984.

Brill, E.: *A Simple Rule–Based Part–of–Speech Tagger.* In Proceedings of the 3rd ACL Conference on Applied Natural Language Processing, 1992.

Brill, E.: *Unsupervised Learning of Disambiguation Rules for Part–of–speech Tagging.* Proceedings of 3rd Workshop on Very Large Corpora, Massachusetts, 1995.

Church, K.W.: *S Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text.* In proc. of 2nd Conference on Applied Natural Language Processing, 1988.

Cover, T.M. and Thomas, J.A. (Editors): *Elements of Information Theory.* John Wiley & Sons, 1991.

Cutting, D., Kupiec, J., Pederson, J. and Sibun, P.: *A Practical Part–of–Speech Tagger.* In proc. of 3rd Conference on Applied Natural Language Processing, 1992.

DeRose, S.J.: *Grammatical Category Disambiguation by Statistical Optimization.* Computational Linguistics 14(1), pp. 31–39.

Daelemans, W., Zavrel, J., Berck, P. and Gillis, S.: *MTB: A Memory-Based Part-of-Speech Tagger Generator.* Proc. of 4th Workshop on Very Large Corpora, 1996.

Garside, R., Leech, G. and Sampson, G.: *The Computational Analysis of English.* London and New York: Longman, 1987.

Greene, B.B., and Rubin, G.M.: *Automatic Grammatical Tagging of English.* Technical Report, Department of Linguistics, Brown University, 1971.

Karlsson, F., Voutilainen, A., Heikkilä, J. and Anttila, A.: *Constraint Grammar. A Language-Independent System for Parsing Unrestricted Text.* Mouton de Gruyter, Berlin, New York, 1995.

Krenn, B. and Samuelsson, C.: *The Linguist's Guide to Statistics. Don't Panic.* Universität des Saarlandes. Saarbrücken. Germany. WWW: http://coli.uni-sb.de

Krovetz, R.: *Homonymy and Polysemy in Information Retrieval.* In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, ACL '97.

López de Mántaras, R.: *A Distance-Based Attribute Selection Measure for Decision Tree Induction.* Machine Learning, Kluwer Academic, 1991.

Magerman, M.: *Learning Grammatical Structure Using Statistical Decision-Trees.* In proc. of the 3rd International Colloquium on Grammatical Inference, ICGI '96.

Marcus, M.P., Marcinkiewicz, M.A. and Santorini, B.: *Building a Large Annotated Corpus of English: The Penn Treebank.* Computational Linguistics, v.19, n.2, 1993.

Màrquez, L. and Rodríguez, H.: *Towards Learning a Constraint Grammar from Annotated Corpora Using Decision Trees.* ESPRIT BRA-7315, WP #15, 1995.

Màrquez, L. and Padró, L.: *A Flexible POS Tagger Using an Automatically Acquired Language Model.* In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, ACL '97.

Màrquez, L. and Rodríguez, H.: *Automatically Acquiring a Language Model for POS Tagging Using Decision Trees.* In Proceedings of the Second Conference on Recent Advances in Natural Language Processing, RANLP '97.

McCarthy, J.F. and Lehnert, W.G.: *Using Decision Trees for Coreference Resolution.* Proceedings of 14th IJCAI, 1995.

Merialdo, B.: *Tagging English Text with a Probabilistic Model.* Computational Linguistics 20(2), pp. 155-171.

Oostdijk, N.: *Corpus Linguistic and the automatic analysis of English.* Rodopi, Amsterdam, 1991.

Padró, L.: *A Hybrid Environment for Syntax-Semantic Tagging.* PhD Thesis, Dep. Llenguatges i Sistemes Informàtics, Universitat Politecnica de Catalunya, 1998.

Quinlan, J.R.: *C4.5: Programs for Machine Learning.* San Mateo, CA. Morgan Kaufmann, 1993.

Rosenfeld, R.: *Adaptive Statistical Language Modeling: A Maximum Entropy Approach.* PhD Thesis. School of Computer Science, Carnegie Mellon University, 1994.

Samuelsson, C., Tapanainen, P. and Voutilainen, A.: *Inducing Constraint Grammars.* Proceedings of the 3rd International Colloquium on Grammatical Inference, 1996.

Samuelsson, C. and Voutilainen, A.: *Comparing a Linguistic and a Stochastic Tagger.* In Proceedings of the 35th Annual Meeting of the ACL, 1997.

Schmid, H.: *Part-of-speech tagging with neural networks.* Proceedings of 15th International Conference on Computational Linguistics, COLING '94.

Schmid, H.: *Probabilistic Part-of-Speech Tagging Using Decision Trees.* In Proceedings of the Conference on New Methods in Language Processing, Manchester, UK, 1994.

Wilks, Y. and Stevenson, M.: *Combining Independent Knowledge Sources for Word Sense Disambiguation.* In Proceedings of the Second Conference on Recent Advances in Natural Language Processing, RANLP '97.