Syntactic Analysis and Semantic Role Labeling for Spanish using Neural Networks

Luis Chiruzzo and Dina Wonsever

Facultad de Ingeniería,
Universidad de la República,
Montevideo, Uruguay
{luischir,wonsever}@fing.edu.uy

Abstract. This paper describes a neural network architecture for parsing Spanish sentences using a feature based grammar. The architecture consists of several LSTM neural network models that produce syntactic analysis and semantic role labeling: first determine how to split a sentence into a binary tree, then assign the rule to be applied for each pair of branches, finally determine the argument structure of the resulting segments. We analyze two variants of this architecture and conclude that merging the split and rule identification models yields better results than training them separately. We train and evaluate the performance of these models against two Spanish corpora: the AnCora corpus and the IULA treebank. Our best models achieve encouraging results for the syntactic parsing process: unlabeled score 87.5%, labeled score 81.9% for AnCora; unlabeled score 93.6%, labeled score 88.9% for IULA; furthermore it achieves 96.2% UAS and 91.3% LAS when transforming back to IULA dependencies, which is comparable to the state of the art. On the other hand, although the intrinsic results for the argument structure identification are promising (90.9% accuracy and 87.1% macro-F1), when combined with the rest of the steps the performance significantly drops, so there is plenty of room for improvement in the process.

Keywords: Parsing · Syntactic Analysis · Semantic Role Labeling · Spanish · LSTM.

1 Introduction

Syntactic parsing is one of the fundamental processes in the Natural Language Processing pipeline, and it has been studied in the field since its inception. The two main paradigms for syntactic analysis are: constituency parsing, where the sentence is structured as a tree of linguistically motivated segments; and dependency parsing, where no explicit constituents are detected but each word is associated to another word that acts as its head. Other approaches, which could be called deep parsing, try to include more information in the parse trees in order to bring it closer to a semantic representation. One way of doing this is enriching the structures of a constituency tree with features that indicate the relation of each constituent or word to other structures in the tree.

In this work we present a process for parsing a sentence in Spanish using a feature based grammar inspired on a simplified version of HPSG [26]. The parsing process has three steps: first it splits the sentence into segments until it builds a binarized tree; then it identifies the grammar rule that should be applied for each pair of branches (siblings) in the tree; and finally it determines which of the segments should fulfill a semantic role in relation to the detected predicates, and which role it is. When the process is finished, the final tree contains both syntactic and semantic role information. In our parsing architecture, the three steps are fulfilled by neural networks.

The rest of the paper is structured as follows. Section 2 shows some background and other approaches to this task. Section 3 presents the corpora we base our work on. Section 4 describes our parsing process in detail and how we train the neural networks. Section 5 shows the results of our experiments using both corpora. Finally, section 6 presents some conclusions and other lines of research that we would like to follow in the future.

2 Related work

Over the last years, neural network models have been applied extensively to many tasks in NLP, and the parsing process is no exception. When focusing on constituency parsing, different approaches have been proposed for applying neural network techniques to this task. For example, [29] uses a recurrent neural network that combines pairs of words and builds a tree bottom-up in order to leverage the syntactic-semantic compositionality to improve sentiment analysis in English sentences.

A rather different approach is followed by [32], which frames the parsing process as a sequence-to-sequence problem: trying to learn a translation model between a sentence in natural language and the bracketed representation of its parse tree. The neural model they use is an encoder-decoder LSTM model with attention mechanism. They found out that training such a model using only standard corpus data (the Penn Treebank [21]) did not yield good results, but if they first pre-trained the model using a larger corpus of sub-standard data (the result of another parser, or a combination of parsers) the performance of the final model improved significantly.

One approach that has been studied lately is to use neural models to represent transition based syntactic parsing. These models consider the parsing process of a sentence as a sequence of actions that are performed to each word of the sentence, transversed from left to right. For example, in [14] they describe a transition based constituency parser that uses neural networks for representing the state of the stack [13], the sentence buffer and the current derivation history. Using all that information as input, the objective of the network is to predict the next action of the parser (shift, reduce, or adding a new non-terminal to the stack). They report good results for English parsing over the WSJ section of the Penn Treebank.

In [19] they combine the transition based approach with sequence-to-sequence modeling. They train a model that tries to translate an input sentence into a sequence of actions that a shift-reduce parser should take. They apply the same idea for a dependency parser [25] and a constituency parser [14], both for English, obtaining state of the art results. The authors of [33] propose a transition based approach for constituency parsing where they model the stack using a RNN or Elman Network, obtaining good results for English and Chinese treebanks, while in [12] the authors describe a transition based constituency parser that uses LSTMs for representing word spans instead of partially derived trees, obtaining good results for parsing English and French.

Another approach that focuses on determining the word spans in the tree is used in [30], which describes a top-down parser that greedily splits a sentence in constituents and assigns labels to them, processing the text spans with LSTMs in order to generate an intermediate representation. This approach is the most similar we found to the one we use, the main differences are that: they train parsers for English and French, while we work for Spanish; they use standard constituency grammars that allow multiple children per rule, while the feature based grammar we use is strictly binarized; and also we add a further step that determines the argument structure of the predicates.

In [28] they describe a parser based on LSTMs trained to predict the syntactic distances (concept related to the distance between words in the expected parse tree) between consecutive words in the sentence. Having the predicted syntactic distances, they proceed to build the tree in a top-down process splitting constituents guided by the relative syntactic distances between words. This process is faster as it only needs to calculate once the syntactic distances for the sentence. They achieve good results for English and Chinese parsing. Also for the English language, [15] proposes a model that calculates the score for each span inside a sentence and a model that predicts a label for those possible spans. Then they adapt a CKY-style algorithm to find the tree with optimal score based on their span model.

Compared to English, there are considerably fewer works that focus on Spanish parsing. For constituency parsing, [11] describes two approaches for improving PCFG parsing in Spanish: one approach involves including morphological information in the probabilistic model that predicts the rules, the other is a reranking method that uses a max-margin criterion trained over a set of global features from the n-best parse trees. The methods are evaluated against the Cast3LB corpus, a subset of the AnCora [31] corpus, achieving a constituent F1 of 83.6 (first approach) and 85.1 (second approach) over the test partition. In [18] they perform experiments in Spanish parsing using a PCFG with latent annotations. Their best model uses a simplified tagset and achieves 85.47 F1 over the test partition of the Cast3LB corpus.

Further work has been done for dependency parsing in Spanish, beginning with CoNLL-X shared task [5] on dependency parsing for multiple languages, including Spanish. The best LAS achieved for Spanish were 82.3 (max span tree approach) and 81.3 (transition based approach). Later on, [20] describes the

construction of a dependency grammar and a rule-based dependency parser for Spanish based on transforming the result of a shallow parser. They achieve 81.13 UAS and 73.88 LAS in AnCora, and 80.93 UAS and 74.33 LAS in SenSem [6]. In [3] they describe a set of experiments using MaltParser [25] to determine how much corpus size, sentence length or other factors contribute to the dependency parsing performance in Spanish.

3 Corpora

We use a feature based grammar inspired on a simplified version of HPSG [26]. The feature structure for representing words contains features for head, syntactic valence and PropBank-style [4] argument structure, as shown in figure 1. The grammar contains rules for: applying specifiers, complements, modifiers or punctuation symbols to the left or to the right of a head, and for applying coordinations. It also supports the use of clitics (an important feature in Spanish grammar) and argument sharing in relative clauses.

$$\begin{bmatrix} \text{word} & & \\ \text{SYN} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{n} \mid \text{v} \mid \dots & \\ \text{AGR} & \begin{bmatrix} \text{PER } 1 \mid 2 \mid 3 \\ \text{NUM sg} \mid \text{pl} \\ \text{GEN f} \mid \text{m} \\ \dots \end{bmatrix} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SPEC} & \langle \text{expr} \rangle \\ \text{COMPS} & \langle \text{expr} \rangle \\ \text{CLITICS} & \langle \text{expr} \rangle \\ \text{MOD} & \langle \text{expr} \rangle \end{bmatrix} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{ARGS} & \begin{bmatrix} \text{ARG0} & \text{expr} \\ \text{ARG1} & \text{expr} \\ \dots & \end{bmatrix} \end{bmatrix} \\ \text{TEXT} & \text{text} \end{bmatrix}$$

Fig. 1. Feature structure for a lexical entry containing syntactic valence and semantic arguments structure.

3.1 AnCora Corpus

AnCora [31] is a corpus of Spanish and Catalan text consisting of about half a million words in approximately 17,000 sentences, most of the text comes from news articles. All sentences in this corpus are annotated with morphological, syntactic and argument structure information. A version of this corpus using

HPSG-style annotations is described in [7–9]. We use this version for our experiments as it contains all the features we want the parser to learn.

The length of AnCora sentences varies widely, being the longest around 150 words. However, sentences this long are rather rare, if we consider sentences up to 60 words long the distribution of sentence lengths becomes more uniform. In our process we used all sentences of the corpus up to 60 words long, which comprise around 97% of the total corpus.

The Tibidabo Treebank[24] is another annotated corpus in Spanish that was built using sentences originally from AnCora. The approach to build this corpus was different, as the annotations in AnCora were not used, but a subset of AnCora sentences were parsed using a Spanish HPSG grammar [22] and their correct analyses were manually selected. However, the version of this corpus that is available for download is a conversion to dependency format, so a further transformation is needed to get all the features we need. As this corpus is smaller and has shorter sentences than the original AnCora, we decided not to use it.

3.2 IULA Treebank

The IULA Treebank [23] is a corpus of Spanish sentences annotated in a dependency grammar style, containing about 590,000 words in 42,000 sentences. Like Tibidabo, the corpus was originally parsed using a HPSG grammar and manually selecting the best parse tree, and the version of this corpus available for download is a transformation into dependency format as well. Because of this, we transformed this corpus back into a format akin to the feature based grammar we use.

Compared to AnCora, the IULA treebank generally contains shorter sentences: the longest sentence in IULA has 33 words, while AnCora has sentences longer than a hundred words. It contains a lot of technical language, as it was built using sentences from documents about law, economy, genomics, medicine, and environment. One problem with this corpus is that, unlike AnCora, it does not provide readily available argument structure information. So in this case we did not train neither evaluate the model for argument structure features.

4 Implementation of the parser

4.1 Parsing process

Our parsing process consists in a series of steps that incrementally build the parse tree: splitting a sentence into a binary tree, finding out the rules that apply to the nodes of the tree, and finally determining what nodes should be labeled with argument structure categories.

Let us walk through the proposed parsing process using the following sample sentence:

[*CiU ha pactado dinero para el delta del Ebro*]
(CiU has agreed to money for the Ebro delta)

Step 1: Splitter In the first stage, the process will take the whole sentence and split it in two sequences of words. The resulting subsequences are expected to be constituents of the sentence. In this case it should split the subject and the predicate. The result will look like the following:

[*CiU*] [*ha pactado dinero para el delta del Ebro*]

This process is repeated for each subsequence with two or more words. In this case, [*CiU*] has only one word, so only the second subsequence will be split. The process should separate the last constituent that affects the verb, resulting in the following:

[*ha pactado dinero*] [*para el delta del Ebro*]

Now both subsequences have more than one word, so they should be split further in smaller subsequences. For example the left one would be:

[*ha pactado*] [*dinero*]

This top-down process is iterated until there are no more subsequences left to split, effectively transforming the original sentence into a binary tree of words. The binary tree after step 1 for this example is shown in figure 2.

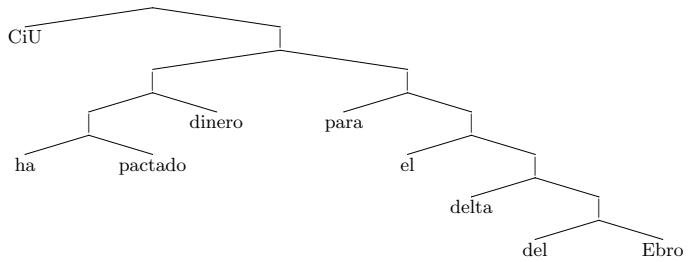

Fig. 2. Tree for *"CiU ha pactado dinero para el delta del Ebro"* ("CiU has agreed to money for the Ebro delta") after step 1.

Step 2: Rules After the tree is created, the second stage transverses all pairs of branches and tries to find the most suitable rule that describes the relation between those branches. For example:

[*ha pactado*] [*dinero*] → `head_comp`

[*CiU*] [*ha pactado dinero para el delta del Ebro*] → `spec_head`

After this stage is completed, the tree looks as shown in figure 3.

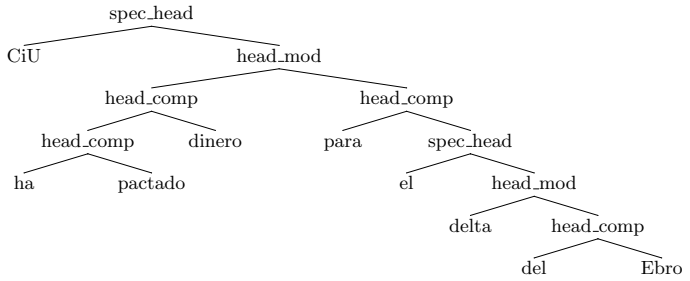

Fig. 3. Tree for *"CiU ha pactado dinero para el delta del Ebro"* ("CiU has agreed to money for the Ebro delta") after step 2.

Step 3: Arguments Once the tree is created and the syntactic features are in place, the third step tries to determine which of the syntactic arguments of the verbs, nouns and adjectives should also be set as semantic arguments.

Given the rules determined after step 2 for each pair of branches in the tree, we can infer the head for each constituent. This third step considers each head and each target argument that belongs to that head. In our example:

$$[\ \underline{CiU}_{SPEC}\ ha\ \underline{pactado}_{HEAD}\ dinero\ para\ el\ delta\ del\ Ebro"\ ] \rightarrow \texttt{arg0}$$

$$[\ CiU\ ha\ \underline{pactado}_{HEAD}\ \underline{dinero}_{COMP}\ para\ el\ delta\ del\ Ebro"\ ] \rightarrow \texttt{arg1}$$

A simplified version of the final tree for our example, after the three stages have been completed, is shown in figure 4. This final version of the tree contains the information about the arguments of each predicate and the semantic roles they have according to the argument structure.

4.2 Neural networks architecture

In our experiments, each of the steps is fulfilled by a neural network. The central layers of the models in all cases are bidirectional LSTM layers [17], which are very useful for processing entire sequences of words and creating a model for the whole sequence that can be used by further layers.

For our parsing process, we created the following models:

- **Step 1: Splitter model** A three-layered stacked LSTM network that returns the probability of each word in the sentence to be the limit of the split.
  Input: Sequence of words.
  Output: Probability for each word.
- **Step 2: Rule model** Two parallel LSTM layers that process two sequences of words and return the probability for each possible rule.
  Input: Two sequences of words.
  Output: Label indicating the rule to use (`head_spec`, `spec_head`, `head_comp`, `comp_head`, `head_comp_sem`, `head_mod`, `mod_head`, `clitic_head`, `head_rel`, `head_punct`, `punct_head`, `coord_left`, `coord_right`).

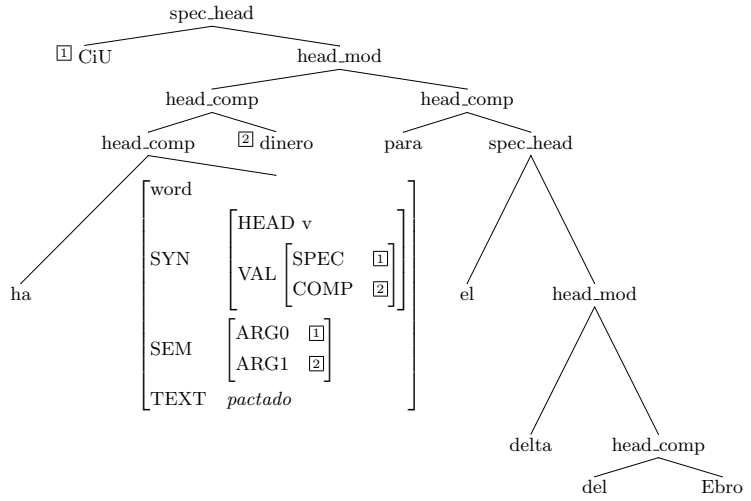

Fig. 4. Simplified tree for *"CiU ha pactado dinero para el delta del Ebro"* ("CiU has agreed to money for the Ebro delta") after step 3.

- **Step 3: Argument model** A network that takes a constituent and the information about the head of the constituent and the target argument to categorize (also belonging to the constituent).
  Input: One word (the head, which is the predicate), two sequences of words (the whole constituent, and the target argument), and the syntactic valence of the target argument.
  Output: Label indicating the arg type (`none`, `arg0`, `arg1`, `arg2`, `arg3`, `arg4`, `argm`, `argc`).

In later experiments, we merged the networks 1 and 2 into a single network which is in charge of splitting a sequence of words and determining the most likely rule to apply to the two resulting segments:

- **Splitter-Rule model** A three-layered stacked LSTM network that transverses the sentence and returns, for each word, the probability of it to be the limit of the split and the most likely rule to be used in that split.
  Input: Sequence of words.
  Output: Split probability and rule to use for each word.

This combined model has the advantage of being much faster than using the other two models, and in theory it could also leverage the information used for both tasks (splitting and determining rule) in order to improve the performance. We will present the results obtained on both architectures.

We also tried building a network that would combine the three steps. However, the information used by the argument detection step is different than the one used for the other networks (it considers more context). The initial experiments we performed trying to merge the three networks achieved good results

for the first two steps, but underperformed significantly for the arguments step, so we did not consider this approach in these experiments.

4.3 Training details

The neural network models were implemented in the `keras` library [10] over `tensorflow` [1]. In all models the words are represented by word embeddings trained from a six billion words Spanish corpus [2] using the `gensim` library [27]. The word embeddings set has 1,146,242 vectors of dimension 300. In order to handle out of vocabulary words, we create an unknown token for each POS tag (considering morphological information) and use the vector corresponding to the most frequent word for that POS.

The AnCora HPSG corpus was split in training-development-test partitions with sizes 80%-10%-10%. The IULA treebank has standard train-test partitions, but we further split the train sentences in training-development sets with sizes 80%-20%.

When training the neural networks, we used the early stopping technique, so we took a validation set of 10% to 20% of each training set, depending on the experiment. We trained several configurations (varying the number of layers, units and activation functions) for each of the models, optimizing against the corresponding development sets. We will only report the results on the best performing models for each set of experiments.

5 Experiments

We evaluated the trained models against the test partitions of AnCora HPSG and IULA. In the case of IULA, we also evaluated the conversion back to dependency trees against the original corpus in order to compare our results against the state of the art.

5.1 Intrinsic evaluation by step

Table 1. Performance for independent Splitter and Rule models, and combined Splitter-Rule model. Average accuracy for split and rule prediction is shown. For the combined model, also accuracy for predicting both values at the same time is shown.

	Independent models		Combined model		
Corpus	Split Acc.	Rule Acc.	Split Acc.	Rule Acc.	Both Acc.
AnCora Dev	85.30	95.68	94.18	95.85	92.44
AnCora Test	85.19	95.66	94.17	95.82	92.37
IULA Dev	93.27	96.73	96.61	96.76	94.85
IULA Test	92.74	96.78	96.56	96.88	94.85

Splitter, Rule and combined Splitter-Rule models Table 1 shows a summary of the performance results for the independent Splitter and Rule models, and the combined Splitter-Rule model. The first two columns show the performance for step 1 and step 2 achieved by the independent models. The best independent split models achieve 85.19% and 92.74% on average over AnCora and IULA. However, as we will see in the next section, due to the nature of the sentences and the sizes of the constituents, the performance of the whole step 1 process is higher than these intrinsic values.

For the independent rule models, the average accuracy achieved for the AnCora and IULA test sets was 95.66% and 96.78% respectively. Table 2 shows the summary of the performance for each class for these models. Notice that the hardest class to predict in all cases was `comp_head` (a complement applied to the left of a head), which is also a class with very few examples in the corpora.

The last three columns in table 1 also show that the combined splitter-rule model outperforms the independent models, particularly for the split step, as the average accuracy for this step increases from 85.19% to 94.17% for AnCora, and from 92.74% to 95.56% for IULA. The performance for the rule identification step also increases, but the difference is not significant. One reason this could be happening is that the combined model could be able to leverage the information of both tasks in order to improve them.

Table 2. Rules model performance.

	Ancora Dev			Ancora Test			IULA Dev			IULA Test		
Rule	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
`head_spec`	86.93	81.40	84.08	85.53	80.45	82.91	90.47	93.72	96.02	90.26	94.07	92.12
`spec_head`	98.39	99.19	98.79	98.22	99.01	98.61	98.43	99.03	98.73	98.46	99.06	98.76
`head_comp`	89.36	89.27	89.31	89.34	89.82	89.58	96.02	96.49	96.25	96.00	96.50	96.25
`head_comp_sem`	98.71	98.85	98.78	98.60	95.58	98.59	-	-	-	-	-	-
`comp_head`	84.54	78.25	81.27	83.14	79.30	81.17	80.85	49.14	61.13	85.88	48.67	62.13
`head_mod`	93.12	93.60	93.36	93.62	93.79	93.71	94.99	93.53	94.25	95.03	93.61	94.32
`mod_head`	94.91	93.44	94.17	95.34	93.19	94.25	93.97	93.56	93.76	94.63	93.75	94.19
`head_rel`	95.87	96.15	96.01	95.79	96.09	95.94	88.36	95.64	91.85	90.24	95.76	92.91
`clitic_head`	99.87	99.74	99.81	99.71	99.71	99.71	99.87	99.93	99.90	99.68	99.68	99.68
`head_punct`	99.54	100.	99.77	99.73	100.	99.86	100.	99.99	99.99	99.96	100.	99.98
`punct_head`	95.51	96.86	96.18	96.16	96.53	96.34	99.22	98.83	99.02	100.	99.66	99.83
`coord_left`	98.21	96.55	97.37	96.86	96.07	96.46	99.65	99.65	99.65	99.48	99.78	99.63
`coord_right`	92.22	90.81	91.51	91.49	91.05	91.27	97.60	98.90	98.25	97.50	99.26	98.37
Macro Avg	94.40	93.39	93.88	94.12	93.12	93.72	94.95	93.20	94.07	95.59	93.32	94.01

Arguments model For step 3, the best performing model on the AnCora development set has 90.71% accuracy and 85.58% macro-F1, and it is similar over the test set with 90.98% accuracy and 87.13% macro-F1. These results look very promising, for example comparing this to CoNLL-2009 shared task

on SRL [16], where the best results for Spanish were 83.31% on the joint task and 80.46% on the SRL-only task. Our intrinsic step evaluation should be more similar to the SRL-only task. However, even if the corpus used for the challenge and the corpus we used are both derived from AnCora, we have to take in consideration that there are differences in the transformation processes of the corpora that might make this comparison not direct. Further analysis would be needed to understand how similar the semantic role information in both corpora is. Furthermore, as we will see in the following section, the final performance of the SRL process drops significantly when considering all the steps together.

5.2 Full process

Table 3 shows the results for the full process over the development and test sets of AnCora. As expected for this kind of process, the performance varies significantly with the sentence size (shorter sentences are easier to analyze), so we report values for sentences up to 20, 40 or 60 words long. The first two columns show F1 score for unlabeled span identification (the results of the split step) and labeled span identification (split step plus rule step). The final two columns show the F1 score for syntactic and semantic features. The syntactic features are generated with a deterministic process from the spans and rules, while the semantic features are the result of using the argument identification step of the process.

Table 3. Performance for the full process over AnCora development and test sets.

Corpus	Length	Unlabeled F1	Labeled F1	Syntactic F1	Semantic F1
Independent Models					
AnCora Dev	≤20	89.30	82.62	82.01	73.36
	≤40	83.50	76.90	78.48	67.32
	≤60	81.24	74.62	77.19	63.41
AnCora Test	≤20	88.74	82.82	82.64	74.47
	≤40	83.12	76.72	78.61	66.06
	≤60	81.46	75.03	77.69	63.06
Combined Models					
AnCora Dev	≤20	93.03	86.54	86.62	78.69
	≤40	89.17	83.31	84.69	74.97
	≤60	87.44	81.71	83.65	71.22
AnCora Test	≤20	92.54	86.92	86.40	79.09
	≤40	88.81	83.24	84.52	74.17
	≤60	87.47	81.87	83.70	70.92

Table 4 shows the results for the IULA corpus. In this case, only the unlabeled and labeled span identification are reported, as we did not have semantic argument information for this corpus.

In order to compare our results with other Spanish parsers that have worked with this corpus, we needed to transform it back to its original dependency

Table 4. Performance for the full process over IULA development and test sets.

Corpus	Length	Unlabeled F1	Labeled F1
Independent Models			
IULA Dev	≤20	92.54	87.36
	≤33	91.73	86.50
IULA Test	≤20	92.35	87.04
	≤33	91.47	86.15
Combined Models			
IULA Dev	≤20	94.30	89.74
	≤33	93.74	89.20
IULA Test	≤20	94.10	89.46
	≤33	93.56	88.96

format. We used a deterministic heuristic process for that, mapping specifiers, complements, modifiers and other features into their corresponding dependency label (`SPEC`, `SUBJ`, `COMP`, `DO`, `MOD`, etc.) and using information from the context of the tree to decide which was the most suitable label. We know this mapping is far from perfect because we could not find a match for some of the original labels of IULA, so our conversion process will have a lower performance than it would have with a correct mapping. Nonetheless, using this transformation to dependency format our parsing process achieves 96.20% UAS and 91.26% LAS over the IULA test set, which is in line with the results of the MaltParser [25] trained for Spanish over IULA (93.14% LAS[1]).

6 Discussion

6.1 Conclusions

We presented an architecture for parsing Spanish sentences in a feature based grammar format that includes both syntactic and semantic role information. The parsing process uses LSTM neural networks for splitting sentence into a binary tree, finding the grammar rules to apply and determining the semantic arguments. Our best models achieve 87.5% unlabeled F1 and 81.9% labeled F1 for the AnCora test partition, and 93.6% unlabeled F1 and 88.9% labeled F1 for the IULA test partition. These results were achieved using a neural network that merged the splitting and rule steps of the process, as we found out this greatly improves the performance of these steps. The performance of our method compared to a dependency parser for Spanish is not yet in the state of the art (91.3% versus 93.1%), but we consider that a better process for identifying the dependency labels would make this gap narrower.

For the semantic argument detection step, the intrinsic evaluation yielded 90.9% accuracy and 87.1% macro-F1 over the AnCora test set. However, its performance dropped to 70.9% when considering the extrinsic evaluation of the

[1] http://www.iula.upf.edu/recurs01_mpars_uk.htm

whole process. This could mean that, although the syntactic steps correctly identify many features, the detected features do not contain all the information the semantic argument process needs in order to correctly determine the labels.

6.2 Future work

The parsing process we use is strictly greedy in all steps: we take the most likely split according to the splitter model, then the most likely rule according to the rule model, and finally the most likely argument according to the argument network. However, as the outputs of all the networks are probability distributions (over words or over labels) it is possible to consider multiple outputs of the parsing process and assign them probabilities. This could be used to feed a beam search style algorithm that could get better results by re-ranking candidates using other statistical models.

It might also be interesting to include argument structure information in the IULA corpus in order to train a complete model for it. One way of doing this is using the argument detection network over the gold IULA segments and manually correcting the outputs.

We would also like to try this parsing architecture on other languages, but a corpus with at least syntactic and semantic argument information is needed in order to reproduce the whole process. One first step would be doing this for English, as there are many linguistic resources for this language.

Finally, we could try to combine the three steps of the process into one neural network that considers the context of the whole sentence (or some constituents) at the same time in order to leverage all the information and improve the argument detection step.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015), http://tensorflow.org/, software available from tensorflow.org
2. Azzinnari, A., Martínez, A.: Representación de Palabras en Espacios de Vectores. Proyecto de grado, Universidad de la República, Uruguay (2016)
3. Ballesteros, M., Herrera, J., Francisco, V., Gervás, P.: Improving parsing accuracy for spanish using maltparser. Procesamiento del Lenguaje Natural **44** (2010)
4. Bonial, C., Babko-Malaya, O., Choi, J.D., Hwang, J., Palmer, M.: PropBank annotation guidelines. Center for Computational Language and Education Research Institute of Cognitive Science University of Colorado at Boulder (2010)
5. Buchholz, S., Marsi, E.: Conll-x shared task on multilingual dependency parsing. In: Proceedings of the tenth conference on computational natural language learning. pp. 149–164. Association for Computational Linguistics (2006)

6. Castellón, I., Fernández-Montraveta, A., Vázquez, G., Alonso, L., Capilla, J.: The sensem corpus: a corpus annotated at the syntactic and semantic level. In: 5th International Conference on Language Resources and Evaluation (LREC 2006). Citeseer (2006)
7. Chiruzzo, L., Wonsever, D.: Transforming the AnCora corpus to HPSG. In: Arnold, D., Butt, M., Crysmann, B., King, T.H., Müller, S. (eds.) Proceedings of the Joint 2016 Conference on Head-driven Phrase Structure Grammar and Lexical Functional Grammar, Polish Academy of Sciences, Warsaw, Poland. pp. 182–193. CSLI Publications, Stanford, CA (2016), http://cslipublications.stanford.edu/HPSG/2016/headlex2016-chiruzzo-wonsever.pdf
8. Chiruzzo, L., Wonsever, D.: Spanish HPSG Treebank based on the AnCora Corpus. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018) (2018)
9. Chiruzzo, L., Wonsever, D.: Building a supertagger for Spanish HPSG. Computer Speech & Language **54**, 44–60 (2019)
10. Chollet, F.: Keras. https://github.com/fchollet/keras (2015)
11. Cowan, B., Collins, M.: Morphology and reranking for the statistical parsing of spanish. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing. pp. 795–802. Association for Computational Linguistics (2005)
12. Cross, J., Huang, L.: Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 1–11. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/D16-1001, http://aclweb.org/anthology/D16-1001
13. Dyer, C., Ballesteros, M., Ling, W., Matthews, A., Smith, N.A.: Transition-based dependency parsing with stack long short-term memory. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 334–343. Association for Computational Linguistics (2015). https://doi.org/10.3115/v1/P15-1033, http://aclweb.org/anthology/P15-1033
14. Dyer, C., Kuncoro, A., Ballesteros, M., Smith, N.A.: Recurrent neural network grammars. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 199–209. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/N16-1024, http://aclweb.org/anthology/N16-1024
15. Gaddy, D., Stern, M., Klein, D.: What's going on in neural constituency parsers? an analysis. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 999–1010. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/N18-1091, http://aclweb.org/anthology/N18-1091
16. Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M.A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štepánek, J., Straňák, P., Surdeanu, M., Xue, N., Zhang, Y.: The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task. pp. 1–18. Association for Computational Linguistics (2009), http://aclweb.org/anthology/W09-1201
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)

18. Le Roux, J., Sagot, B., Seddah, D.: Statistical parsing of spanish and data driven lemmatization. In: ACL 2012 Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages (SP-Sem-MRL 2012). pp. 6–pages (2012)
19. Liu, J., Zhang, Y.: Encoder-decoder shift-reduce syntactic parsing. In: Proceedings of the 15th International Conference on Parsing Technologies. pp. 105–114. Association for Computational Linguistics, Pisa, Italy (September 2017), http://www.aclweb.org/anthology/W17-6315
20. Lloberes, M., Castellón, I., Padró, L.: Spanish freeling dependency grammar. In: LREC. vol. 10, pp. 693–699 (2010)
21. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: The Penn Treebank. Computational linguistics **19**(2), 313–330 (1993)
22. Marimon, M.: The spanish resource grammar. In: Proceedings of the International Conference on Language Resources and Evaluation, LREC. pp. 17–23. Valletta, Malta (2010)
23. Marimon, M., Fisas, B., Bel, N., Vivaldi, J., Torner, S., Lorente, M., Vázquez, S., Villegas, M.: The IULA Treebank. In: Lrec. pp. 1920–1926 (2012)
24. Marimon Felipe, M.: The Tibidabo Treebank. Procesamiento del lenguaje natural, 2010, vol. 45, num. 1, p. 113-119 (2010)
25. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., Marsi, E.: MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering **13**(2), 95–135 (2007)
26. Pollard, C., Sag, I.A.: Head-driven phrase structure grammar. University of Chicago Press (1994)
27. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010), http://is.muni.cz/publication/884893/en
28. Shen, Y., Lin, Z., Jacob, A.P., Sordoni, A., Courville, A., Bengio, Y.: Straight to the tree: Constituency parsing with neural syntactic distance. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1171–1180. Association for Computational Linguistics (2018), http://aclweb.org/anthology/P18-1108
29. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 1631–1642. Association for Computational Linguistics (2013), http://aclweb.org/anthology/D13-1170
30. Stern, M., Andreas, J., Klein, D.: A Minimal Span-Based Neural Constituency Parser. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 818–827. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/P17-1076, http://aclweb.org/anthology/P17-1076
31. Taulé, M., Martí, M.A., Recasens, M.: AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In: Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Tapias, D. (eds.) Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08). European Language Resources Association (ELRA), Marrakech, Morocco (may 2008), http://www.lrec-conf.org/proceedings/lrec2008/

32. Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2. pp. 2773–2781. NIPS'15, MIT Press, Cambridge, MA, USA (2015), http://dl.acm.org/citation.cfm?id=2969442.2969550
33. Watanabe, T., Sumita, E.: Transition-based neural constituent parsing. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 1169–1179. Association for Computational Linguistics (2015). https://doi.org/10.3115/v1/P15-1113, http://aclweb.org/anthology/P15-1113