Procesamiento del Lenguaje Natural, Revista nº 49 septiembre de 2012, pp 139-146 recibido 04-05-12 revisado 01-06-12 aceptado 05-06-12

# A Hybrid Approach to Treebank Construction

## *Una aproximación híbrida a la construcción de treebanks*

**Montserrat Marimon**
Dpt. de Lingüística General
Universitat de Barcelona
Barcelona, Spain
montserrat.marimon@ub.edu

**Lluís Padró**
TALP Research Center
Universitat Politècnica de Catalunya
Barcelona, Spain
padro@lsi.upc.edu

**Resumen:** Este artículo describe investigación sobre los efectos de la desambiguación morfosintáctica usada como un preproceso de un analizador sintáctico profundo basado en HPSG, en el contexto del desarrollo de un *treebank* del español de código abierto, en el entorno de DELPH-IN. La anotación treebank se realiza manualmente tomando las decisiones apropiadas entre las opciones propuestas por el sistema y ordenadas por un módulo estadístico. Los experimentos presentados muestran que el uso de un etiquetador reduce la ambigüedad de las frases, y contribuye a limitar la cantidad de frases cuyo análisis sobrepasaa el límite de tiempo, y ayuda a al módulo estadístico a clasificar el árbol correcto entre los $n$ mejores. Por un lado, nuestros resultados validan los beneficios ya reportados en la literatura de tal preproceso de análisis profundo con respecto a la velocidad, cobertura y precisión. Por otro lado, proponemos una estrategia basada en existentes herramientas de código abierto y recursos para desarrollar con alta consitencia treebanks de sintaxis profunda para idiomas con limitada disponibilidad de recursos lingüísticos.
**Palabras clave:** Anotación sintáctica profunda de corpus, análisis HPSG, desambiguación morfosintáctica.

**Abstract:** This paper describes research on the effects of PoS tagging as a preprocess for HPSG-based deep parsing in the context of an open-source Spanish treebank development in the DELPH-IN framework. The treebank annotation is performed by hand selecting the proper decisions among the choices proposed by the system and ranked by a statistical module. The presented experiments show that the use of a tagger lowers the ambiguity of the sentences, both reducing the amount of sentences that reach time-out before the entire parse forest is built, and helping the ranker to place the right tree among the $n$-best trees. On the one hand, our results validate the benefits –already reported in the literature– of such preprocess to deep parsing with regard to speed, coverage, and accuracy. On the other hand, we propose a strategy based on existing open-source tools and resources to develop highly-consistent deep-annotated treebanks for languages with limited availability of linguistic resources.
**Keywords:** Deep syntax treebank annotation, HPSG parsing, PoS tagging.

## *1 Introduction*

Linguistically interpreted natural language texts constitute a crucial resource both for theoretical linguistic investigations about language use and for practical NLP purposes. Thus, in recent years, there has been an increasing interest in the construction of treebanks and, nowadays, both theory-neutral and theory-grounded treebanks have been developed for a great variety of languages.[1]

While first efforts in treebank building used manual annotation, recent significant advances in the development of large-scale robust efficient grammars and hybrid statistical/symbolic approaches for resolving ambiguities have made it possible to use sophisticated linguistic handcrafted deep-syntax frameworks (such as HPSG or LFG) to support the annotation task (Riezler et al., 2002; Prins and van Noord, 2003; Toutanova et al., 2005).

However, a drawback of these approaches is that their detailed granularity produces a huge ambiguity, creating efficiency problems to the parser machinery and making the effective use of the results difficult. Ambiguity not only slows down processing, but it also impoverishes the grammar performance in terms of coverage due to time-out problems when parsing long sen-

[1] Some of these treebanks are presented in (Hinrichs and Simov, 2004).

ISSN 1135-5948 

tences. Besides, it also leads to negative effects on parsing accuracy, caused by the combinatorial explosion of the search space. In the case of treebank development, this may represent a severe slow down –with consequent cost increase– and that some sentences can not be parsed and must be excluded from the resource. Thus, strategies must be devised to extend the coverage and the efficiency of deep parsers used in treebank development. Such strategies should rely on basic existing state-of-the-art resources (e.g. a PoS tagger) in order to be potentally appliable to the development of deep syntax annotated corpus for a wide range of languages.

Many research lines have been pursued to improve the performance of deep parsers, most of them relying on hybrid systems that combine shallow and deep NLP paradigms. The prime motivation for most of the published hybrid directions was to improve the efficiency of the parsers (Bangalore et al., 1997; Bangalore and Joshi, 1999; Ciravegna and Lavelli, 1997; Watanabe, 2000; Grover and Lascarides, 2001; Marimon, 2002; Crysmann et al., 2002; Prins and van Noord, 2003; Daum, Foth, and Menzel, 2003; Frank et al., 2003; Clark and Curran, 2004; Zhang, Matsuzaki, and Tsujii, 2009). Besides, some of the integrated shallow-deep processing also showed improvements in the robustness (Marimon, 2002; Crysmann et al., 2002; Daum, Foth, and Menzel, 2003; Adolphs et al., 2006) and the precision (Prins and van Noord, 2003; Daum, Foth, and Menzel, 2003; Sagae, Miyao, and Tsujii, 2007) of rule-based symbolic grammars.

As for the level of shallow information that the hybrid architectures integrate to achieve their goals, it ranges from simple morphological information and PoS information to different shallow syntactic analysis.

The works by Grover and Lascarides (2001) and by Prins and van Noord (2003) are two examples of the benefits of using basic PoS information. Grover and Lascarides (2001) interface PoS tag information with the existing lexicon of the Alvey Natural Language Tools system: if a word exists in the lexicon, the PoS tag is used as a filter, accessing only those entries of the appropriate category, if the word is unknown to the system, a basic underspecified entry for the PoS tag is used as its lexical entry. An experiment with 200 sentences shows how performance improves a 37.5%, with a precision of 30.5%. Prins and van Noord (2003) show how a HMM n-gram PoS tagger can be used to filter unlikely lexical categories to increase the speed of a parsing system based on a wide-coverage HPSG for Dutch. Experimental results with a test set of 216 sentences show that the use of the tagger greatly reduces parsing time, and, in addition, yields an increase of parsing precision.

Other proposals have extended the integrated information and exploit shallow syntactic analysis as produced by different shallow tools. In this line, Bangalore et al. (1997) present a system which applies a statistical disambiguation technique prior to parsing in the LTAG framework. A (trigram) disambiguation model is used to disambiguate so-called supertags, tags that represent the syntactic behavior of words and have a 1-to-1 mapping with the grammar lexical types. The task of the parser is thus reduced to establish the dependency links, with a parsing speed-up of about factor 30, with a tag accuracy of 68%. Later experiments reported in (Bangalore and Joshi, 1999) improve the tag accuracy to 92% by using much larger amount of training data and adding some smoothing techniques. The benefits of suppertagging in parsing speed has also been demonstrated in other lexicalised formalisms like CCG (Clark and Curran, 2004) and HPSG (Zhang, Matsuzaki, and Tsujii, 2009; Dridan, 2009). In another line of research, Ciravegna and Lavelli (1997) propose to use text chunking for controlling an agenda-based bottom-up chart parser; preliminary text chunking allows them to focus directly on the constituents that seem more likely, reducing the spurious ambiguity. The chunking process is done via finite state automaton, taking the output of a PoS tagger. They claim that experiments show a reduction of about 68% of constituents generated and of 78% of time consumed. Frank et al. (2003) combine macro-structural constraints derived from a probabilistic topological field parser for German with a constraint-based HPSG parser and report a performance gain of factor 2.25 on a set of 5060 sentences. Watanabe (2000) describes an algorithm for accelerating the CFG-Parsing process by using dependency information provided by stochastic parsers, interactive systems and linguistic annotations added in the source text. Reported reduction of processing time is about 45% and 15%. And, more recently, Sagae, Miyao, and Tsujii (2007) combine dependency and HPSG parsing and report a 1% absolute improvement in precision and recall of predicate-argument identification in HPSG parsing over a strong baseline.

The contribution of more than one shallow

component to the performance of deep analysis has also been investigated, though to a smaller extend. Marimon (2002) integrates a cascade of shallow components performing PoS tagging and chunk recognition as a pre-processing module of a HPSG-based grammar of Spanish implemented in the ALEP system. Experimental results show that the efficiency of the overall analysis improves an average of 65% and that the system also provides robustness to the linguistic processing, while maintaining both the recall and the precision of the grammar. The same approach is used by Crysmann et al. (2002) within the LKB system, where they use partial analyses from shallow processing to guide the deep parser to identify relevant candidates for deep processing. Also, Daum, Foth, and Menzel (2003) investigate the contributions of both taggers and chunkers to the performance of a deep syntactic parser with a Weighted Constraint Dependency Grammar of German and report to achieve a high degree of lexical robustness, reduced run time requirements, and a considerably improved parsing accuracy on a set of 1845 sentences.

This paper describes research on the effects of a state-of-the-art PoS tagger in deep parsing of unrestricted Spanish text, carried out in the context of on-going work for the creation of a new open-source resource for Spanish –an HPSG-based treebank called Tibidabo–. We focus on investigating to what extend using a tagger affects the system results both in terms of *coverage* (measured as the percentage of sentences for which it produces an output in the allocated time) and *accuracy* (measured as the percentage of sentences for which the right parse tree is ranked among the best ones). Additionaly, our research contributes to validate the benefits of a PoS tagger on parsing speed already reported in the literature.

Note that, being our goal to build a treebank, the preprocess must rely on existing state-of-the-art tools and we can not resort to more sophisticated techniques –e.g. supertagging– due to the lack of training material.

The following two sections summarize the set-up and motivation of our research. Section 3.1 describes experiments on the influence of tagging on deep parsing of unrestricted Spanish text, and section 5 presents some conclusions and some directions for future work.

## *2 The Annotation Environment*

As we have already mentioned, the research we describe in this paper is carried out in the context of on-going work for the construction of a new open-source language resource for Spanish: an HPSG-based treebank.[2]

Our investigation uses the DELPH-IN open-source tools for writing and processing HPSG grammars and the DELPH-IN publicly available Spanish Resource Grammar.[3]

The treebanking environment in the DELPH-IN framework is based on the selection[4] of the correct analysis among all the analyses that are produced by a symbolic grammar, instead of using only human annotation. It also provides a Maximum Entropy (ME) based stochastic learner (Toutanova et al., 2005) that observes decisions taken by the annotators and applies the same in unseen parses to reduce the outputs generated by the grammar and, therefore, the manual annotation effort in treebanking even with long sentences.

Nevertheless, some sentences still can not be included in the treebank due to: (a) the parser can not build the complete parse forest in the allocated time and exits with a time-out, or (b) the parser generates a large number of possible analysis and the right one is not ranked between the solutions offered to the annotator.

We will study whether the use of a PoS tagger reduces the timed-out sentences and whether it increases the number of sentences for which the right analysis is present among those ranked best by the statistical component.

### 2.1 Parser and Grammar

The Spanish Resource Grammar is a broad-coverage precise grammar for Spanish that aims at full parsing of unrestricted text.

The grammar is implemented on the *Linguistic Knowledge Builder* (LKB) system –an interactive grammar development environment for typed feature structure grammars– (Copestake, 2002).

The Spanish Resorce Grammar is grounded in the theoretical framework of HPSG (Pollard and Sag, 1987; Pollard and Sag, 1994), a constraint-based lexicalist approach to grammatical theory where all linguistic objects (i.e., words and phrases) are represented as typed feature structures, and they use the *Minimal Recursion Semantics* (MRS) semantic representation (Copes-

[2]The current treebank version is already publicly available within the DELPH-IN framework.

[3]See http://www.delph-in.net/.

[4]Selection is done by rejecting (or, alternatively, selecting) the lexical items and grammar rules that originate the multiple parses to incrementally disambiguate the sentence until a single analysis is left.

take et al., 2006). Using unification of typed feature structures, the MRS representation assigns a syntactically flat semantic representation to linguistic expressions which offers, by means of labeling of arguments and their co-indexation, a list of semantic relations and a set of syntactic limitations on possible scope relations among them.

The Spanish Resource Grammar has a full coverage lexicon of closed word classes (pronouns, determiners, prepositions and conjunctions) and it contains about 50,000 lexical entries for open word classes.[5] These lexical entries are defined by a set of about 500 lexical types that represent the type of words in the lexicon. Following well-established theoretical HPSG proposals, these lexical types are organized into a multiple inheritance type hierarchy (i.e., subtypes may inherit properties from more than one supertype higher in the hierarchy) allowing for lexical generalizations shared by several subtypes to be captured only once. The grammar also has 70 lexical rules to perform valence changing operations on lexical items (e.g. movement and removal of complements) which reduces the number of lexical entries to be manually encoded in the lexicon, and 230 phrase structure rules to combine words and phrases into larger constituents and to compositionally build up the semantic representation.

The Spanish Resource Grammar deals with a wide range of constructions in Spanish, including: main clauses with canonical word order surface and word order variations, valence alternations, determination, agreement, null-subject, compound tenses and periphrastic forms, raising and control, passives, (basic) comparatives and superlatives, all types of relative clauses, unbounded dependency constructions, cliticization phenomena, constructions with *se*, coordination, and nominal and verbal ellipsis.

## 2.2 PoS tagger

In our system, before parsing input sentences, raw text is pre-processed by FreeLing, an open-source language analysis tool suite performing shallow processing functionalities (Padró et al., 2010).[6]

FreeLing receives a sentence, morphologically annotates each word by dictionary look-up, and performs state-of-the-art HMM disambiguation, with an estimated accuracy around 97%. The morphological analysis step includes the application of a cascade of specialized processors that annotate punctuation symbols, multi-words, numerical expressions, date/time expressions, ratios, percentages, monetary amounts, and proper nouns.[7]

The integration of FreeLing is done using the LKB *Simple PreProcessor Protocol* (SPPP) which maps PoS tags into partial feature structures.[8] This SPPP interfacing module allows the definition of some adaptation rules aiming to ensure the smooth integration of both tools and to provide the best balance between parsing efficiency and accuracy. For instance, a list of words or tags causing ambiguities not solved with high reliability by the HMM tagger (like the ambiguity pronoun-conjunction of the word *que* (that), or proper names at sentence beginning) can be specified. For those words and tags, the PoS tagger decisions will be ignored (no analysis will be discarded) when found at the specified position, passing all possibilities to the deep parsing to be resolved by the symbolic grammar.

Also, this interfacing module can be configured with a list of substitutions of certain categories in FreeLing output by the category expected by the grammar. In this way, we avoid parsing failures due to discrepancies in the FreeLing tagset and the lexical categories assumed by the Spanish Resource Grammar (this is the case, for instance, of deictic adverbs like *here*, *there*, *today*, *tomorrow*, etc., which FreeLing tags as adverbs while the grammar lexicon encodes them as pronominal signs).

## 2.3 Target corpus

To create the treebank Tibidabo we chose newspaper text we borrowed from the corpus AnCora, a corpus of 528,000 words (17,363 sentences) (Taulé, Martí, and Recasens, 2008). Table 1 shows the number of sentences and ratio distributed along the sentence length.

Although the AnCora corpus already provides syntactic annotation, semantic roles, coreference, and other linguistic markup similar to what a deep analysis framework as HPSG and MRS can

[5] The grammar also includes a set of generic lexical entry templates for open classes to deal with unknown words for virtually unlimited lexical coverage.

[6] The FreeLing toolkit may be downloaded from: http://nlp.lsi.upc.edu/freeling.

[7] FreeLing also includes a guesser to deal with words which are not found in the lexicon by computing the probability of each possible PoS tag given the longest observed termination string for that word.

[8] SPPP assumes that a pre-processor runs as an external process to the LKB system and communicates with its caller through its standard input and output channels. See http://wiki.delph-in.net/moin/LkbSppp.

| *Sentence length* | *# sentences* | *% of the corpus* |
|---|---|---|
| 1-5 | 872 | 5.02 |
| 6-10 | 1,420 | 8.17 |
| 11-15 | 1,877 | 10.81 |
| 16-20 | 2,029 | 11.62 |
| 21-25 | 2,051 | 11.81 |
| 26-30 | 1,987 | 11.44 |
| 31-35 | 1,871 | 10.77 |
| 36-40 | 1,701 | 9.79 |
| 41-45 | 1,318 | 7.59 |
| 46-50 | 997 | 5.74 |
| 51+ | 1,246 | 7.17 |
| Total | 17,363 | 100 |

Table 1: Distribution of sentence lengths in the corpus.

offer, the annotations are hand created. Even if a thorough metodology and detailed criteria are used, human annotators are error-prone or may misinterpret the criteria. Any human-annotated resource unavoidably suffers from a certain degree of error or inconsistent criteria due to this fact.

We believe that providing a corpus consisting of the same text annnotated under a different paradigm –where the annotation criteria are enforced by a deep analysis lexical grammar instead of human annotators– may be a valuable resource for research. Such corpus can be useful in studying the variability of human annotation, the ability of machine learing algorithms to capture the structures annotated in each approach, the study of how different linguistic criteria can be mapped to each other, among many other possibilities.

## *3 Experimental Setting*

A rough idea of the coverage of the current version of the grammar may be drawn from the fact that about 30.4% of the sentences of up to 50 words receive at least a full parse.[9]

Parsing failures in the remaining 70% of sentences are basically due to two reasons. First, the processing components –as any other complex software in development stage– certainly show some deficiencies –lack of coverage, errors and unanticipated interactions, lack of robustness– that are responsible for 12.2% of the parsing failures. Second, 57.4% of the input sentences reach time-out limit set in the parsing engine (which was set at 60 seconds per sentence), because they get a too large number of analyses. The failure ratio due to time-out limit increases considerably with longer sentences (see table 3), which clearly shows up the need for improving the efficiency of the system to enable parsing of unrestricted Spanish text.

The 30% of sentences up to 50 words that receive at least a full parse get, in fact, an average of 5,040 parses/sentence. This amount of possible trees requires too many reject/select decisions by the human annotator, increasing the difficulty of the task and dramatically slowing down the treebank construction. To palliate this, the stochastic ranker in the DELPH-IN framework is trained and used to select a reduced number of parse trees to be presented to the annotator, thus reducing the number of decisions needed to disambiguate the sentence. Nevertheless, a huge amount of possible parses poses a more difficult challenge to the ranker, and the right tree may not always be among those selected.

Both the time-out problem and the large number of trees the ranker has to deal with reduce the number of sentences that can be annotated and included in the treebank. Thus, overcoming these issues is a crucial step to build a complete and useful resource.

Since lexical ambiguity is a cause shared by both problems, our approach is to use a PoS tagging preprocessor that reduces the ambiguity the parser has to deal with. In the following section we present two experiments that measure the influence of tagging on both the efficiency of the parser –which assigns (multiple) analyses to input sentences– and the accuracy of the ranking model –which chooses the best ones among them.

### 3.1 Corpus Ambiguity

Before reporting the results of our experiments, we present some statistics on the morphological, lexical, and syntactic ambiguities in the corpus.

We denote as *morphological ambiguity* the PoS ambiguity that is typically addressed by a tagger. Table 2 shows a summary of the morphological ambiguities (tags per word) in the corpus.

| | *Ambig. words* | | *All words* | |
|---|---|---|---|---|
| | *# words* | *tg/w* | *# words* | *tg/w* |
| open-class | 83,000 | 2.30 | 235,000 | 1.46 |
| closed-class | 144,000 | 2.62 | 293,000 | 1.80 |
| Total | 227,000 | 2.46 | 528,000 | 1.63 |

Table 2: Morphological ambiguity profiles of the corpus.

[9]Longer inputs can not be parsed within established time-out limits.

The Spanish grammar implemented in the DELPH-IN system is grounded in the theoretical framework of HPSG, a heavily lexicalist approach to grammatical theory where words are assigned many lexical classes that differ, for example, in the valence frame.[10] We denote as *lexical ambiguity* the average number of lexical classes per word that the parser takes into account. In the case of our corpus, it is 7.0 lexical classes per word.

Given an input sentence, the parser considers, for each word, all lexical classes matching the valid PoS tags for that word. Then, all possible parses consistent with those possibilities are built, producing a large amount of full syntactic analyses. We denote as *syntactic ambiguity* the average amount of possible full parses per sentence generated by the parser.

The average syntactic ambiguity for the 30% of sentences up to 50 words that get some analysis, is 5,040 parses/sentence.

## *4 Experiments*

### 4.1 Experiment 1: Influence on Coverage

To investigate the effects of the PoS tagger on the efficiency of the Spanish grammar we parsed the whole corpus with and without the tagger and compared the system performance.

Table 3 shows the ratio of sentences that received at least a full parse, as well as the percentage of sentences for which the parser timed out, distributed along the sentence length.[11]

Not surprisingly, due to tagging errors, PoS tagging caused a small loss in the number of short sentences receiving an analysis: A 3% less of sentences under 10 words were analyzed, but since there are relatively few sentences in that range, this represents a loss of only 0.4% over the whole corpus. However, the tagger certainly had a positive impact on longer sentences –with lengths between 11 and 40 words– where the observed coverage increase was 7.2%. Note that sentences in this length range constitute two thirds of the whole corpus. Thus, the overall ratio of sentences in the whole corpus that received an analysis increased in 6.9%.

The PoS tagger reduced the morphological ambiguity from 1.63 to 1.03 tags/word,[12] which reduced from 7.0 to 4.7 lexical classes per word

| *Sent. length* | % of corpus | *Parsed sentences* | | *Timeout ratio* | |
|---|---|---|---|---|---|
| | | *no tag* | *tag* | *no tag* | *tag* |
| 1-5 | 5.0 | 91.4 | 87.4 | 0 | 0 |
| 6-10 | 8.2 | 89.2 | 86.6 | 2.6 | 0.8 |
| 11-15 | 10.8 | 73.8 | 74.3 | 12.6 | 4.6 |
| 16-20 | 11.6 | 49.9 | 61.2 | 34.3 | 10.6 |
| 21-25 | 11.8 | 25.2 | 38.0 | 58.7 | 34.8 |
| 26-30 | 11.5 | 10.3 | 21.1 | 74.4 | 50.3 |
| 31-35 | 10.8 | 3.5 | 9.2 | 82.0 | 61.6 |
| 36-40 | 9.8 | 1.2 | 3.4 | 86.6 | 71.5 |
| 41-45 | 7.6 | 0.5 | 1.0 | 88.8 | 75.2 |
| 46-50 | 5.7 | 0.2 | 0.2 | 89.6 | 77.3 |
| 51+ | 7.2 | 0 | 0 | 100.0 | 100.0 |
| Total | 100.0 | 30.4 | 37.3 | 57.4 | 42.6 |

Table 3: Percentages of parsed and timed-out sentences.

the lexical ambiguity the parser has to deal with. This caused the parser to build less constituents not contributing to the final parse, making it possible to parse 7% more sentences, for which the parser timed-out before. The syntactic ambiguity when using PoS tagging slightly increased (from 5,040 to 5,434 analysis/sentence) due to the fact that longer sentences –which are more ambiguous– that were not parsed before are now included in the count.

As we expected, morphological disambiguation also had a positive impact on parsing time and reduced average processing time from 38.4 to 30.4 sec/sentence (even when longer sentences are now included in the count).

### 4.2 Experiment 2: Influence on Accuracy

To evaluate the impact of tagging on the accuracy of the ME ranking model, we calculated the ratio of sentences for which the parse in the gold standard is ranked among the $n$ best by the stochastic model. Note that an output analysis includes both a phrase structure tree and a MRS semantic representation, and that exact match is required.

The corpus used in this experiment was a small part of the whole treebank, consisting of 2,570 sentences of lengths up to 16 words. The experiment was performed using 5-fold cross-validation.

Table 4 shows the accuracy of the parse selection model (percentage of sentences for which the right full parse tree was ranked by the model

[10] For example, the average numbers of entries per verb is 1.84, however, some verbs have as many as 8 lexical entries.

[11] The corpus was parsed with a Quad-Core 2.83GHz with 8Gb RAM.

[12] In a few cases where the tagger has large error rates, such as the conjunction/relative ambiguity for the word *que*, the tagger output is ignored and the ambiguity maintained.

among the $n$ best) when the used treebank was parsed either without or with the tagger. Differences are significant at a 95% confidence degree according to a paired t-test for all values of $n$ except $n = 10$.

| $n$-best | no tagger | tagger |
|---|---|---|
| 1 | 54.8% | 56.3% |
| 2 | 65.1% | 66.9% |
| 3 | 70.8% | 73.0% |
| 4 | 74.1% | 77.0% |
| 5 | 78.1% | 79.5% |
| 10 | 85.7% | 85.6% |
| 20 | 91.4% | 89.6% |
| 30 | 94.7% | 91.6% |

Table 4: Accuracy of the parse model selection model for $n$-best analyses with and without the tagger.

The reason why the use of a PoS tagging improves parsing accuracy is that it reduces the number of candidate analyses, largely reducing the search space that the selection model has to deal with. For this set of sentences, the grammar assigned an average of 7.7 lexical classes per word and produced an average of 1,235 parse trees per sentence. If PoS tagging is used, these figures are reduced to 4.3 classes per word and 606 analyses per sentence.

## 5 Conclusions and Future Work

This paper describes research that shows the usefulness of PoS tagging for improving speed, coverage, and accuracy of HPSG-based deep parsers used in treebank development. A first experiment shows that, by improving parsing speed, PoS tagging increases parsing coverage for long sentences. The second experiment shows a statistically significant improvement in parsing accuracy when using a Maximum-Entropy based model to rank the $n$-best analysis for each sentence.

Presented results show a 14.8% decrease of timed-out sentences (from 57.4% to 42.6%) consisting of a 6.9% of coverage increase, plus a 7.9% of sentences that no longer time out but are not yet analyzed due to tagger errors or to the lack of the appropriate rules in the grammar.

They also show that the use of a PoS tagger yields a significant increase in the percentage of sentences for which the right tree is ranked among the best ones by the statistical module.

The presented results are most informative in order to design an optimal annotation strategy aiming to maximize the annotation speed while maintaining high levels of accuracy: about 50% of the sentences in the corpus can be annotated using the tagger and setting the ranker to select a small number of trees. Given this reduced forest, the annotation process is very fast, since each sentence requires only a few annotator decisions to be disambiguated. The 20% of sentences that have not been annotated (e.g. because the right tree was not among those proposed) can be processed again with a higher number of candidate trees. The remaining sentences can be annotated at a slower rate without the tagger. Finally, a higher time-out can be set to have the parser analyze sentences where it timed out before, and repeat the process. This strategy is based on existing open-source tools and resources[13], and makes it possible to develop highly-consistent deep-annotated treebanks for languages with limited availability of linguistic resources.

Future work will include the extension of the grammar coverage, the extension of the treebank. We will also study the viability of training a high-precision ranker that allows the automatic annotation of a large number of sentences in the treebank.

## Acknowledgments

This work has been partially funded by the European Union through project X-LIKE (FP7-ICT-2011-288342), by the Spanish Government through the programe *Ramón y Cajal* and the project KNOW2 (TIN2009-14715-C04-03/04), and by the Catalan Government via the mobility programe *Beques per a estades per a la recerca fora de Catalunya*.

## References

Adolphs, P., S. Oepen, U. Callmeier, B. Crysmann, D. Flickinger, and B. Kiefer. 2006. Some fine points of hybrid natural language parsing. In *Proceedings of the 5th Internatinal Conference LREC*, Genoa, Italy.

Bangalore, S., C. Doran, B.A. Hockey, and A. Joshi. 1997. An approach to robust partial parsing and evaluation metrics. In *Proceedings of the 5th International Workshop on Parsing Technologies*, Boston, MA.

Bangalore, S. and A. Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 2(25):237–265.

[13] All the used DELPH-IN resources –the software, the SRG grammar, and the treebank– as well as the FreeLing toolkit are licensed under GPL or LGPL.

Ciravegna, F. and A. Lavelli. 1997. Controlling bottom–up chart parsers though text chunking. In *Proceedings of the 5th International Workshop on Parsing Technologies*, Boston, MA.

Clark, S. and J.R. Curran. 2004. The importance of supertagging for wide-coverage CCG parsing. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*, Geneva, Switzerland.

Copestake, A. 2002. *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford.

Copestake, A., D. Flickinger, C.J. Pollard, and I.A. Sag. 2006. Minimal recursion semantics: an introduction. *Research on Language and Computation*, 3(4):281–332.

Crysmann, B., A. Frank, B. Kiefer, S. Müller, G. Neumann, J. Piskorski, U. Schäfer, M. Siegel, H. Uszkoreit, F. Xu, M. Becker, and H.U. Krieger. 2002. An integrated architecture for shallow and deep processing. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Pittsburgh.

Daum, M., K.A. Foth, and W. Menzel. 2003. Constraint based integration of deep and shallow parsing techniques. In *Proceeding of the 10th Conference of the EACL*, Budapest.

Dridan, R. 2009. Using lexical statistics to improve HPSG parsing. Master's thesis, Saarland University, Sarbrücken, Germany.

Frank, A., M. Becker, B. Crysmann, B. Kiefer, and U. Schäfer. 2003. Integrated shallow and deep parsing: Topp meets HPSG. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*.

Grover, C. and A. Lascarides. 2001. XML-based data preparation for robust deep parsing. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, Toulouse, France.

Hinrichs, E.W. and K. Simov, editors. 2004. *Research on Language and Computation*, volume 2(4). Kluwer Academic Publishers.

Marimon, M. 2002. Integrating shallow linguistic processing into a unification-based spanish grammar. In *Proceedings of the 19th International Conference on Computational Linguistics*, Taipei, Taiwan.

Padró, L., M. Collado, S. Reese, M. Lloberes, and I. Castelón. 2010. Freeling 2.1: Five years of open-source language processing tools. In *Proceedings of 7th Language Resources and Evaluation Conference (LREC'10)*, La Valletta, Malta.

Pollard, C.J. and I.A. Sag. 1987. *Information-Based Syntax and Semantics, Volume 1: Fundamentals*. CSLI Lecture Notes, Stanford University, CA.

Pollard, C.J. and I.A. Sag. 1994. *Head-driven Phrase Structure Grammar*. The University of Chicago Press and CSLI Publications, Chicago.

Prins, R. and G. van Noord. 2003. Reinforcing parser preferences through tagging. *Special issue on Evolutions in Parsing of the journal Traitement Automatique des Langues 44(3)*, pages 121–139.

Riezler, S., T.H. King, R.M. Kaplan, R. Crouch, J.T. Maxwell, and M.Johnson. 2002. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, Philadephia, PA.

Sagae, K., Y. Miyao, and J. Tsujii. 2007. HPSG parsing with shallow dependency constraints. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic.

Taulé, M., M.A. Martí, and M. Recasens. 2008. AnCora: Multilevel annotated corpora for catalan and spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC-2008*, Marrakech, Morocco.

Toutanova, K., C.D. Manning, D. Flickinger, and S. Oepen. 2005. Stochastic HPSG parse disambiguation using the redwoods corpus. *Journal of Logic and Computation*.

Watanabe, H. 2000. A method for accelerating CFG-parsing by using dependency information. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrücken, Luxembourg, Nancy.

Zhang, Y.-Z., T. Matsuzaki, and J. Tsujii. 2009. HPSG supertagging: A sequence labeling view. In *Proceedings of the 11th International Conference on Parsing Technology (IWPT'09)*, Paris, France.