Identifying Patterns for Unsupervised Grammar Induction

Jesús Santamaría
U. Nacional de Educación a Distancia
NLP-IR Group, Madrid, Spain.
`jsant@lsi.uned.es`

Lourdes Araujo
U. Nacional de Educación a Distancia
NLP-IR Group, Madrid, Spain.
`lurdes@lsi.uned.es`

Abstract

This paper describes a new method for unsupervised grammar induction based on the automatic extraction of certain patterns in the texts. Our starting hypothesis is that there exist some classes of words that function as separators, marking the beginning or the end of new constituents. Among these separators we distinguish those which trigger new levels in the parse tree. If we are able to detect these separators we can follow a very simple procedure to identify the constituents of a sentence by taking the classes of words between separators. This paper is devoted to describe the process that we have followed to automatically identify the set of separators from a corpus only annotated with Part-of-Speech (POS) tags. The proposed approach has allowed us to improve the results of previous proposals when parsing sentences from the Wall Street Journal corpus.

1 Introduction

Most works dealing with Grammar Induction (GI) are focused on Supervised Grammar Induction, using a corpus of syntactically annotated sentences, or treebank, as a reference to extract the grammar. The existence of a treebank for the language and for a particular type of texts from which we want to extract the grammar is a great help to GI, even taking into account the theoretical limitations of GI, such as the fact that grammars cannot be correctly identified from positive examples alone (Gold, 1967). But the manual annotation of thousands of sentences is a very expensive task and thus there are many languages for which there are not treebanks available. Even in languages for which there is a treebank, it is usually composed of a particular kind of texts (newspaper articles, for example) and may not be appropriate for other kind of texts, such as tales or poetry. These reasons have led to the appearance of several works focused on unsupervised GI.

Thanks to our knowledge of the language we know that some classes of words are particularly influential to determine the structure of a sentence. For example, let us consider the tree in Figure 1, for which the meaning of the POS tags appears in Table 1. We can observe that the tag MD (Modal) breaks the sentence into two parts. Analogously, in the tree appearing in Figure 2 the POS tag VBZ breaks the sentence. In both cases, we can see that after the breaking tag, a new level appears in the parse tree. A similar effect is observed for other POS tags, such as VB in the tree of Figure 1 and IN in the tree of Figure 2. We call these kind of POS tags *separators*. There are also other POS tags which are frequently the beginning or the end of a constituent[1]. For example in the tree in Figure 1 we can find the sequences (DT NN) and (DT JJ NN), which according to the parse tree are constituents. In the tree in Figure 2 we find the sequence (DT NNP VBG NN). In both trees we can also find sequences beginning with the tag NNP: (NNP NNP) and (NNP CD) in the tree in Figure 1 and (NNP NNP), which appears twice, in the tree in Figure 2. This suggests that there are classes of words with a trend to be the beginning or the end of constituents without giving rise to new levels in the parse tree. We call these POS tags *subseparators*. These observations reflect some of our intuitions, such as the fact that most sentences are composed of a noun phrase and a verb phrase, being frequently the verb the beginning of the verbal phrase, which usually leads to a new level of the parse tree. We also know that determiners (DT) are frequently the beginning of the noun phrases.

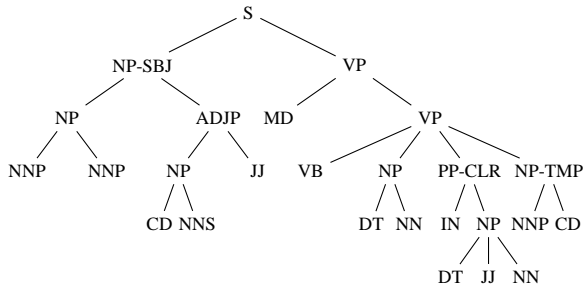[1] Constituents are language units in which we can arrange the structure of a sentence.

Figure 1: Parse tree for the sentence *Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.* from the Penn Treebank
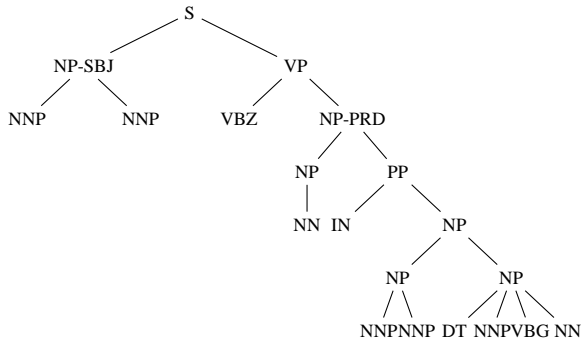

Figure 2: Parse tree for the sentence *Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.* from the Penn Treebank

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition / subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PP$	Possessive pronoun
RB	Adverb.
RBR	Adverb, comparative
RBS	Adverb., superlative
RP	Particle
SYM	Symbol (mathematical or scientific)
TO	To
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund / present participle
VBN	Verb, past participle
VBP	Verb, non-3rd ps. sing. present
VBZ	Verb, 3rd ps. sing. present
WDT	wh-determiner
WP	wh-pronoun
WP$	Possessive wh-pronoun
WRB	wh-adverb

Table 1: Alphabetical list of part-of-speech tags used in the Penn Treebank, the corpus used in our experiments

At this point we could either try to figure out what is the set of tags which work as separators, or to compute them from a parsed corpus for the considered language, provided it is available. However, because we do not want to rely on the existence of a treebank for the corresponding language and type of texts we have done something different: we have devised a statistical procedure to automatically capture the word classes which function as separators. In this way our approach can be applied to most languages, and apart from providing a tool for extracting grammars and parsing sentences, it can be useful to study the different classes of words that work as separators in different languages.

Our statistical mechanism to detect separators is applied to a corpus of sentences annotated with POS tags. This is not a strong requirement since there are very accurate POS taggers (about 97%) for many languages. The grammar that we obtain does not specify the left-hand-side of the rules, but only sequences of POS tags that are constituents.

At this point we have followed the Klein and Manning (2005) setting for the problem, which allows us to compare our results to theirs. As far as we know these are the best results obtained so far for unsupervised GI using a monolingual corpus. As they do, we have used the Penn treebank (Marcus et al., 1994) for our experiments, employing the syntactic annotations that it provides for evaluation purposes only. Specifically, we have used WSJ10, composed of 6842 sentences, which is the subset of the Wall Street Journal section of the Penn Treebank, containing only those sentences of 10 words or less after removing punctuation and null elements, such as $, ", etc.

The rest of the paper is organized as follows: section 2 reviews some related works; section 3 describes the details of the proposal to automatically extract the separators from a POS tagged corpus; section 4 is devoted to describe the procedure to find a parse tree using the separators; section

5 presents and discusses the experimental results, and section 6 draws the main conclusions of this work.

2   State of the Art

A growing interest in unsupervised GI has been observed recently with the appearance of several works in the topic. Some of these works have focused on finding patterns of words (Solan et al., 2005) more than syntactic structures. It has been noted that the rules produced by GI can also be interpreted semantically (David et al., 2003), where a non-terminal describes interchangeable elements which are instances of the same concepts.

Distributional approaches to unsupervised GI exploit the principle of substitutability: constituents of the same type may be exchanged with one another without affecting the syntax of the surrounding context. Distributional approaches to grammar induction fall into two categories, depending on their treatment of nested structure. The first category covers Expectation-Maximization (EM) systems (Dempster et al., 1977). These systems propose constituents based on analysis of the text, and then select a non-contradictory combination of constituents for each sentence that maximizes a given metric, usually parsing probability. One of the most successful proposals in this area is the one by Klein and Manning (2005), which, as mentioned before, starts from a corpus labelled only with POS tags. The key idea of the model proposed in this work is that constituents appear in constituent contexts. However, the EM algorithm presents some serious problems: it is very slow (Lari and Young, 1990), and is easily trapped in local maxima (Carroll and Charniak, 1992). Alignment Based Learning (ABL) (van Zaanen and Leeds, 2000) is the only EM system applied directly to raw text. However, ABL is relatively inefficient and has only been applied to small corpora. Brooks (Brooks, 2006) reverses the notion of distributional approaches: if we can identify "surrounding context" by observation, we can hypothesize that word sequences occurring in that context will be constituents of the same type. He describes a simplified model of distributional analysis (for raw test) which uses heuristics to reduce the number of candidate constituents under consideration. This is an interesting idea in spite that Brook showed that the system was only capable of learning a small subset of constituent structures in a large test corpus.

The second category is that of incremental learning systems. An incremental system analyzes a corpus in a bottom-up fashion: each time a new constituent type is found it is inserted into the corpus to provide data for later learning. The EMILE (Adriaans, 1999) and ADIOS (David et al., 2003) systems are examples for this category, not yet evaluated on large corpora.

Bilingual experiments have been also conducted with the aim to exploit information from one language to disambiguate another. Usually such a setting requires a parallel corpus or another annotated data that ties the two languages. Cohen and Smith (2009) use the English and Chinese treebanks, which are not parallel corpora, to train parsers for both languages jointly. Their results shown that the performance on English improved in the bilingual setting. Another related work (Snyder et al., 2009) uses three corpora of parallel text. Their approach is closer to the unsupervised bilingual parsing model developed by Kuhn (2004), which aims to improve monolingual performance.

The approach considered in this work follows a different direction, trying to identify certain patterns that can determine the structure of the parse trees.

3   Extracting Separators from the Corpus

To automatically extract the set of separators and sub-separators from a corpus of POS tagged sentences we start from some assumptions:

- The most frequent sequence (of any length) of POS tags in the corpus is a constituent, that we call *safe constituent* (sc). It is quite a sensible assumption, since we can expect that at least for the most frequent constituent the number of occurrences overwhelms the number of sequences appearing by chance.

- We also assume that the POS tag on the left, L_{sc}, and on the right, R_{sc}, of the safe constituent are a kind of context for other sequences that play the same role. According to this, other extended sequences with L_{sc} and R_{sc} at the ends but with other POS tags inside are also considered constituents. This assumption is somehow related to the Klein and Manning's (2005) idea underlying their unsupervised GI proposal. According

to them, constituents appear in constituent contexts. Their model exploits the fact that long constituents often have short, common equivalents, which appear in similar contexts and whose constituency as a grammar rule is more easily found.

- According to the previous point, we use the tag on the left (L_{sc}) and on the right (R_{sc}) of the safe constituent as discriminant with respect to which to study the behavior of each POS tag. A POS tag E can have a bias to be inside the safe constituent, to be outside the safe constituent (separator), or not to have a bias at all (sub-separator). We define the *determining side* of a tag E, as the end tag, L_{sc} or R_{sc}, of the *sc* with the greater difference on the number of occurrences of E on both sides of the end tag. For example, if the ratio of occurrences of E on the left and on the right of L_{sc} is smaller (they are more different) than the ratio of E on the left and on the right of R_{sc}, then L_{sc} is the determining side of E, $ds(E)$[2]. Then:

  - E is considered a separator in the following cases:
    * if L_{sc} is the determining side for E and E appears a 75% more often to the left of L_{sc} than to the right (the 75% has been fixed after some estimates described below), or
    * if R_{sc} is the determining side for E and E appears a 75% more often to the right of R_{sc} than to the left.
  - E is considered a sub-separator if the following conditions hold:
    * if L_{sc} is the determining side for E and E appears a 75% less often to the left of L_{sc} than to the right (the ratios are very similar) , or
    * if R_{sc} is the determining side for E and E appears a 75% less often to the right of R_{sc} than to the left.
  - In the remaining cases E is considered to be part of a constituent (the preference is to be inside).

Let us introduce some notation to define more formally the separators and sub-separators. Let

---

[2]If the number of occurrences of E on any side of L_{sc} or R_{sc} is zero, then we compare differences between occurrences instead of ratios.

$\#(E_1, \cdots, E_n)$ be the number of occurrences of the sequence of tags $(E_1, \cdots, E_n)$. We define a predicate *sim* to denote the similarity between the number of occurrences of a sequence of two tags and the one with reverse order, as

$$sim(E_1, E_2)) =$$

$$\frac{\#(E_1, E_2)}{\#(E_2, E_1)} \geq 0.75 \text{ if } \#(E_1, E_2) \leq \#(E_2, E_1)$$

$$\frac{\#(E_2, E_1)}{\#(E_1, E_2)} \geq 0.75 \text{ if } \#(E_2, E_1) \leq \#(E_1, E_2)$$

Then a tag E is considered a separator if the following predicate is true:

$$sep(L_{sc}, E, R_{sc}) =$$
$$(sd(L_{sc}) \wedge (\#(E, L_{sc}) > \#(L_{sc}, E) \wedge$$
$$\neg sim(E, L_{sc}))) \vee$$
$$(sd(R_{sc}) \wedge (\#(R_{sc}, E) > \#(E, R_{sc}) \wedge \neg sim(E, R_{sc}))$$

A tag is considered a sub-separator when the following predicate is true:

$$subsep(L_{sc}, E, R_{sc}) =$$
$$(sd(L_{sc}) \wedge sim(E, L_{sc})) \vee$$
$$(sd(R_{sc}) \wedge sim(E, R_{sc}))$$

We have computed the number of occurrences of every sequence of POS tags in the corpus, finding that the most frequent sequence of tags is (DT,NN). This sequence, which is our safe constituent, appears 2222 times in the considered corpus WSJ10.

Applying our procedure to the corpus we have obtained the following sets of separators and sub-separators:

Separators	MD, PRP, IN, RB, RBR, CC, TO, VB, VBD, VBN, VBZ, VBP, VBG, EX, LS, RP, UH, WP, WRB, WDT
Sub-separators	DT, PDT, POS, SYM, NN, NNS, NNP, NNPS

For selecting a threshold value to discriminate the preference of a POS tag to be inside or outside of a constituent we have studied the results obtained for different threshold values greater than 50%. Table 2 shows the results. We can observe all of them are very similar for all the thresholds, as long as they are greater than 50%. Analyzing the set of POS-tags that have been classified as separators and sub-separators with each threshold we have found that the only differences are that the tag *POS* (Possessive ending), which is classified as sub-separator using a threshold between 50%

and 75%, is classified as separator using higher thresholds, and the tag *SYM* (Symbol), which is classified as sub-separator using a threshold between 50% and 75%, is classified neither as a separator nor as a sub-separator using higher thresholds. We have adopted a threshold value of 75% because higher values can be too restrictive, and in fact provide worse results.

Similarity	F1
55%	74.55
65%	74.55
75%	74.55
85%	72.24
95%	72.24

Table 2: F-measure results obtained for different values of the threshold used to classify the set of POS-tags.

Sub-separators can be grouped to their right or to their left, depending on the case. In order to measure the bias of each of them for one direction or another we have compared the number of occurrences of the most frequent sequence composed of the sub-separator and a POS tag on the right and on the left. We choose as preference direction for a sub-separator the corresponding to the most frequent sequence. Table 3 shows the results obtained, the preference direction of each sub-separator appearing in the last column. In the case of NNP, for which the frequency of the most frequent tag to the right and to the left are the same, we have looked at the second most frequent sequence to choose the grouping direction.

sub-sep	left freq.	right freq.	D
DT	(DT, NN)(2222)	(IN,DT)(894)	L
PDT	(PDT,DT)(28)	(NN,PDT)(14)	L
POS	(POS, NN)(169)	(NNP, POS)(223)	R
SYM	(SYM, IN)(11)	(NN,SYM)(4)	L
NN	(NN, IN)(892)	(DT,NN)(2222)	R
NNS	(NNS, VBP)(591)	(JJ,NNS)(797)	R
NNP	(NNP, NNP)(2127)	(NNP,NNP)(2127)	R
NNPS	(NNPS, NNP)(42)	(NNP,NNPS)(82)	R

Table 3: Preference direction to which each sub-separator clusters. The first column corresponds to the sub-separator, the second one to the most frequent sequence composed of the sub-separator and a tag on its right, the third one to the most frequent sequence of the sub-separator and a tag on its left, and the last column to the resulting direction.

4 Identifying Constituents

Once we have the sets of separators and sub-separators the procedure to identify the constituents of each sentence is as follows:

- We identify the separators in the sentence. For example, if we consider the sentence:

  CC DT NN IN NNP NNP POS NN VBZ

  the separators are marked in boldface:

  CC DT NN **IN** NNP NNP POS NN **VBZ**

- The next step is to split the sentence according to the separators. The first separator which is a verb, if any, is used to split the sentence into two parts. Each separator can give rise to two groups: one composed of the tag sequence between the separator and the next separator, and another one which includes the separator and the POS tags up to the end of the part of the sentence in which it appears (usually sentences are divided into two parts using the first separator which is a verb). In our example, this mechanism leads to the following structure:

  [[**CC** [DT NN] [**IN** [NNP NNP POS NN]]] [**VBZ**]]

- Now it is the turn of the sub-separators (DT, PDT, POS, SYM, NN, NNS, NNP, NNPS), which are underlined in the sentence:

  [[**CC** [<u>DT NN</u>] [IN [<u>NNP NNP POS NN</u>]]] [**VBZ**]]

- Finally, each group of the sentence is split according to the sub-separators. Each sub-separator has been assigned a preference direction to form the group with the next POS tag. Looking at Table 3, which tells us the direction in which each sub-separator forms the group, we apply this step to our sentence example, obtaining:

  [[CC [DT NN] [IN [[NNP NNP POS] NN]]] [VBZ]]

The sub-separator DT is grouped with the tags on its right, while NN is grouped with the tags on its left, thus composing the group (DT NN). When two or more sub-separators appear in a sequence, they are grouped together in a unique constituent whenever they have the same grouping direction. In our sentence example this criterion leads to [NNP NNP POS] instead of [NNP[NNP[POS]]]. A constituent finishes if the next POS tag is a separator or if it is a sub-separator that makes groups towards the left. Since POS (Possessive ending) tends to be grouped with the POS tag on its left, it is the end of the constituent.

Figure 3 represents the obtained structure as a parse tree. Figure 4 represents the correct parse tree according to the Penn treebank. We can observe that both structures are very similar. The method based on separators has been able to capture most of the constituent appearing in the parse tree: (DT, NN), (NNP, NNP, POS), (NNP, NNP, POS, NN), (IN, NNP, NNP, POS, NN). The differences between both trees come from our criterion of splitting the sequence of tags into two subsequences using the first verb. This problem will be tackled in the future in a more refined model.
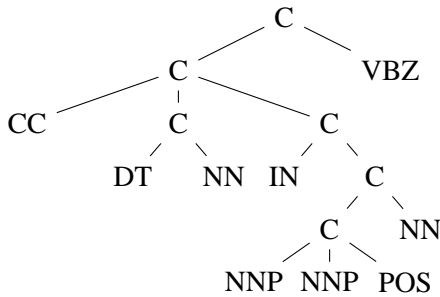

Figure 3: Parse tree for the sentence *And the nose on Mr. Courter's face grows* from the Penn treebank (WSJ), obtained with our separators method.

5 Evaluation

Our proposal has been evaluated by comparing the tree structures produced by the system to the gold-standard trees produced by linguists, which can be found in the Penn Treebank. Because we do not assign class name to our constituents, i.e. a left hand side symbol for the grammar rules, as the linguists do in treebanks, the comparison ignores the class labels, considering only groups of tags.
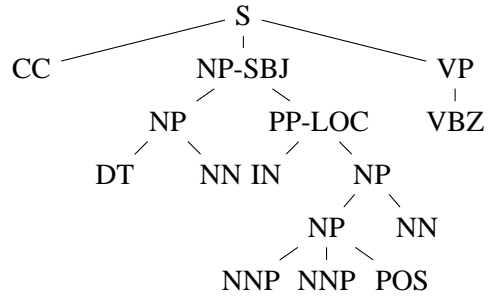

Figure 4: Parse tree appearing in the Penn treebank (WSJ) for the sentence *And the nose on Mr. Courter's face grows*.

The results presented in the work by Klein and Manning (2005) have been our reference, since as far we know they are the best ones obtained so far for unsupervised GI. For the sake of comparison, we have considered the same corpus and the same measures. Accordingly, we performed the experiments on the 6842 sentences[3] of the WSJ10 selection from the Penn treebank Wall Street Journal section.

In order to evaluate the quality of the obtained grammar we have used the most common measures for parsing and grammar induction evaluation: recall, precision, and their harmonic mean (F-measure). They are defined assuming a bracket representation of a parse tree.

Precision is given by the number of brackets in the parse to evaluate which match those in the correct tree and *recall* measures how many of the brackets in the correct tree are in the parse. These measures have counterparts for unlabeled trees, the ones considered in this work – in which the label assigned to each constituent is not checked. Constituents which could not be wrong (those of size one and those spanning the whole sentence) have not been included in the measures.

The definitions of Unlabeled Precision (UP) and Recall (UR) of a proposed corpus $P = [P_i]$ against a gold corpus $G = [G_i]$ are:

$$UP(P,G) = \frac{\sum_i |\text{brackets}(P_i) \cap \text{brackets}(G_i)|}{\sum_i |\text{brackets}(P_i)|},$$
$$UR(P,G) = \frac{\sum_i |\text{brackets}(P_i) \cap \text{brackets}(G_i)|}{\sum_i |\text{brackets}(G_i)|}.$$

Finally, UF (Unlabeled F-measure) is given by:

$$UF = \frac{2 \cdot UP(P,G) \cdot UR(P,G)}{UP(P,G) + UR(P,G)}.$$

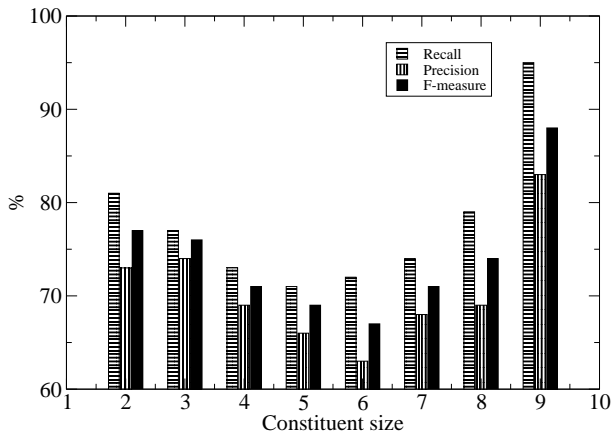---

[3]More precisely sequences of POS tags

Figure 5: Results obtained per constituent size: unlabeled recall, precision, and F-measure.

Figure 5 shows the results of unlabeled recall, precision and F-measure obtained per constituent size. We can observe that recall and precision, and thus the corresponding F-measure, are quite similar for every constituent size. This is important, because obtaining a high F-measure thanks to a very high recall but with a poor precision, is not so useful. We can also observe that the best results are obtained for short and long constituents, with lower values for middle lengths, such as 5 and 6. We believe that this is because intermediate size constituents present more variability. Moreover, for intermediate sizes, the composition of the constituents is more dependent on sub-separators, for which the statistical differences are less significant than for separators.
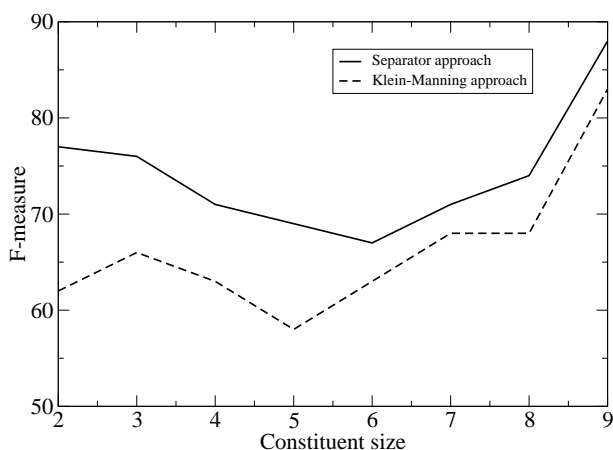

Figure 6: Comparison of the separator approach and Klein and Manning's approach per constituent size.

We have compared our results to those obtained by Klein and Manning (2005) for the same corpus.

Table 4 shows the obtained results for WSJ10. We can observe that we have obtained more balanced values of recall and precision, as well as a better value for the F-measure. Thus the method proposed in this work, that we expect to refine by assigning different probabilities to separators and sub-separators, depending on the context they appear in, provides a very promising approach.

	UR	UP	UF
Separ. A.	77,63%	71,71%	74,55%
KM	80.2%	63.8%	71.1%

Table 4: Results (unlabeled recall, precision, and F-measure), obtained with the separator approach (first row) and with the Klein and Manning approach (second row) for the WSJ10 corpus.

Figure 6 compares the F-measure for the two approaches by constituents length. We can observe that the separator approach obtains better results for all the lengths. The figure also shows that the results per constituent length follow the same trend in both approaches, thus reflecting that the difficulty for middle length constituents is greater.

6 Conclusions

We have proposed a novel approach for unsupervised grammar induction which is based on identifying certain POS tags that very often divide the sentences in particular manners. These separators are obtained from POS tagged texts, thus making the model valid for many languages. The constituents corresponding to a sentence are found by means of a simple procedure based on the separators. This simple method has allowed us to improve the results of previous proposals.

We are currently working in defining a more refined statistical model which takes into account the probability of a tag to be a separator or sub-separator, depending on its context. We plan to apply a similar study to other languages, in order to study the different classes of words that function as separator in each of them.

References

Pieter Adriaans. 1999. Learning Shallow Context-Free Languages under Simple Distributions. Technical Report, Institute for Logic, Language, and Computation, Amsterdam.

David J. Brooks. 2006. Unsupervised grammar induction by distribution and attachment. In *CoNLL-X '06: Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 117–124. Association for Computational Linguistics.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In *Working Notes of the Workshop Statistically-Based NLP Techniques*, pages 1–13. AAAI.

Shay B. Cohen and Noah A. Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 74–82. Association for Computational Linguistics.

Zach Solan David, David Horn, and Shimon Edelman. 2003. Unsupervised efficient learning and representation of language structure. In *Proc. 25th Conference of the Cognitive Science Society*, pages 2577–3596. Erlbaum.

A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Royal statistical Society B*, 39:1–38.

E. Mark Gold. 1967. Language identification in the limit. *Information and Control*, 10(5):447–474.

Dan Klein and Christopher D. Manning. 2005. Natural language grammar induction with a generative constituent-context model. *Pattern Recognition*, 38(9):1407–1419.

Jonas Kuhn. 2004. Experiments in parallel-text based grammar induction. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 470. Association for Computational Linguistics.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

Benjamin Snyder, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1*, pages 73–81. Association for Computational Linguistics.

Zach Solan, David Horn, Eytan Ruppin, and Shimon Edelman. 2005. Unsupervised learning of natural languages. *Proceedings of the National Academy of Sciences of the United States of America*, 102(33):11629–11634.

Menno van Zaanen and Ls Jt Leeds. 2000. Learning structure using alignment based learning. In *Universities of Brighton and Sussex*, pages 75–82.