An Environment for Morphosyntactic Processing of Unrestricted Spanish Text

J. Carmona*, S. Cervell†, L. Màrquez*, M.A. Martí†, L. Padró*, R. Placer†, H. Rodríguez*, M. Taulé†, J. Turmo*.

* Software Department – Universitat Politècnica de Catalunya
c/ Jordi Girona 1–3, 08034 Barcelona, Catalonia.

† Computational Linguistics Laboratory – Universitat de Barcelona
Gran Via 585, 08007 Barcelona, Catalonia.

Abstract

We present in this paper a fast, broad-coverage, accurate morphological analyzer for Spanish words, MACO+, which is an extended and improved version of that described in (Acebo *et al.*, 1994). The earlier version had two main flaws: it was not transportable, and it was too slow to enable massive text processing. The presented system not only overcomes those two flaws, but also offers improved coverage and accuracy. We also present two general part-of-speech taggers, which can be used to disambiguate the output of the morphological analyzer. All modules run in any Unix/Linux machine as a pipeline process and they may also be used inside the GATE environment for NLP (Cunningham *et al.*, 1996). The system is currently being used to annotate the LexEsp corpus, a 5.5 million word corpus of Spanish, in a bootstrapping refining procedure. Initial evaluation and results are reported.

Keywords: Morphological analysis, corpus linguistics, POS tagging, linguistic resources.

1 Introduction and Motivation

We present in this paper a fast, broad-coverage, accurate morphological analyzer for Spanish words, MACO+, which is an extended and improved version of that described in (Acebo *et al.*, 1994).

The output of the morphological analyzer can be used as the input for a part–of–speech (POS) tagger, we have used two different taggers to disambiguate the analyzed text, in a similar way to that described in (Màrquez & Padró, 1997).

The whole system is being used to develop a 5.5Mw corpus of unrestricted nowadays Spanish.

The analyzers have been developed and used in the framework of the ITEM and LexEsp projects. Both projects aim to integrate and develop tools and resources for NLP and for linguistic research in Spanish. Both projects are briefly described below.

1.1 ITEM Project

ITEM is a project funded by Spanish Research Department (CICYT) consisting basically of integrating different existing NLP tools and resources in a unique environment, in order to enable and ease the construction of multilingual information extraction and retrieval systems.

The environment includes tools for NLP of Catalan, Basque and Spanish. The integrated tools include basic NL tasks (tokenizers, morphological analyzers, taggers, parsers, etc.) as well as higher level tools oriented to information extraction. New tools and resources are also being developed, and existing tools are improved and integrated, as is the case of the morphological analyzer presented in this paper.

The integration environment also contains several lexical resources such as corpus, machine–readable dictionaries (MRDs), lexicons, taxonomies, grammars, etc.

All the integrated tools and resources are documented, available and transportable. The software used to support this integration is GATE (Cunningham *et al.*, 1996).

Partners in this project are the Computational Linguistics Group from the University of Barcelona (http://www.ub.es/ling/labcat.htm), the NLP research group from the Technical University of Catalonia (http://www.lsi.upc.es/ acquilex/nlrg.html), the NLP group from the Basque Country University (http://www.ji.si.ehu.es/Groups/IXA/), and the NLP group from the Spanish Open University, UNED (http://sensei.ieec.uned.es/item/grupoLN.htm).

1.2 LexEsp Project

The LexEsp Project is a multi–disciplinary effort impulsed by the Psychology Department from the University of Oviedo. It aims to create a large database of language usage in order to enable and potentiate research activities in a wide range of fields, from linguistics to medicine, through psychology and artificial intelligence, among others.

One of the main issues of that database of linguistic resources is the LexEsp corpus, which contains 5.5 Mw of written material, including general news, sports news, literature, scientific articles, etc.

The corpus will be morphologically analyzed and disambiguated and syntactically parsed. The tagset used is PAROLE compliant, and consists of some 230 tags[1] fully expanded (using all information about gender, number, person, tense, etc.) which can be re-

[1] There are potentially many more possible tags, but they do not actually occur.

duced to 62 tags when only category and subcategory are considered.

This paper is organized as follows: In section 2 we describe the linguistic criteria used to develop the morphological analyzer MACO+, as well its main implementation issues. In section 3 we describe the two taggers which were used to disambiguate MACO+ output. Finally, in section 4 we outline how these tools are being used to annotate and disambiguate the LEXPESP corpus.

2 MACO+ Description

The construction of the MACO+ morphological analyzer consisted of two steps:

1. The set of inflectional rules used by the old MACO (Acebo *et al.*, 1994) for analysing each word was used (reverting the engine from analysis to generation) to generate, from a big root dictionary, all *possible* Spanish words (according to these rules) which were stored in a dictionary.

2. An efficient look-up procedure and other specific modules were written to exploit the data.

The implementation of MACO+ is Unix-PERL based. This makes it easily transportable and overcomes the first flaw of the first version.

The incorporation of new forms is always possible by generating them with the appropriate root and models, adding them to the form base and reindexing it.

The linguistic model followed to create the root dictionary and the inflectional rules, as well as the final word–form dictionary are described in section 2.1. Sections 2.2 and 2.3 are devoted to the description of the modules used for text segmentation and information retrieval.

2.1 Form Generation Linguistic Model

Linguistic data has been organized in order to generate all the inflexional word forms with their morphological attributes, their lemma, and all the possible interpretations.

We have taken in consideration three kinds of morphological information:

- The form segments (roots and suffixes) and the models they have associated.
- The lemma.
- One or more morphological attributes.

Words are considered to be composed by a root and an inflectional suffix. Each root and suffix is assigned a model (paradigm) of inflection, and all the correct combinations of models have been declared. The root and suffixes dictionaries have the structures described in tables 1 and 2. In addition, it must be stated that root model AM combines with suffix model IPU (to construct forms as *amo, amas,* etc.), NEF with FE (to construct *liebre, liebres,* etc.) and so on. The root dictionary, consisting of about 12,000 verbal roots, 85,000 nominal and adjectival roots and 3,000 closed–category roots, was automatically extracted from existing MRD's and available corpora. Models of inflection were semi-automatically assigned and validated. The suffix dictionary is quite small and was manually constructed.

Root	Lemma	Model
am-	amar	AM
salt-	saltar	AM
estudi-	estudiar	AM
liebr-	liebre	NEF
fiebr-	fiebre	NEF

Table 1: Organization of the root dictionary

Suffix	Model
-o	IPU
-as	IPU
-e	FE
-es	FE

Table 2: Organization of the suffix dictionary

Morphological attributes can be associated to roots, suffixes and models. When an attribute is associated to a model, it is valid for all the roots or suffixes belonging to it. This implies a generalization about the morphological behaviour of the language; when an attribute is assigned to a word segment, it is considered to be specific for it. For example the model AM has an associated information about the category (verb); the model IPU has associated information about tense (present) and mode (indicative), and the suffix '-o' has associated information about person and number (first and singular).

The linguistic analysis has been carried out following morpho-ortographic criteria because we have to analize written texts: each variant of a root has been declared in the dictionary withs its corresponding model. For instance, verbs like *dormir* (to sleep) has three roots: 'dorm-', 'durm-', 'duerm-', accounting for forms like *dormido* (slept), *durmiendo* (sleeping) and *duermo* (I sleep).

Derivation is very productive in Spanish, but we have not implemented it in our system because many problems for assigning the lemma would arise: it would have to be declared, for each step in the derivational process, which was the lemma and the rules to generate it. It doesn't seem appropiate because the system would loose its simplicity and it would probably overgenerate[2].

The total number of root models for nouns and adjectives is 29 and the number of root models for verbs is 6 for the first Spanish paradigm (verbs with infinitive ending in '-ar'), 18 for the second (verbs ending in

[2]A treatment of derivation based on lexical rules is planned to be incorporated in the short run.

'-er') and 21 for the third (verbs ending in '-ir'). The number of rules combining models of roots and models of suffixes is about 400. Irregular forms of verbs *ser* (to be), *haber* (to be) and *ir* (to go) have been solved one by one.

The generation of all possible forms was automatically performed in a MacIntosh platform and it took a few days of processing time. The result is the *Spanish Word Form Dictionary* (SWFD), a dictionary of about one million entries, containing for each form the lemma and a PAROLE compliant morphological tag describing information such as category, subcategory, gender, number, person, mode, etc.

The current SWFD[3] contains about 770,000 verbal entries and about 225,000 entries for nouns, adjectives and adverbials (inflection is much more productive for verbs in Spanish). The required disk space for the current codification is 21Mb.

2.2 Architecture of MACO+

The architecture of the morphological analyzer is a modular pipeline of specialized recognizers, as showed in figure 1.

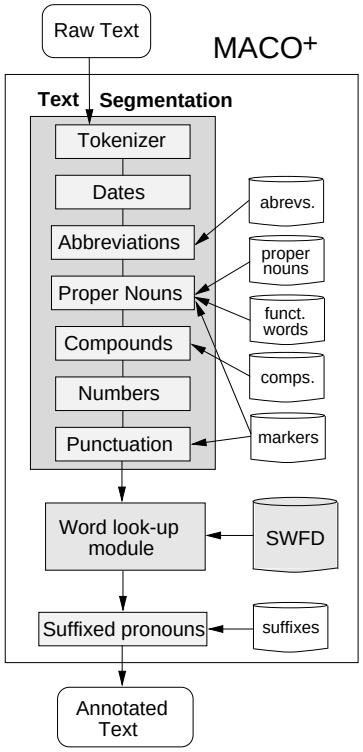

Figure 1: General architecture of MACO+

The first block of modules, labelled *Text Segmentation*, performs a proper segmentation of the text, labelling the punctuation marks and joining groups of words identified as one lexical unit (e.g. proper nouns or compounds such as *'aparte de'*, *'sin embargo'*), date or numerical expressions, etc. They are a collection of specific heuristics to identify the following special items:

- Simple date patterns: '23/3/79', *'año* 1983', '13 *de diciembre'*, '30 *de julio de* 1993', ...
- Abbreviations: *cm.*, *Hz.*, *Sr.*, ...
- Proper nouns: *'María Elena'*, *'San Cristóbal de las Casas'*, *'Ministerio de Cultura'*, *'Universidad de Lodz'*, ...
- Multi–word compounds: *'sin embargo'*, *'no obstante'*, ...
- Numbers and numerical expressions: '12,12', 11.000, 1-3-1, 33942206-S, ...
- Punctuation marks.

These modules use a set of files containing compilations of typical abbreviations, proper nouns (personal, geographical, marks, enterprises, etc.), multi–word compounds, functional words allowed to be inside compounds, punctuation marks, etc.

Modules can be activated or deactivated for each particular analysis and, obviously, heuristics in each module can be improved independently.

All the tokens not recognized by any of the preceding modules are pipelined to the word look–up module, which is the *real analyzer*, containing the fast algorithms for retrieving information from the SWFD. This module is described in section 2.3.

Finally, a post process is performed on the non-recognized words in order to identify verbal forms with suffixed pronouns (named *clíticos*). This is a type of pronouns that are added as suffixes to the verb forms, acting usually as syntactic objects. For instance, the form *dándonosla* ('giving it (*fem.*) to us') has two suffixed pronouns: '-nos' and '-la' indicating first person plural indirect object and 3rd person singular femenine direct object, respectively. These particular forms were not generated and included in the dictionary because there exist potentially infinite combinations due to the possible recursive application of suffixes. Even restricting to the combinations of two pronouns (which is a realistic simplification) would result in an unfeasible increase of the dictionary.

Words that remain unrecognized after the pipeline are labelled as *unknown*. Empirical results (see section 4.1) show that they are about 0.5% in a free Spanish text.

2.3 Word Look-up Module

The Word look-up Module architecture is presented in figure 2.

The search algorithm uses three sources of information in the order indicated below:

1. A hash table containing the non–content words.
2. A hash table containing the most frequent words.
3. A trie index for accessing the SWFD.

[3]The word form dictionary is continuously increased and improved as new errors and lacks are discovered.

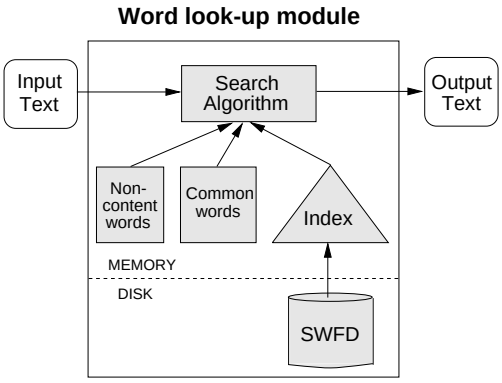

Figure 2: General architecture of the word look–up module

The list of most frequent words has been extracted from the LexEsp corpus and it consists of about 125,000 word forms. The list of non-content words contain about 400 entries and they have been manually written and validated. These lists are stored in hash structures in primary memory, so we have direct acces to all these forms. For the rest of forms we have a trie structure to access the SWFD. The trie is also in primary memory, so the access is also very fast.

Memory requirements of several architectures do not allow to load all this information in primary memory. In these cases only a portion of the list of most common words and a not completely expanded trie structure are loaded into memory. This means that the search of a word through the trie involves two steps: 1) reach the corresponding leaf and 2) perform a binary search in the part (ideally small) of the dictionary adressed by this leaf. This binary search is performed in disk, but if the part to look is relatively small, namely not greater than 2,5Kb, the average number of real seek operations (involving a physical access to disk blocks) is very small (about 2–3). In addition, this situation only occurs with words not previously found in the two hash tables, which are usually less than 5%.

In order to construct this partial trie structures a pruning process is performed to adjust its depth according to two basic parameters: the maximum size allowed for the portions of disk addressed by leaves and the representativity of this portions, that is, the importance (in terms of frequency) of the word forms they include. The idea is to have deeper branches –and thus, faster access– for common words than for rare words. We have performed several experiments on Unix/Linux architectures, including Sun–Sparc/UltraSparc workstations and Pentium processors, with different memory capacities (from 24Mb to 196Mb) to properly establish pruning parameters and to decide the best memory charges. The list of experiments and conclusions are described in detail in (Carmona, 1998).

As a result, a list of tries of increasing sizes were generated together with a set of configuration tables that decide which are the best choices (the size of frequent–words hash and the proper trie to load) depending on the amount of memory available and the size of the text to be analyzed.

To give an idea of the performance achieved, the selection for a Sun-Ultra1 workstation with a primary memory of 94Mb, when analyzing a text of more than 500,000 entries, was to load 5Mb of information in memory (3Mb for the hash of most common words and 2Mb for the dictionary index). The average number of real disk seeks per word was 0.14, and the retrieving speed about 9000 words/sec. (it is important to note that this time does not include output time in writting results, but only the time of searching and taking to memory the dictionary entries).

On the other hand, the speed achievable in a more modest machine is also noticeable: in a Pentium–120/24Mb configuration the average speed was almost 4000 words/sec.

Performance would be clearly improved in a C implementation, but current speed is enough for our purposes[4] and the PERL implementation makes the system more transportable since there is no need of re–compilation from one machine to another.

3 Morphosyntactic Disambiguation

The results produced by the morphological analyzer described in section 2 can be pipelined into a morphological disambiguator –POS tagger– to obtain the appropriate reading in the given context.

In the framework of the ITEM and LexEsp projects, two different POS taggers will be used to annotate the LexEsp corpus. First, a decision–tree based tagger (Màrquez & Rodríguez, 1997), which learns a language model from a tagged corpus, as well as prediction rules for the possible readings for words not found in the dictionary. Second, a relaxation labelling based tagger (Padró, 1998), which can use and combine information from different sources (n–gram, decision trees, manually written, etc.) provided it is put in the form of context constraints.

We are studying whether it is possible to take advantage of their collaboration and to integrate them inside GATE in a broader system oriented to information extraction.

3.1 A Tree Based Tagger

TreeTagger is a general Part-of-speech tagger that uses Statistical Decision Trees for disambiguating. It consists, basically, of two parts:

- A machine-learning supervised algorithm used for learning the base of statistical decision trees.
- An algorithm for combining these trees in order to disambiguate the text.

A general description of both parts is given below, however, we refer the reader to the two previ-

[4] MACO+, running with all modules, took 2.54 hours (including input/output processing time) to analyze the 5.5 million words LexEsp corpus in a Sun-Ultra2 workstation. (See section 4.1 for more details).

ous papers (Màrquez & Rodríguez, 1997; Màrquez & Rodríguez, 1998) for a detailed explanation.

Acquiring the Tree Base POS tagging can be seen as a problem of classification. In our case, classes are identified with tags and examples correspond to the words to disambiguate togeter with a set of features referring to its context of appearance. So 'classify a new example' is equivalent to decide which is the correct tag for the word in its particular context.

More particularly, we have grouped the whole set of examples into classes corresponding to the sets of tags they can take (i.e, 'noun-adjective', 'noun-adjective-verb', etc.). We call this sets *ambiguity classes* and we consider a classification problem for each of them. Decision trees (and in particular statistical decision trees), recently used in several NLP tasks, such as tagging (Schmid, 1994; Màrquez & Rodríguez, 1997; Daelemans *et al.*, 1996), parsing (McCarthy & Lehnert, 1995; Magerman, 1996), sense disambiguation (Mooney, 1996) and information extraction (Cardie, 1994), are a good tool for representing classification rules for each ambiguity class classification problem.

The algorithm used for acquiring the statistical decision trees is quite standard and belongs to the TDIDT (Top Down Induction of Decision Trees) family of machine learning supervised algorithms (Quinlan, 1993). The decision trees are acquired from annotated corpora and contain, basically, contextual and orthographical information: words and tags of a context window of six items, information about capitalization, prefixes, suffixes, etc. In some sense they represent statistical information about the distribution of tags and words in some relevant contexts.

Using the Model for Disambiguating Using the model described above, we have implemented a *reductionistic* tagger in the sense of Constraint Grammars (Karlsson *et al.*, 1995). In a first step a word-form frequency dictionary or a convenient morphological analyzer 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.

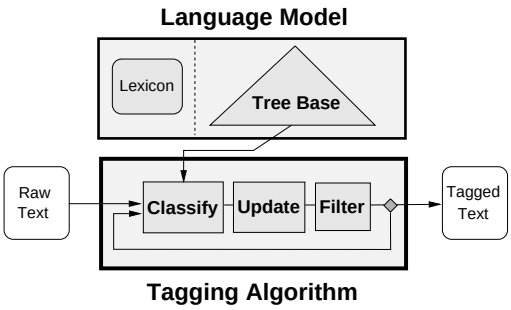

Figure 3: Architecture of TreeTagger

More particularly, at each step and for each ambiguous word (at a sentence level) the work to be done in parallel is: 1) Classify the word using the corresponding decision tree; 2) Use the resulting probability distribution to update the probability distribution of the word; and 3) Discard the tags with almost zero probability.

After the stopping criterion is satisfied some words could still remain ambiguous. Then there are two possibilities: 1) Choose the most–likely tags among the *survivors* to completely disambiguate the text. 2) Accept the residual ambiguity (for treating it in successive stages). 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 to get more accurate results.

Convergence properties have not been theoretically studied, so the convergence can not be guaranteed. However, empirical experiments suggest that convergence is usually reached in a moderate number of iterations and that the performance increases up to a unique maximum and then softly decreases as the number of iterations increases. For the experiments reported in the following sections, the number of iterations was simply fixed to three.

The tagger has been succesfully tested on the Wall Street Journal corpus with an accuracy over 97% and a speed between 300 and 600 words/sec. depending on the implementation.

3.2 A Relaxation Labelling Based Tagger

Relaxation labelling is a well-known technique used to solve consistent labelling problems (CLP). The algorithm finds a combination of values for a set of variables such that satisfies –to the maximum possible degree– a set of given constraints. Since CLPs are closely related to constraint satisfaction problems (Larrosa & Meseguer, 1995), relaxation labelling is a suitable algorithm to apply a constraint-based language model.

Relaxation operations had been long used in engineering fields to solve systems of equations (Southwell, 1940), but they got their biggest success when the extension to symbolic domain –relaxation labelling– was applied to constraint propagation field, specially in low-level vision problems (Waltz, 1975; Rosenfeld *et al.*, 1976). The possibility of applying it to NLP tasks was pointed out by (Pelillo & Refice, 1994) who use a toy POS tagging problem to evaluate their constraint compatibility estimating method. It has been applied more massively to NLP disambiguation tasks in (Padró, 1998).

The presented tagger has the architecture described in figure 4. It consists of an engine which applies the constraints contained in the language model in order to iteratively update the weights for each possible label for each word. If constraints are consistent, the algorithm converges to a local optimum which sat-

isfies as much as possible the constraint set. For a deeper discussion on the convergence of the algorithm, see (Padró, 1998).

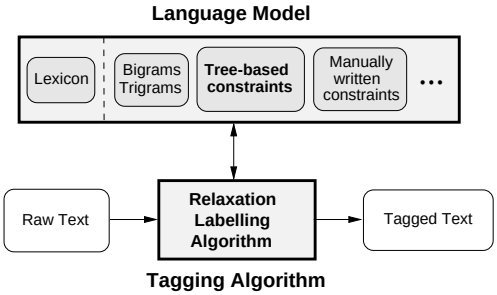

Figure 4: Architecture of the relaxation labelling based tagger

The main features of the RL based tagger are the following:

- It uses a constraint oriented language model. This enables it to deal with many kinds of information (n–grams, decision tree branches, linguistic information, etc.) provided they are expressed in the form of constraints.
- It performs parallel constraint satisfaction, that is, the constraints are not applied in a predefined order.
- It enables the use of heterogeneous information. For instance, to perform POS tagging, one can use constraints on the POS tags for words in context, but also their morphological, semantic o syntactic features, if available.
- It enables the simultaneous resolution of several disambiguation tasks. For instance, by choosing among a set of pairs *(tag, sense)* instead of among a set of *tags*, one can perform simultaneously POS tagging and WSD, and use constraints which take advantadge of the cross information between both tasks.

For more details on the algorithm and its application to different NLP tasks, such as WSD or shallow parsing, see (Padró, 1996; Màrquez & Padró, 1997; Voutilainen & Padró, 1997; Padró, 1998).

The algorithm has been used with different language models for English, combining statistical (n–grams), hand–written, and machine–learned (decision trees) information, producing accuracies over 97%. For Spanish, statistical and decision trees information are being used.

4 Annotating the LexEsp Spanish Corpus

4.1 Morphological Analysis

MACO+, with all modules activated, was run on a Sun Ultra2 workstation to analyze the whole LexEsp corpus. It took 2.54 hours (including input/output processing time) to analyze the 5.5 million words at an average speed of 600 words/sec. Comparatively, the time that would take running on a Pentium-120/24Mb architecture was estimated to be 7.64 hours, at an average speed of 200 words/sec.

The resulting coverage is about 99.5%, which is remarkable in a free text as LexEsp.

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.

The recall (words that get the correct tag among the proposed) is estimated to be 99.3%.

4.2 Morphosyntactic Disambiguation

Trained on a hand disambiguated subset of 70 Kw, and tested on a fresh, also hand-disambiguated, 25 Kw subset, the obtained results were those detailed in table 4.

The results marked RL are those produced by the Relaxation Labelling tagger, using different language models: B stands for bigram, T for trigram, and C for a constraint model obtained by writting in the form of constraints the decision tree branches acquired by the learning procedure of TreeTagger (TT). Baseline results obtained by a bigram HMM tagger (Elworthy 1993) and by a non contextual most–likely–tag tagger (MLT) on the same training and test corpus are presented in table 3.

Tagger	ambiguous	overall
MLT	88.48%	95.47%
HMM	91.67%	96.83%

Table 3: Results of baseline taggers

Tagger	ambiguous	overall
TT	91.77%	96.89%
RL-B	92.95%	97.33%
RL-T	92.67%	97.23%
RL-BT	93.14%	97.41%
RL-C	92.54%	97.18%
RL-BC	93.29%	97.46%
RL-TC	93.35%	97.49%
RL-BTC	93.61%	97.59%

Table 4: Results of our taggers using every combination of constraint kinds

4.3 Joint Use of both Taggers

The ratio of agreement between both taggers has been studied in order to establish whether it is possible to take advantage of this agreement to more accurately disambiguate POS in Spanish.

The procedure used for that starts by using a small hand-tagged portion of the corpus (about 70 Kw) as an initial training set.

Both taggers are then used to disambiguate further material (some 200Kw), which is used to enlarge the language model, incorporating it to the training set

and retraining the taggers. In order to minimize the errors in this automatically disambiguated portion of the new training set, only the cases where both taggers coincide are used, since experiments show that the error rate when both taggers coincide is significatively lower than that obtained by any of them separately.

This procedure can be iterated in a bootstrapping process that should lead to progressively better language models. For instance, using the new training set, one can re-estimate n-gram and tree models and use them to disambiguate 200Kw more, choose the words in which the taggers coincide, produce a larger training corpus, repeating until no improvement is produced.

The obtained results (see table 5) point that the precision when both taggers propose the same tag (TT = RL-BT) is higher (98.36%) than when only one tagger is used. Although this cannot be used to completely disambiguate a corpus, it may be useful as a way to automatically obtain larger training sets with a relatively small amount of noise.

Anyway, if we accept a certain ambiguity in the tagger output, the combination of the outputs of both taggers will obviously produce a higher recall. This result can be found in the row marked as TT $\cup$ RL-BT in table 5. It corresponds to the precision/recall of a tagger that proposes a unique tag when TT and RL-BT coincide, and two tags when they do not. This *voting taggers* approach may easily be extended to a larger number of taggers as we plan to do in the short run.

Tagger	ambiguous	overall
TT	91.77%	96.89%
RL-BT	93.14%	97.41%
TT = RL-BT	95.54%	98.38%
	prec – recall	prec – recall
TT $\cup$ RL-BT	89.75% – 95.65%	95.97% – 98.36%

Table 5: Results of tagger colaboration

Given that the cases in which both taggers coincide in their predictions represent over 90%, by using only those cases we obtain a large enough reasonably accurate new training corpus. Nevertheless if one wants to exploit the cases in which the taggers disagree, it is possible to hand analyze them with a low effort since they represent a small percentage and in most cases one of the two taggers proposes the right tag, thus reducing the hand disambiguating taks to a binary choice.

For instance, using a first new set of 200Kw and given that both taggers agree in 97.5% of the cases and that 98.38 of those cases are correctly tagged, we get a new corpus of 195Kw with an error rate of 1.62%. If we add the 70Kw manually tagged (assumed error free) from the initial training corpus we get a 265Kw corpus with an 1.19 error rate. By hand correcting the ambiguous words of the 5000 disagreement cases (totalling 1963 given the 39.26% ambiguity ratio) and adding them to the previous set we finally obtain a 270Kw corpus with a 1.17 error rate, which can be used to retrain the taggers.

As stated above, we hope that this increase of the training corpus size will result in higher tagger performances, in spite of the noise introduced.

5 Summary and Further Work

In this paper we have described a fast and accurate morphological analyzer for unrestricted Spanish text.

The output of the analyzer can be input to a POS tagger. We have described and tested two of them, which are currently being used to develop from scratch a disambiguated corpus of Spanish of over 5 Mw.

Performed experiments show that the precision when both taggers coincide is significatively higher than the results obtained by any of them separately. Further work will focus on to what extent this can be used to build large training corpus keeping the noise to a minimum level, and in using more than two taggers in the voting collaboration approach.

6 Acknowledgments

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).

References

Acebo, S.; Ageno, A.; Climent, S.; Farreres, X.; Padró, L.; Ribas, F.; Rodríguez, H. & Soler, O. (1994). MACO: Morphological Analyzer Corpus-Oriented. ESPRIT BRA-7315 Acquilex II, Working Paper #31.

Cardie, C. (1994). *Domain Specific Knowledge Acquisition for Conceptual Sentence Analysis.* PhD Thesis, University of Massachusets, Amherst, MA.

Carmona, J. (1998). *Un sistema eficient de representacio i accés a grans volums d'informació lèxica.* Technical Report. Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya (forthcoming).

Cunningham, H.; Wilks, Y. & Gaizauskas, R. (1996). GATE - a General Architecture for Text Engineering. In *Proceedings of 16th International Conference on Computational Linguistics*, COLING '96. Copenhagen, Denmark.

Daelemans, W.; Zavrel, J.; Berck, P. & Gillis, S. (1996). MTB: A Memory–Based Part–of–Speech Tagger Generator. In *Proceedings of 4th Workshop on Very Large Corpora, Copenhagen.*

Elworthy, D. (1993). Part–of–Speech and Phrasal Tagging. Technical Report, ESPRIT BRA–7315 Acquilex II, WP #10.

Karlsson, F.; Voutilainen, A.; Heikkilä, J. & Anttila, A. (1995). *Constraint Grammar. A Language-Independent System for Parsing Unrestricted Text.* Mouton de Gruyter.

Larrosa, J. & Meseguer, P. (1995) Constraint Satisfaction as Global Optimization. In *Proceedings of 14th International Joint Conference on Artificial Intelligence*, IJCAI '95.

Magerman, M. (1996). Learning Grammatical Structure Using Statistical Decision–Trees. In *Proceedings of the 3rd International Colloquium on Grammatical Inference, ICGI '96.* Lecture Notes in Artificial Intelligence 1147.

Màrquez, L. & Padró, L. (1997). A Flexible POS Tagger Using an Automatically Acquired Language Model. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, E/ACL '97. Madrid, Spain.

Màrquez, L. & Rodríguez, H. (1997). 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. Tzigov Chark, Bulgaria.

Màrquez, L. & Rodríguez, H. (1998). Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the 10th European Conference on Machine Learning*, ECML'98. Chemnitz, Germany.

McCarthy, J. & Lehnert, W. (1997). Using Decision Trees for Coreference Resolution. In *Proceedings of 14th International Joint Conference on Artificial Intelligence*, IJCAI '95.

Mooney, R. (1996). Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning. In *Proceedings of Conference on Empirical Methods in NLP*, EMNLP '96.

Padró, L. (1996). POS Tagging Using Relaxation Labelling. In *Proceedings of 16th International Conference on Computational Linguistics*, COLING '96. Copenhagen, Denmark.

Padró, L. (1998). *A Hybrid Environment for Syntax-Semantic Tagging.* PhD Thesis. Dept. Llenguatges i Sistemes Informatics. Universitat Politecnica de Catalunya. Barcelona.

Pelillo, M. & Refice, M. (1994). Learning Compatibility Coefficients for Relaxation Labeling Processes. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.16, n.9.

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

Rosenfeld, R.; Hummel, R. & Zucker, S. (1976). Scene labelling by relaxation operations. In *IEEE Transactions on Systems, Man and Cybernetics*, Vol.6, n.6.

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

Southwell, R. (1940). *Relaxation Methods in Engineering Science.* Clarendon.

Voutilainen, A. & Padró, L. (1997). Developing a Hybrid NP parser. In *Proceedings 5th ACL Conference on Applied Natural Language Processing*, ANLP '97, Washington DC.

Waltz, D. (1975). Understanding line drawings of scenes with shadows. *Psycology of Computer Vision.* P. Winston, New York: McGraw-Hill.