Procesamiento del Lenguaje Natural, Revista nº 40, marzo de 2008, pp. 59-66 recibido 30-01-08, aceptado 03-03-08

Experiments with an ensemble of Spanish dependency parsers*

Experimentos con un sistema combinado de analizadores sintácticos de dependencias para el español

Roser Morante Vallejo
Tilburg University
Postbus 90153, 5000 LE Tilburg, The Netherlands
R.Morante@uvt.nl

Resumen: Este artículo presenta un sistema combinado de analizadores sintácticos de dependencias del español que integra tres analizadores basados en aprendizaje automático. El sistema opera en dos etapas. En la primera cada analizador procesa una frase y produce un grafo de dependencias. En la segunda un sistema de votación decide cual es el análisis final a partir de los análisis producidos en la primera etapa.
Palabras clave: Analizadores sintácticos de dependencias, sistema combinado, MaltParser, aprendizaje basado en memoria.

Abstract: This article presents an ensemble system for dependency parsing of Spanish that combines three machine-learning-based dependency parsers. The system operates in two stages. In the first stage, each of the three parsers analyzes an input sentence and produces a dependency graph. In the second stage, a voting system distills a final dependency graph out of the three first-stage dependency graphs.
Keywords: Dependency parsers, ensemble system, MaltParser, memory-based learning.

1 Introduction

This article presents the results of experiments with an ensemble system for dependency parsing of Spanish. The system has been developed as part of the project *Técnicas semiautomáticas para el etiquetado de roles semánticos en corpus del español*, which focuses on researching semiautomatic techniques for semantic role labeling. The final goal of the project is to annotate with semantic roles a seventy million word corpus, starting from an eighty thousand word train corpus. It is well known that semantic role labelers that use syntactic information perform better. This is why a parser is needed in the project that performs as accurately as possible. Since parser combination has proved to improve the performance of individual parsers (Henderson and Brill, 1999; Zeman and Žabokrtský, 2005; Sagae and Lavie, 2006), experimenting with an ensemble of parsers that integrates one of the best dependency parsers for Spanish (MaltParser) seemed to be an appropriate first step.

The system combines three machine-learning-based dependency parsers: Nivre's MaltParser (Nivre, 2006; Nivre et al., 2006), Canisius' memory-based constraint-satisfaction inference parser (Canisius and Tjong Kim Sang, 2007), and a new memory-based parser that operates with a single word-pair relation classifier.

Like in Sagae and Lavie (2006), the ensemble system operates in two stages. In the first stage, each of the three parsers analyzes an input sentence and produces a dependency graph. The unlabeled attachment scores in this stage range from 82 to 87 %, according to the evaluation metrics used in the CoNLL Shared Task 2006 (Buchholz and Marsi, 2006). In the second stage, a voting system distills a final dependency graph out of the three first-stage dependency graphs. The system achieves a 4.44% error reduction over the best parser.

* This research has been funded by the postdoctoral grant EX2005–1145 awarded by the Ministerio de Educación y Ciencia of Spain to the project *Técnicas semiautomáticas para el etiquetado de roles semánticos en corpus del español.*

ISSN 1135-5948

N.	FORM	LEMMA	CPOS	POS	FEATS	HEAD	DEP.REL
1	Asimismo	asimismo	r	rg	_	2	MOD
2	defiende	defender	v	vm	num=s|per=3|mod=i|tmp=p	0	ROOT
3	la	el	d	da	num=s|gen=f	4	ESP
4	financiación	financiación	n	nc	num=s|gen=f	2	CD
5	pública	pública	a	aq	num=s|gen=f	4	CN
6	de	de	s	sp	for=s	4	CN
7	la	el	d	da	num=s|gen=f	8	ESP
8	investigación	investigación	n	nc	num=s|gen=f	6	_
9	básica	básico	a	aq	num=s|gen=f	8	CN
10	y	y	c	cc	_	2	CTE
11	pone	poner	v	vm	num=s|per=3|mod=i|tmp=p	10	CDO
12	de	de	s	sp	for=s	11	CC
13	manifiesto	manifiesto	n	nc	gen=m|num=s	12	_
14	que	que	c	cs	_	18	_
15	las	el	d	da	gen=f|num=p	16	ESP
16	empresas	empresa	n	nc	gen=f|num=p	18	SUJ
17	se	él	p	p0	per=3	18	_
18	centran	centrar	v	vm	num=p|per=3|mod=i|tmp=p	11	CD
19	más	más	r	rg	_	20	_
20	en	en	s	sp	for=s	18	CREG
21	la	el	d	da	num=s|gen=f	22	ESP
22	I+D	I+D	n	np	_	20	_
23	con	con	s	sp	for=s	18	CC
24	objetivos	objetivo	n	nc	gen=m|num=p	23	_
25	de	de	s	sp	for=s	24	CN
26	mercado	mercado	n	nc	gen=m|num=s	25	_
27	.	.	F	Fp	_	2	PUNC

Table 1: Example sentence of the revised Cast3LB–CoNLL corpus of Spanish.

The results presented here are preliminary. Because the MaltParser performs substantially better than the other two parsers, the results of the ensemble do not improve significantly over the results of the MaltParser. Consequently, more parsers will have to be added to the ensemble, and additional combination techniques will have to be experimented.

The article is structured as follows. The corpus used is described in Section 2. Section 3 presents the parsers that were integrated in the ensemble, which is introduced in Section 4. The results are reported in Section 5, and compared to related work in Section 6. Finally, some conclusions are put forward in Section 7.

2 The Cast3LB–CoNLL corpus of Spanish

The experiments described in this paper were carried out on the Cast3LB–CoNLL Corpus of Spanish (Morante, 2006), which is a revised version of the Cast3LB treebank (Civit, Martí, and Bufí, 2006; Civit, 2003; Navarro et al., 2003) used in the CoNLL Shared Task 2006 (Buchholz and Marsi, 2006). It contains 89199 words in 3303 sentences. As for verbs, it contains 11023 forms, and 1443 lemmas, and not all verbs are equally frequent[1].

Table 1 shows an example sentence of the corpus. Like in the CoNLL Shared Task 2006 sentences are separated by a blank line and fields are separated by a single tab character. A sentence consists of tokens, each one starting on a new line. A token consists of the following 8 fields that contain information about morphosyntactic features and non-projective dependencies:

1. ID: token counter, starting at 1 for each new sentence.
2. FORM: word form or punctuation symbol.
3. LEMMA: lemma of word form.
4. CPOSTAG: coarse-grained part-of-speech tag.
5. POSTAG: fine-grained part-of-speech tag.

[1] 1369 verbs appear less than 20 times; 54 verbs, from 20 to 50 times; 12 verbs, 50 to 100 times: *tratar* (51), *dejar* (53), *acabar* (55), *pasar* (59), *parecer* (62), *seguir* (62), *quedar* (67), *encontrar* (68), *llevar* (68), *poner* (68), *deber* (75), *querer* (78), *dar* (86). 6 verbs, from 100 to 300 times: *saber* (101), *llegar* (107), *ver* (121), *ir* (132), *decir* (210), *tener* (243), *hacer* (253), *poder* (282), *estar* (296); and 2 verbs appear more than 800 times: *ser*, 1348 times and *haber*, 812 times.

6. FEATS: unordered set of syntactic and/or morphological features, separated by a vertical bar. If features are not available, the value of the feature is an underscore.

   The complete description of the CPOSTAG, POSTAG, and FEATS tags can be found in Civit (2002).

7. HEAD: head of the current token, which is either a value of ID or zero ('0') for the sentence root.

8. DEPREL: dependency relation to the HEAD. The set of tags is described in Morante (2006).

3 Single parsers

This section describes the parsers that were integrated into the ensemble system and their results.

3.1 MaltParser 0.4 (MP)

The MaltParser 0.4[2] (Nivre, 2006; Nivre et al., 2006) is an inductive dependency parser that, according to Nivre et al. (2006), uses four essential components: a deterministic algorithm for building labeled projective dependency graphs; history-based feature models for predicting the next parser action; support vector machines for mapping histories to parser actions; and graph transformations for recovering non-projective structures.

The MaltParser participated in the CoNLL-X Shared Task on multi-lingual dependency parsing obtaining the second best results for Spanish (81.29 % labeled attachment score). In these experiments we used the following model for Spanish:

The learner type was support vector machines (LIBSVM (Chang and Lin, 2005)), with the same parameter options used by Nivre et al. (2006) in the CoNLL Shared Task 2006. The parser algorithm used was Nivre, with the options arc order eager, shift before reduce and allow reduction of unattached tokens.

3.2 Memory-based constraint satisfaction parser (MB1)

The memory-based constraint satisfaction parser (Canisius and Tjong Kim Sang, 2007) uses three memory-based classifiers that predict weighted soft-constraints on the structure of the parse tree. Each predicted constraint covers a small part of the complete dependency tree, and overlap between them ensures that global output structure is taken into account. A dynamic programming algorithm for dependency parsing is used to find the optimal solution to the constraint satisfaction problem thus obtained.

POS	STACK				
POS	INPUT				
POS	INPUT	1			
POS	INPUT	2			
POS	INPUT	3			
POS	STACK	1			
POS	STACK	0	0	1	
POS	STACK	0	0	0	-1
POS	STACK	0	0	0	1
POS	INPUT	0	0	0	-1
POS	STACK	0	1		
POS	INPUT	0	-1		
POS	STACK	2			
FEATS	STACK				
FEATS	INPUT				
FEATS	INPUT	1			
FEATS	STACK	0	0	1	
DEP	STACK				
DEP	STACK	0	0	0	-1
DEP	STACK	0	0	0	1
DEP	INPUT	0	0	0	-1
LEX	STACK				
LEX	INPUT				
LEMMA	STACK				
LEMMA	INPUT				
LEMMA	INPUT	0	0	0	-1
CPOS	STACK				
CPOS	INPUT				
CPOS	INPUT	1			
CPOS	STACK	0	0	1	

Table 2: Model of the MaltParser used.

3.3 Memory-based single classifier parser (MB2)

The memory-based single classifier parser is a new parser developed for performing the experiments reported here. It consists of a single classifier that predicts the relation between two words in a sentence, and a decision heuristics that chooses among the dependency relations that the classifier has predicted for one word, based on information from the classifier output.

Given two words, w1 and w2, the task that the classifier performs is predicting at the same time the direction of the dependency and the type of dependency. A dummy class NONE represents absence of

[2]Web page of MaltParser 0.4: http://w3.msi.vxu.se/∼nivre/research/MaltParser.html.

relation. For a sentence like *El gato come pescado*, the instances in the train corpus would be:

w1:el w2:gato features class
w1:el w2:come features class
w1:el w2:pescado features class
w1:gato w2:come features class
w1:gato w2:pescado features class
w1:come w2:pescado features class

An instance is composed of the following features:

- Lemma, POS, CPOS gender, number, person, mode, tense of the focus word w1 and focus word w2, and of the two previous and two next words to the focus words.
- Features that express if w2 is placed between w1 and the first coordination / main verb / preposition / noun / adjective to the right of w1.
- Features that expresses if w2 is placed between w1 and the second coordination / main verb / preposition / noun / adjective to the right of w1.
- Features that expresses if w1 is placed between w2 and the first coordination / main verb / preposition / noun / adjective to the left of w2.
- Features that expresses if w1 is placed between w2 and the second coordination / main verb / preposition / noun / adjective to the left of w2.
- Number of coordinative conjunctions, subordinate conjunctions, prepositions, punctuation signs, main verbs, auxiliary verbs, pronouns, relative pronouns, nouns, and adjectives.

We performed 10-fold cross-validation experiments. Instances with the NONE class in the train corpus were downsampled in a 1:1 proportion.

We use the IB1 classifier as implemented in TiMBL (version 6.0) (Daelemans et al., 2007), a supervised inductive algorithm for learning classification tasks based on the k-nearest neighbor classification rule (Cover and Hart, 1967). In IB1, similarity is defined by a feature-level distance metric between a test instance and a memorized example. The metric combines a per-feature value distance metric with global feature weights that account for relative differences in discriminative power of the features. The IB1 algorithm was parametrized by using Overlap as the similarity metric, Information Gain for feature weighting, 11 k-nearest neighbors, and weighting the class vote of neighbors as a function of their inverse linear distance (Daelemans et al., 2007).

Because the classifier might predict more than one dependency relation for one word, a decision heuristics is applied in order to disambiguate. The decision heuristics uses information about the class distribution and the distance to the nearest neighbor produced by TiMBL.

Algorithm 1 Heuristics to filter the output of the classifier in MB1.

```
if the predicted class is different than NONE then
    if there is not a NONE class among the nearest neighbors then
        if the distance is bigger than 6 then
            turn the prediction into NONE;
        else
            keep the predicted and tag it with a "not-none" flag;
        end if
    else if there is a NONE class among the nearest neighbors then
        if its class distribution is bigger than 0.70, and the difference between the probability of the predicted class and the NONE class is lower than 3 then
            turn the prediction into NONE;
        else
            keep the predicted class and tag it with a "possible-none" flag;
        end if
    end if
else
    keep the NONE prediction;
end if
```

In the first step the output of the classifier is filtered according to Algorithm 1. In the second step the dependency tree is reconstructed and the dependency relations are disambiguated, if more than one dependency is predicted for a word. The system gives preference to the class tagged with a "not-none" flag that has the lower distance to the nearest neighbor. If no classes are tagged with the "not-none" flag, the system gives preference to the class tagged with a "possible-none" flag that has the lower distance to the nearest neighbor.

DEPREL	n.train	MP			MB1			MB2		
		rec	prec	F1	rec	prec	F1	rec	prec	F1
AP	64	45.31	54.72	49.57	40.62	50.00	44.82	51.56	55.00	**53.22**
ATR	142	79.58	84.96	**82.18**	79.58	75.33	77.39	80.28	73.55	76.76
AUX	92	95.65	93.62	**94.62**	86.96	93.02	89.88	90.22	86.46	88.29
CA	152	72.37	72.37	**72.37**	63.16	66.67	64.86	69.08	61.05	64.81
CAG	4	50.00	66.67	**57.14**	50.00	50.00	50.00	50.00	66.67	**57.14**
CC	660	71.67	63.15	**67.14**	53.64	54.29	53.96	48.48	61.19	54.09
CD	450	78.89	71.43	**74.97**	74.44	70.38	72.35	72.44	69.81	71.10
CDO	326	70.86	66.38	**68.54**	66.56	58.49	62.26	71.47	54.31	61.71
CI	67	56.72	79.17	**66.09**	50.75	68.00	58.12	53.73	60.00	56.69
CN	1171	82.49	80.10	**81.27**	81.81	72.80	77.04	83.60	73.33	78.12
CPRED.CD	9	33.33	75.00	**46.15**	0.00	0.00	0.00	0.00	0.00	0.00
CPRED.SUJ	28	57.14	72.73	**63.99**	42.86	70.59	53.33	0.00	0.00	0.00
CREG	83	57.83	67.61	**62.33**	33.73	66.67	44.79	51.81	55.13	53.41
CTE	263	61.22	62.65	**61.92**	55.13	54.51	54.81	55.51	54.28	54.88
ENUM	3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ESP	1313	94.59	92.89	**93.73**	95.05	92.10	93.55	93.60	91.58	92.57
ET	68	41.18	54.90	47.06	41.18	65.12	**50.45**	29.41	31.75	30.53
IMPERS	11	81.82	69.23	75.00	63.64	87.50	73.68	81.82	75.00	**78.26**
MOD	50	42.00	72.41	**53.16**	36.00	66.67	46.75	42.00	48.84	45.16
NEG	76	84.21	88.89	86.48	85.53	89.04	**87.24**	85.53	82.28	83.87
PASS	35	85.71	90.91	**88.23**	48.57	68.00	56.66	85.71	88.24	86.95
PER	64	73.44	75.81	**74.60**	65.62	76.36	70.58	89.06	55.34	68.26
ROOT	331	91.54	71.46	**80.26**	76.74	74.05	75.37	61.03	85.59	71.25
SUJ	532	75.75	80.12	**77.87**	68.80	72.76	70.72	65.79	74.63	69.93
-	1896	82.70	90.64	**86.48**	80.80	84.69	82.69	81.75	84.19	82.95

Table 3: Precision, recall and F1 of MP, MB1 and MB2 per dependency relation.

3.4 Results of the individual parsers

Table 3 shows precision, recall, and F1 of each of the single parsers per syntactic function. The *n.train* column contains the number of instances that have a certain dependency relation in the train corpus. The MP has the best F1 for 19 of the 25 dependency relations. This fact indicates that it is difficult to improve over the MP results with the ensemble system. MB1 has the best F1 for dependency relation ET and NEG, and MB2 for AP and IMPERS.

	MP	**MB1**	**MB2**
LAS	80.45 %	75.74 %	75.44 %
UAS	87.42 %	82.44 %	82.75 %
LAc	85.12 %	81.95 %	81.35 %

Table 4: Results of the individual parsers.

The global results of the three parsers are shown in Table 4 in terms of Labeled Attachment Score (LAS), Unlabeled Attachment Score (UAS), and Label Accuracy (LAc) according to the evaluation metrics used in the CoNLL Shared Task 2006 (Buchholz and Marsi, 2006). The MP performs significantly better than MB1 and MB2, whereas MB1 and MB2 perform similarly in spite of the fact that their approach to memory-based learning is different: MB1 applies constraint satisfaction, and MB2 is based on only one classifier and heuristics that rely on the distance of the predicted class to the nearest neighbor and on the class distribution.

4 Ensemble dependency parser

The ensemble system operates in two stages. In the first stage, each of the three parsers analyzes an input sentence and produces a dependency graph. The results of the individual parsers were presented in Table 4 in the previous section. In the second stage, a voting system distills a final dependency graph out of the three first-stage dependency graphs. Voting techniques have been previously applied to dependency parsing (Sagae and Lavie, 2006; Zeman and Žabokrtský, 2005).

We provide results of three different voting systems, that take into account agreement among classifiers and/or the normalized F1 value of each classifier for each dependency

relation:

- **VS1**: the system votes for the solution of the single classifier that has the higher F1 for the dependency relation that the single classifier predicts.
- **VS2**: the system votes for the solution of the MP, unless MB1 and MB2 agree, in which case the MB1 and MB2 solution is chosen.
- **VS3**: the system votes for the solution of the MP, unless MB1 and MB2 agree or the three parsers disagree. In the first case, the MB1 and MB2 solution is chosen, and in the second, the system votes for the solution of the single classifier that has the higher F1 for the syntactic function that the single classifier predicts.
- **VS4**: the system votes for system VS1 unless two single systems agree. In this case, the system votes for the solution agreed by them.

As Sagae and Lavie (2006) point out "This very simple scheme guarantees that the final set of dependencies will have as many votes as possible, but it does not guarantee that the final voted set of dependencies will be a well–formed dependency tree". We are aware of this limitation. Future research will focus on converting the resulting graph into a well-formed tree.

5 Results

The results of the different versions of the ensemble system are presented in Tables 5, 6, 7, and 8, as well as the improvement over the MP. Results show that combined systems VS1, VS2 and VS3 perform better than the best parser, although the difference is insignificant, since it reduces the error of MP in less than 5% (4.44%). Combined system VS4 improves only in accuracy over the results of the best system.

	VS1	**VS1 vs MP**
LAS	80.53%	+0.08
UAS	87.43%	+0.01
LAc	85.22%	+0.10

Table 5: LAS, UAS, and LAc of VS1.

	VS2	**VS2 vs MP**
LAS	81.04%	+0.59
UAS	87.68%	+0.26
LAc	85.71%	+0.59

Table 6: LAS, UAS, and LAc of VS2.

	VS3	**VS3 vs MP**
LAS	81.09%	+0.64
UAS	87.68%	+0.26
LAc	85.78%	+0.66

Table 7: LAS, UAS, and LAc of VS3.

	VS3	**VS3 vs MP**
LAS	79.71%	-0.74
UAS	86.07%	-1.35
LAc	85.92%	+0.80

Table 8: LAS, UAS, and LAc of VS4.

VS1 is the system that improves the least because the MP has the better F1 scores for 19 of the 25 dependency relations. That VS2 and VS3 do no improve significantly might be due to the fact that some agreement cases between MB1 and MB2 can be errors.

VS3 is the voting system that performs better: by voting for the agreement between MB1 and MB2, or for the system with higher F1 in case of complete disagreement, more errors are eliminated than errors are introduced. For further research it would be interesting to analyze if it is possible to eliminate more errors by introducing specific voting strategies per dependency relation.

Table 9 shows that precision and recall in VS3 increase for some dependency relations (AP, ATR, CD, NEG, PASS, PER, SUJ), as compared to precision and recall per dependency relation of the MaltParser, although they also decrease for other (AUX, CC, ET).

6 Related work

The related work we are aware of deals with languages other than Spanish. Zeman and Žabokrtský (2005) tested several approaches for combining dependency parsers for Czech. They found that the best method was accuracy-aware voting, which reduced the error of the best parser in 13%. Differences between their approach an ours are that

	MP		VS3	
	rec	prec	rec	prec
AP	45.31	54.72	+7.81	+1.60
ATR	79.58	84.96	+4.93	+2.20
AUX	95.65	93.62	-1.08	-0.07
CA	72.37	72.37	+0.66	-4.69
CAG	50.00	66.67	0.00	0.00
CC	71.67	63.15	-5.76	-1.97
CD	78.89	71.43	+0.84	+3.99
CDO	70.86	66.38	+3.68	-1.23
CI	56.72	79.17	+2.98	-2.25
CN	82.49	80.10	+1.71	-2.03
CPRED.CD	33.33	75.00	-11.11	+0.25
CPRED.SUJ	57.14	72.73	-3.57	+6.22
CREG	57.83	67.61	-2.41	+1.05
CTE	61.22	62.65	0.00	-0.96
ENUM	0.00	0.00	0.00	0.00
ESP	94.59	92.89	+0.99	+0.06
ET	41.18	54.90	-2.94	-3.92
IMPERS	81.82	69.23	0.00	+12.59
MOD	42.00	72.41	0.00	+2.59
NEG	84.21	88.89	+2.63	+2.78
PASS	85.71	90.91	+5.72	+0.52
PER	73.44	75.81	+6.25	+0.31
ROOT	91.54	71.46	-1.21	+6.81
SUJ	75.75	80.12	+2.82	+2.65
-	82.70	90.64	+0.69	+0.33

Table 9: Recall and precision of VS3 compared to precision and recall of MP per dependency relation.

they experiment with seven parsers, they perform stacking, and they check that the resulting structure is a well-formed tree.

Sagae and Lavie (2006) experiment with six parsers on the Wall Street Journal corpus. They apply a two stage procedure of reparsing focusing on unlabeled dependencies. In the first stage, m different parsers analyze an input sentence. In the second stage, a parsing algorithm is applied taking into account the analyses produced by each parser in the first stage. They reparse the sentence based on the output of m parsers in order to maximize the number of votes for a well-formed dependency structure. Their experiments increase the accuracy of the best parser in 1.7%.

Nivre et al. (2007) combined the outputs of the parsers participating in the CoNLL Shared Task 2007 on dependency parsing using the method of Sagae and Lavie (2006). They show that accuracy never falls below the performance of the top three systems, although it degrades after ten different parsers have been added.

7 Conclusions and future research

In this paper we presented an ensemble system for dependency parsing of Spanish that combines three machine-learning-based dependency parsers. As far as we know, this is the first attempt to combine dependency parsers for Spanish.

The results of the ensemble of parsers are only slightly better than the results of the best parser; the error reduction of the label accuracy score reaches 4.44%. This is due to the fact that there are only three parsers, one of which performs clearly better than the other two, which perform very similarly. The best results were obtained by the voting system that gives priority to the decisions of the best parser, unless the other two parsers agree, in which case their solution is chosen, or the three parsers disagree, in which case the system votes for the solution of the single classifier that has the higher F1 for the dependency relation that the single classifier predicts.

We consider the results to be promising enough to continue our research. In the future we will integrate more parsers in the ensemble and we will explore additional combination techniques, like metalearning, and additional voting strategies that allow us to build well-constructed trees.

Acknowledgements

We would like to thank Sander Canisius and Joakim Nivre for making their parsers available and for being very helpful. Thanks also to the three anonymous reviewers for their valuable comments.

References

Buchholz, S. and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the X CoNLL Shared Task*. SIGNLL.

Canisius, S. and E. Tjong Kim Sang. 2007. A constraint satisfaction approach to dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1124–1128.

Chang, C.C. and C.J. Lin. 2005. LIBSVM: A library for support vector machines. URL:http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf.

Civit, M. 2002. Guía para la anotación morfológica del corpus CLiC-TALP (versión 3). X-TRACT-II WP-00-06, CliC–UB.

Civit, M. 2003. Guía para la anotación sintáctica de Cast3LB: un corpus del español con anotación sintáctica, semántica y pragmática. X-TRACT-II WP-03-06 y 3LB-WP-02-01, CliC–UB.

Civit, M., M.A. Martí, and N. Bufí, 2006. *Advances in Natural Language Processing (LNAI, 4139)*, chapter Cat3LB and Cast3LB: from Constituents to dependencies, pages 141–153. Springer Verlag, Berlin.

Cover, T. M. and P. E. Hart. 1967. Nearest neighbor pattern classification. *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, 13:21–27.

Daelemans, W., J. Zavrel, K. van der Sloot, and A. van den Bosch. 2007. TiMBL: Tilburg memory based learner, version 6, reference guide. Technical Report Series ILK 07-03, Tilburg University, Tilburg, The Netherlands.

Henderson, J. and E. Brill. 1999. Exploiting diversity in natural language processing: combining parsers. In *Proceedings of the Fourth Conference on Empirical Methods in Natural Language Processing (EMNLP)*, College Park, Maryland.

Morante, R. 2006. Semantic role annotation in the Cast3LB-CoNNL-SemRol corpus. Induction of Linguistic Knowledge Research Group Technical Report ILK 06-03, Tilburg University, Tilburg.

Navarro, B., M. Civit, M.A. Martí, R. Marcos, and B. Fernández. 2003. Syntactic, semantic and pragmatic annotation in cast3lb. In *Proceedigns of the Shallow Processing of Large Corpora (SProLaC) Workshop of Corpus Linguistics 2003*, Lancaster,UK.

Nivre, J. 2006. *Inductive Dependency Parsing*. Springer.

Nivre, J., J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL-2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague.

Nivre, J., J. Hall, J. Nilsson, G. Eryigit, and S. Marinov. 2006. Labeled pseudo–projective dependency parsing with support vector machines. In *Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X*, New York City, NY, June.

Sagae, K. and A. Lavie. 2006. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference on the North American Chapter of the ACL*, pages 129–132, New York. ACL.

Zeman, D. and Z. Žabokrtský. 2005. Improving parsing accuracy by combining diverse dependency parsers. In *Proceedings of the International Workshop on Parsing Technologies*, Vancouver, Canada.