Unsupervised Argument Identification for Semantic Role Labeling

Omri Abend[1] **Roi Reichart**[2] **Ari Rappoport**[1]

[1]Institute of Computer Science , [2]ICNC
Hebrew University of Jerusalem
{omria01|roiri|arir}@cs.huji.ac.il

Abstract

The task of Semantic Role Labeling (SRL) is often divided into two sub-tasks: verb argument identification, and argument classification. Current SRL algorithms show lower results on the identification sub-task. Moreover, most SRL algorithms are supervised, relying on large amounts of manually created data. In this paper we present an unsupervised algorithm for identifying verb arguments, where the only type of annotation required is POS tagging. The algorithm makes use of a fully unsupervised syntactic parser, using its output in order to detect clauses and gather candidate argument collocation statistics. We evaluate our algorithm on PropBank10, achieving a precision of 56%, as opposed to 47% of a strong baseline. We also obtain an 8% increase in precision for a Spanish corpus. This is the first paper that tackles unsupervised verb argument identification without using manually encoded rules or extensive lexical or syntactic resources.

1 Introduction

Semantic Role Labeling (SRL) is a major NLP task, providing a shallow sentence-level semantic analysis. SRL aims at identifying the relations between the predicates (usually, verbs) in the sentence and their associated arguments.

The SRL task is often viewed as consisting of two parts: argument identification (ARGID) and argument classification. The former aims at identifying the arguments of a given predicate present in the sentence, while the latter determines the type of relation that holds between the identified arguments and their corresponding predicates. The division into two sub-tasks is justified by the fact that they are best addressed using different feature sets (Pradhan et al., 2005). Performance in the ARGID stage is a serious bottleneck for general SRL performance, since only about 81% of the arguments are identified, while about 95% of the identified arguments are labeled correctly (Màrquez et al., 2008).

SRL is a complex task, which is reflected by the algorithms used to address it. A standard SRL algorithm requires thousands to dozens of thousands sentences annotated with POS tags, syntactic annotation and SRL annotation. Current algorithms show impressive results but only for languages and domains where plenty of annotated data is available, e.g., English newspaper texts (see Section 2). Results are markedly lower when testing is on a domain wider than the training one, even in English (see the WSJ-Brown results in (Pradhan et al., 2008)).

Only a small number of works that do not require manually labeled SRL training data have been done (Swier and Stevenson, 2004; Swier and Stevenson, 2005; Grenager and Manning, 2006). These papers have replaced this data with the VerbNet (Kipper et al., 2000) lexical resource or a set of manually written rules and supervised parsers.

A potential answer to the SRL training data bottleneck are unsupervised SRL models that require little to no manual effort for their training. Their output can be used either by itself, or as training material for modern supervised SRL algorithms.

In this paper we present an algorithm for unsupervised argument identification. The only type of annotation required by our algorithm is POS tag-

Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pages 28–36,
Suntec, Singapore, 2-7 August 2009.

ging, which needs relatively little manual effort.

The algorithm consists of two stages. As preprocessing, we use a fully unsupervised parser to parse each sentence. Initially, the set of possible arguments for a given verb consists of all the constituents in the parse tree that do not contain that predicate. The first stage of the algorithm attempts to detect the minimal clause in the sentence that contains the predicate in question. Using this information, it further reduces the possible arguments only to those contained in the minimal clause, and further prunes them according to their position in the parse tree. In the second stage we use pointwise mutual information to estimate the collocation strength between the arguments and the predicate, and use it to filter out instances of weakly collocating predicate argument pairs.

We use two measures to evaluate the performance of our algorithm, precision and F-score. Precision reflects the algorithm's applicability for creating training data to be used by supervised SRL models, while the standard SRL F-score measures the model's performance when used by itself. The first stage of our algorithm is shown to outperform a strong baseline both in terms of F-score and of precision. The second stage is shown to increase precision while maintaining a reasonable recall.

We evaluated our model on sections 2-21 of Propbank. As is customary in unsupervised parsing work (e.g. (Seginer, 2007)), we bounded sentence length by 10 (excluding punctuation). Our first stage obtained a precision of 52.8%, which is more than 6% improvement over the baseline. Our second stage improved precision to nearly 56%, a 9.3% improvement over the baseline. In addition, we carried out experiments on Spanish (on sentences of length bounded by 15, excluding punctuation), achieving an increase of over 7.5% in precision over the baseline. Our algorithm increases F–score as well, showing an 1.8% improvement over the baseline in English and a 2.2% improvement in Spanish.

Section 2 reviews related work. In Section 3 we detail our algorithm. Sections 4 and 5 describe the experimental setup and results.

2 Related Work

The advance of machine learning based approaches in this field owes to the usage of large scale annotated corpora. English is the most studied language, using the FrameNet (FN) (Baker et al., 1998) and PropBank (PB) (Palmer et al., 2005) resources. PB is a corpus well suited for evaluation, since it annotates every non-auxiliary verb in a real corpus (the WSJ sections of the Penn Treebank). PB is a standard corpus for SRL evaluation and was used in the CoNLL SRL shared tasks of 2004 (Carreras and Màrquez, 2004) and 2005 (Carreras and Màrquez, 2005).

Most work on SRL has been supervised, requiring dozens of thousands of SRL annotated training sentences. In addition, most models assume that a syntactic representation of the sentence is given, commonly in the form of a parse tree, a dependency structure or a shallow parse. Obtaining these is quite costly in terms of required human annotation.

The first work to tackle SRL as an independent task is (Gildea and Jurafsky, 2002), which presented a supervised model trained and evaluated on FrameNet. The CoNLL shared tasks of 2004 and 2005 were devoted to SRL, and studied the influence of different syntactic annotations and domain changes on SRL results. *Computational Linguistics* has recently published a special issue on the task (Màrquez et al., 2008), which presents state-of-the-art results and surveys the latest achievements and challenges in the field.

Most approaches to the task use a multi-level approach, separating the task to an ARGID and an argument classification sub-tasks. They then use the unlabeled argument structure (without the semantic roles) as training data for the ARGID stage and the entire data (perhaps with other features) for the classification stage. Better performance is achieved on the classification, where state-of-the-art supervised approaches achieve about 81% F-score on the in-domain identification task, of which about 95% are later labeled correctly (Màrquez et al., 2008).

There have been several exceptions to the standard architecture described in the last paragraph. One suggestion poses the problem of SRL as a sequential tagging of words, training an SVM classifier to determine for each word whether it is inside, outside or in the beginning of an argument (Hacioglu and Ward, 2003). Other works have integrated argument classification and identification into one step (Collobert and Weston, 2007), while others went further and combined the former two along with parsing into a single model (Musillo

and Merlo, 2006).

Work on less supervised methods has been scarce. Swier and Stevenson (2004) and Swier and Stevenson (2005) presented the first model that does not use an SRL annotated corpus. However, they utilize the extensive verb lexicon VerbNet, which lists the possible argument structures allowable for each verb, and supervised syntactic tools. Using VerbNet along with the output of a rule-based chunker (in 2004) and a supervised syntactic parser (in 2005), they spot instances in the corpus that are very similar to the syntactic patterns listed in VerbNet. They then use these as seed for a bootstrapping algorithm, which consequently identifies the verb arguments in the corpus and assigns their semantic roles.

Another less supervised work is that of (Grenager and Manning, 2006), which presents a Bayesian network model for the argument structure of a sentence. They use EM to learn the model's parameters from unannotated data, and use this model to tag a test corpus. However, ARGID was not the task of that work, which dealt solely with argument classification. ARGID was performed by manually-created rules, requiring a supervised or manual syntactic annotation of the corpus to be annotated.

The three works above are relevant but incomparable to our work, due to the extensive amount of supervision (namely, VerbNet and a rule-based or supervised syntactic system) they used, both in detecting the syntactic structure and in detecting the arguments.

Work has been carried out in a few other languages besides English. Chinese has been studied in (Xue, 2008). Experiments on Catalan and Spanish were done in SemEval 2007 (Màrquez et al., 2007) with two participating systems. Attempts to compile corpora for German (Burdchardt et al., 2006) and Arabic (Diab et al., 2008) are also underway. The small number of languages for which extensive SRL annotated data exists reflects the considerable human effort required for such endeavors.

Some SRL works have tried to use unannotated data to improve the performance of a base supervised model. Methods used include bootstrapping approaches (Gildea and Jurafsky, 2002; Kate and Mooney, 2007), where large unannotated corpora were tagged with SRL annotation, later to be used to retrain the SRL model. Another approach used similarity measures either between verbs (Gordon and Swanson, 2007) or between nouns (Gildea and Jurafsky, 2002) to overcome lexical sparsity. These measures were estimated using statistics gathered from corpora augmenting the model's training data, and were then utilized to generalize across similar verbs or similar arguments.

Attempts to substitute full constituency parsing by other sources of syntactic information have been carried out in the SRL community. Suggestions include posing SRL as a sequence labeling problem (Màrquez et al., 2005) or as an edge tagging problem in a dependency representation (Hacioglu, 2004). Punyakanok et al. (2008) provide a detailed comparison between the impact of using shallow vs. full constituency syntactic information in an English SRL system. Their results clearly demonstrate the advantage of using full annotation.

The identification of arguments has also been carried out in the context of automatic subcategorization frame acquisition. Notable examples include (Manning, 1993; Briscoe and Carroll, 1997; Korhonen, 2002) who all used statistical hypothesis testing to filter a parser's output for arguments, with the goal of compiling verb subcategorization lexicons. However, these works differ from ours as they attempt to characterize the behavior of a verb type, by collecting statistics from various instances of that verb, and not to determine which are the arguments of specific verb instances.

The algorithm presented in this paper performs unsupervised clause detection as an intermediate step towards argument identification. Supervised clause detection was also tackled as a separate task, notably in the CoNLL 2001 shared task (Tjong Kim Sang and Dèjean, 2001). Clause information has been applied to accelerating a syntactic parser (Glaysher and Moldovan, 2006).

3 Algorithm

In this section we describe our algorithm. It consists of two stages, each of which reduces the set of argument candidates, which a-priori contains all consecutive sequences of words that do not contain the predicate in question.

3.1 Algorithm overview

As pre-processing, we use an unsupervised parser that generates an unlabeled parse tree for each sen-

tence (Seginer, 2007). This parser is unique in that it is able to induce a bracketing (unlabeled parsing) from raw text (without even using POS tags) achieving state-of-the-art results. Since our algorithm uses millions to tens of millions sentences, we must use very fast tools. The parser's high speed (thousands of words per second) enables us to process these large amounts of data.

The only type of supervised annotation we use is POS tagging. We use the taggers MXPOST (Ratnaparkhi, 1996) for English and TreeTagger (Schmid, 1994) for Spanish, to obtain POS tags for our model.

The first stage of our algorithm uses linguistically motivated considerations to reduce the set of possible arguments. It does so by confining the set of argument candidates only to those constituents which obey the following two restrictions. First, they should be contained in the minimal clause containing the predicate. Second, they should be k-th degree cousins of the predicate in the parse tree. We propose a novel algorithm for clause detection and use its output to determine which of the constituents obey these two restrictions.

The second stage of the algorithm uses pointwise mutual information to rule out constituents that appear to be weakly collocating with the predicate in question. Since a predicate greatly restricts the type of arguments with which it may appear (this is often referred to as "selectional restrictions"), we expect it to have certain characteristic arguments with which it is likely to collocate.

3.2 Clause detection stage

The main idea behind this stage is the observation that most of the arguments of a predicate are contained within the minimal clause that contains the predicate. We tested this on our development data – section 24 of the WSJ PTB, where we saw that 86% of the arguments that are also constituents (in the gold standard parse) were indeed contained in that minimal clause (as defined by the tree label types in the gold standard parse that denote a clause, e.g., S, SBAR). Since we are not provided with clause annotation (or any label), we attempted to detect them in an unsupervised manner. Our algorithm attempts to find sub-trees within the parse tree, whose structure resembles the structure of a full sentence. This approximates the notion of a clause.

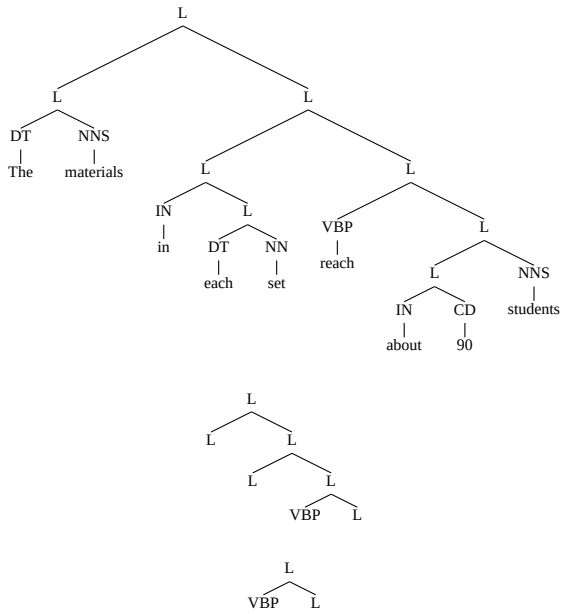

Figure 1: An example of an unlabeled POS tagged parse tree. The middle tree is the ST of 'reach' with the root as the encoded ancestor. The bottom one is the ST with its parent as the encoded ancestor.

Statistics gathering. In order to detect which of the verb's ancestors is the minimal clause, we score each of the ancestors and select the one that maximizes the score. We represent each ancestor using its *Spinal Tree* (ST). The ST of a given verb's ancestor is obtained by replacing all the constituents that do not contain the verb by a leaf having a label. This effectively encodes all the k-th degree cousins of the verb (for every k). The leaf labels are either the word's POS in case the constituent is a leaf, or the generic label "L" denoting a non-leaf. See Figure 1 for an example.

In this stage we collect statistics of the occurrences of STs in a large corpus. For every ST in the corpus, we count the number of times it occurs in a form we consider to be a clause (positive examples), and the number of times it appears in other forms (negative examples).

Positive examples are divided into two main types. First, when the ST encodes the root ancestor (as in the middle tree of Figure 1); second, when the ancestor complies to a clause lexico-syntactic pattern. In many languages there is a small set of lexico-syntactic patterns that mark a clause, e.g. the English 'that', the German 'dass' and the Spanish 'que'. The patterns which were used in our experiments are shown in Figure 2.

For each verb instance, we traverse over its an-

English
TO + VB. The constituent starts with "to" followed by a verb in infinitive form.
WP. The constituent is preceded by a Wh-pronoun.
That. The constituent is preceded by a "that" marked by an "IN" POS tag indicating that it is a subordinating conjunction.
Spanish
CQUE. The constituent is preceded by a word with the POS "CQUE" which denotes the word "que" as a conjunction.
INT. The constituent is preceded by a word with the POS "INT" which denotes an interrogative pronoun.
CSUB. The constituent is preceded by a word with one of the POSs "CSUBF", "CSUBI" or "CSUBX", which denote a subordinating conjunction.

Figure 2: The set of lexico-syntactic patterns that mark clauses which were used by our model.

cestors from top to bottom. For each of them we update the following counters: $sentence(ST)$ for the root ancestor's ST, $pattern_i(ST)$ for the ones complying to the i-th lexico-syntactic pattern and $negative(ST)$ for the other ancestors[1].

Clause detection. At test time, when detecting the minimal clause of a verb instance, we use the statistics collected in the previous stage. Denote the ancestors of the verb with $A_1 \dots A_m$. For each of them, we calculate $clause(ST_{A_j})$ and $total(ST_{A_j})$. $clause(ST_{A_j})$ is the sum of $sentence(ST_{A_j})$ and $pattern_i(ST_{A_j})$ if this ancestor complies to the i-th pattern (if there is no such pattern, $clause(ST_{A_j})$ is equal to $sentence(ST_{A_j})$). $total(ST_{A_j})$ is the sum of $clause(ST_{A_j})$ and $negative(ST_{A_j})$.

The selected ancestor is given by:

$$(1)\ A_{max} = argmax_{A_j} \frac{clause(ST_{A_j})}{total(ST_{A_j})}$$

An ST whose $total(ST)$ is less than a small threshold[2] is not considered a candidate to be the minimal clause, since its statistics may be unreliable. In case of a tie, we choose the lowest constituent that obtained the maximal score. If there is only one verb in the sentence[3] or if $clause(ST_{A_j}) = 0$ for every $1 \leq j \leq m$, we choose the top level constituent by default to be the minimal clause containing the verb. Otherwise, the minimal clause is defined to be the yield of the selected ancestor.

Argument identification. For each predicate in the corpus, its argument candidates are now defined to be the constituents contained in the minimal clause containing the predicate. However, these constituents may be (and are) nested within each other, violating a major restriction on SRL arguments. Hence we now prune our set, by keeping only the siblings of all of the verb's ancestors, as is common in supervised SRL (Xue and Palmer, 2004).

3.3 Using collocations

We use the following observation to filter out some superfluous argument candidates: since the arguments of a predicate many times bear a semantic connection with that predicate, they consequently tend to collocate with it.

We collect collocation statistics from a large corpus, which we annotate with parse trees and POS tags. We mark arguments using the argument detection algorithm described in the previous two sections, and extract all (predicate, argument) pairs appearing in the corpus. Recall that for each sentence, the arguments are a subset of the constituents in the parse tree.

We use two representations of an argument: one is the POS tag sequence of the terminals contained in the argument, the other is its head word[4]. The predicate is represented as the conjunction of its lemma with its POS tag.

Denote the number of times a predicate x appeared with an argument y by n_{xy}. Denote the total number of (predicate, argument) pairs by N. Using these notations, we define the following quantities: $n_x = \Sigma_y n_{xy}$, $n_y = \Sigma_x n_{xy}$, $p(x) = \frac{n_x}{N}$, $p(y) = \frac{n_y}{N}$ and $p(x,y) = \frac{n_{xy}}{N}$. The pointwise mutual information of x and y is then given by:

[1] If while traversing the tree, we encounter an ancestor whose first word is preceded by a coordinating conjunction (marked by the POS tag "CC"), we refrain from performing any additional counter updates. Structures containing coordinating conjunctions tend not to obey our lexico-syntactic rules.

[2] We used 4 per million sentences, derived from development data.

[3] In this case, every argument in the sentence must be related to that verb.

[4] Since we do not have syntactic labels, we use an approximate notion. For English we use the Bikel parser default head word rules (Bikel, 2004). For Spanish, we use the leftmost word.

(2) $PMI(x,y) = log\frac{p(x,y)}{p(x)\cdot p(y)} = log\frac{n_{xy}}{(n_x \cdot n_y)/N}$

PMI effectively measures the ratio between the number of times x and y appeared together and the number of times they were expected to appear, had they been independent.

At test time, when an (x,y) pair is observed, we check if $PMI(x,y)$, computed on the large corpus, is lower than a threshold α for either of x's representations. If this holds, for at least one representation, we prune all instances of that (x,y) pair. The parameter α may be selected differently for each of the argument representations.

In order to avoid using unreliable statistics, we apply this for a given pair only if $\frac{n_x \cdot n_y}{N} > r$, for some parameter r. That is, we consider $PMI(x,y)$ to be reliable, only if the denominator in equation (2) is sufficiently large.

4 Experimental Setup

Corpora. We used the PropBank corpus for development and for evaluation on English. Section 24 was used for the development of our model, and sections 2 to 21 were used as our test data. The free parameters of the collocation extraction phase were tuned on the development data. Following the unsupervised parsing literature, multiple brackets and brackets covering a single word are omitted. We exclude punctuation according to the scheme of (Klein, 2005). As is customary in unsupervised parsing (e.g. (Seginer, 2007)), we bounded the lengths of the sentences in the corpus to be at most 10 (excluding punctuation). This results in 207 sentences in the development data, containing a total of 132 different verbs and 173 verb instances (of the non-auxiliary verbs in the SRL task, see 'evaluation' below) having 403 arguments. The test data has 6007 sentences containing 1008 different verbs and 5130 verb instances (as above) having 12436 arguments.

Our algorithm requires large amounts of data to gather argument structure and collocation patterns. For the statistics gathering phase of the clause detection algorithm, we used 4.5M sentences of the NANC (Graff, 1995) corpus, bounding their length in the same manner. In order to extract collocations, we used 2M sentences from the British National Corpus (Burnard, 2000) and about 29M sentences from the Dmoz corpus (Gabrilovich and Markovitch, 2005). Dmoz is a web corpus obtained by crawling and cleaning the URLs in the Open Directory Project (dmoz.org). All of the above corpora were parsed using Seginer's parser and POS-tagged by MXPOST (Ratnaparkhi, 1996).

For our experiments on Spanish, we used 3.3M sentences of length at most 15 (excluding punctuation) extracted from the Spanish Wikipedia. Here we chose to bound the length by 15 due to the smaller size of the available test corpus. The same data was used both for the first and the second stages. Our development and test data were taken from the training data released for the SemEval 2007 task on semantic annotation of Spanish (Màrquez et al., 2007). This data consisted of 1048 sentences of length up to 15, from which 200 were randomly selected as our development data and 848 as our test data. The development data included 313 verb instances while the test data included 1279. All corpora were parsed using the Seginer parser and tagged by the "TreeTagger" (Schmid, 1994).

Baselines. Since this is the first paper, to our knowledge, which addresses the problem of unsupervised argument identification, we do not have any previous results to compare to. We instead compare to a baseline which marks all k-th degree cousins of the predicate (for every k) as arguments (this is the second pruning we use in the clause detection stage). We name this baseline the ALL COUSINS baseline. We note that a random baseline would score very poorly since any sequence of terminals which does not contain the predicate is a possible candidate. Therefore, beating this random baseline is trivial.

Evaluation. Evaluation is carried out using standard SRL evaluation software[5]. The algorithm is provided with a list of predicates, whose arguments it needs to annotate. For the task addressed in this paper, non-consecutive parts of arguments are treated as full arguments. A match is considered each time an argument in the gold standard data matches a marked argument in our model's output. An unmatched argument is an argument which appears in the gold standard data, and fails to appear in our model's output, and an excessive argument is an argument which appears in our model's output but does not appear in the gold standard. Precision and recall are defined accordingly. We report an F-score as well (the harmonic mean of precision and recall). We do not attempt

[5] http://www.lsi.upc.edu/∼srlconll/soft.html#software.

to identify multi-word verbs, and therefore do not report the model's performance in identifying verb boundaries.

Since our model detects clauses as an intermediate product, we provide a separate evaluation of this task for the English corpus. We show results on our development data. We use the standard parsing F-score evaluation measure. As a gold standard in this evaluation, we mark for each of the verbs in our development data the minimal clause containing it. A minimal clause is the lowest ancestor of the verb in the parse tree that has a syntactic label of a clause according to the gold standard parse of the PTB. A verb is any terminal marked by one of the POS tags of type verb according to the gold standard POS tags of the PTB.

5 Results

Our results are shown in Table 1. The left section presents results on English and the right section presents results on Spanish. The top line lists results of the clause detection stage alone. The next two lines list results of the full algorithm (clause detection + collocations) in two different settings of the collocation stage. The bottom line presents the performance of the ALL COUSINS baseline.

In the "Collocation Maximum Precision" setting the parameters of the collocation stage (α and r) were generally tuned such that maximal precision is achieved while preserving a minimal recall level (40% for English, 20% for Spanish on the development data). In the "Collocation Maximum F-score" the collocation parameters were generally tuned such that the maximum possible F-score for the collocation algorithm is achieved.

The best or close to best F-score is achieved when using the clause detection algorithm alone (59.14% for English, 23.34% for Spanish). Note that for both English and Spanish F-score improvements are achieved via a precision improvement that is more significant than the recall degradation. F-score maximization would be the aim of a system that uses the output of our unsupervised ARGID by itself.

The "Collocation Maximum Precision" achieves the best precision level (55.97% for English, 21.8% for Spanish) but at the expense of the largest recall loss. Still, it maintains a reasonable level of recall. The "Collocation Maximum F-score" is an example of a model that provides a precision improvement (over both the baseline and the clause detection stage) with a relatively small recall degradation. In the Spanish experiments its F-score (23.87%) is even a bit higher than that of the clause detection stage (23.34%).

The full two–stage algorithm (clause detection + collocations) should thus be used when we intend to use the model's output as training data for supervised SRL engines or supervised ARGID algorithms.

In our algorithm, the initial set of potential arguments consists of constituents in the Seginer parser's parse tree. Consequently the fraction of arguments that are also constituents (81.87% for English and 51.83% for Spanish) poses an upper bound on our algorithm's recall. Note that the recall of the ALL COUSINS baseline is 74.27% (45.75%) for English (Spanish). This score emphasizes the baseline's strength, and justifies the restriction that the arguments should be k-th cousins of the predicate. The difference between these bounds for the two languages provides a partial explanation for the corresponding gap in the algorithm's performance.

Figure 3 shows the precision of the collocation model (on development data) as a function of the amount of data it was given. We can see that the algorithm reaches saturation at about 5M sentences. It achieves this precision while maintaining a reasonable recall (an average recall of 43.1% after saturation). The parameters of the collocation model were separately tuned for each corpus size, and the graph displays the maximum which was obtained for each of the corpus sizes.

To better understand our model's performance, we performed experiments on the English corpus to test how well its first stage detects clauses. Clause detection is used by our algorithm as a step towards argument identification, but it can be of potential benefit for other purposes as well (see Section 2). The results are 23.88% recall and 40% precision. As in the ARGID task, a random selection of arguments would have yielded an extremely poor result.

6 Conclusion

In this work we presented the first algorithm for argument identification that uses neither supervised syntactic annotation nor SRL tagged data. We have experimented on two languages: English and Spanish. The straightforward adaptability of un-

	English (Test Data)			Spanish (Test Data)		
	Precision	Recall	F1	Precision	Recall	F1
Clause Detection	52.84	67.14	**59.14**	18.00	33.19	23.34
Collocation Maximum F–score	54.11	63.53	58.44	20.22	29.13	**23.87**
Collocation Maximum Precision	**55.97**	40.02	46.67	**21.80**	18.47	20.00
ALL COUSINS baseline	46.71	74.27	57.35	14.16	45.75	21.62

Table 1: Precision, Recall and F1 score for the different stages of our algorithm. Results are given for English (PTB, sentences length bounded by 10, left part of the table) and Spanish (SemEval 2007 Spanish SRL task, right part of the table). The results of the collocation (second) stage are given in two configurations, Collocation Maximum F-score and Collocation Maximum Precision (see text). The upper bounds on Recall, obtained by taking all arguments output by our unsupervised parser, are 81.87% for English and 51.83% for Spanish.

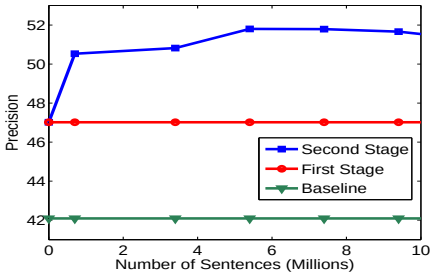

Figure 3: The performance of the second stage on English (squares) vs. corpus size. The precision of the baseline (triangles) and of the first stage (circles) is displayed for reference. The graph indicates the maximum precision obtained for each corpus size. The graph reaches saturation at about 5M sentences. The average recall of the sampled points from there on is 43.1%. Experiments were performed on the English development data.

supervised models to different languages is one of their most appealing characteristics. The recent availability of unsupervised syntactic parsers has offered an opportunity to conduct research on SRL, without reliance on supervised syntactic annotation. This work is the first to address the application of unsupervised parses to an SRL related task.

Our model displayed an increase in precision of 9% in English and 8% in Spanish over a strong baseline. Precision is of particular interest in this context, as instances tagged by high quality annotation could be later used as training data for supervised SRL algorithms. In terms of F–score, our model showed an increase of 1.8% in English and of 2.2% in Spanish over the baseline.

Although the quality of unsupervised parses is currently low (compared to that of supervised approaches), using great amounts of data in identifying recurring structures may reduce noise and in addition address sparsity. The techniques presented in this paper are based on this observation, using around 35M sentences in total for English and 3.3M sentences for Spanish.

As this is the first work which addressed unsupervised ARGID, many questions remain to be explored. Interesting issues to address include assessing the utility of the proposed methods when supervised parses are given, comparing our model to systems with no access to unsupervised parses and conducting evaluation using more relaxed measures.

Unsupervised methods for syntactic tasks have matured substantially in the last few years. Notable examples are (Clark, 2003) for unsupervised POS tagging and (Smith and Eisner, 2006) for unsupervised dependency parsing. Adapting our algorithm to use the output of these models, either to reduce the little supervision our algorithm requires (POS tagging) or to provide complementary syntactic information, is an interesting challenge for future work.

References

Collin F. Baker, Charles J. Fillmore and John B. Lowe, 1998. *The Berkeley FrameNet Project.* ACL-COLING '98.

Daniel M. Bikel, 2004. *Intricacies of Collins' Parsing Model.* Computational Linguistics, 30(4):479–511.

Ted Briscoe, John Carroll, 1997. *Automatic Extraction of Subcategorization from Corpora.* Applied NLP 1997.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Pad and Manfred Pinkal, 2006 *The SALSA Corpus: a German Corpus Resource for Lexical Semantics.* LREC '06.

Lou Burnard, 2000. *User Reference Guide for the British National Corpus.* Technical report, Oxford University.

Xavier Carreras and Lluìs Màrquez, 2004. *Introduction to the CoNLL–2004 Shared Task: Semantic Role Labeling.* CoNLL '04.

Xavier Carreras and Lluìs Màrquez, 2005. *Introduction to the CoNLL–2005 Shared Task: Semantic Role Labeling.* CoNLL '05.

Alexander Clark, 2003. *Combining Distributional and Morphological Information for Part of Speech Induction.* EACL '03.

Ronan Collobert and Jason Weston, 2007. *Fast Semantic Extraction Using a Novel Neural Network Architecture.* ACL '07.

Mona Diab, Aous Mansouri, Martha Palmer, Olga Babko-Malaya, Wajdi Zaghouani, Ann Bies and Mohammed Maamouri, 2008. *A pilot Arabic PropBank.* LREC '08.

Evgeniy Gabrilovich and Shaul Markovitch, 2005. *Feature Generation for Text Categorization using World Knowledge.* IJCAI '05.

Daniel Gildea and Daniel Jurafsky, 2002. *Automatic Labeling of Semantic Roles.* Computational Linguistics, 28(3):245–288.

Elliot Glaysher and Dan Moldovan, 2006. *Speeding Up Full Syntactic Parsing by Leveraging Partial Parsing Decisions.* COLING/ACL '06 poster session.

Andrew Gordon and Reid Swanson, 2007. *Generalizing Semantic Role Annotations across Syntactically Similar Verbs.* ACL '07.

David Graff, 1995. *North American News Text Corpus.* Linguistic Data Consortium. LDC95T21.

Trond Grenager and Christopher D. Manning, 2006. *Unsupervised Discovery of a Statistical Verb Lexicon.* EMNLP '06.

Kadri Hacioglu, 2004. *Semantic Role Labeling using Dependency Trees.* COLING '04.

Kadri Hacioglu and Wayne Ward, 2003. *Target Word Detection and Semantic Role Chunking using Support Vector Machines.* HLT-NAACL '03.

Rohit J. Kate and Raymond J. Mooney, 2007. *Semi-Supervised Learning for Semantic Parsing using Support Vector Machines.* HLT–NAACL '07.

Karin Kipper, Hoa Trang Dang and Martha Palmer, 2000. *Class-Based Construction of a Verb Lexicon.* AAAI '00.

Dan Klein, 2005. *The Unsupervised Learning of Natural Language Structure.* Ph.D. thesis, Stanford University.

Anna Korhonen, 2002. *Subcategorization Acquisition.* Ph.D. thesis, University of Cambridge.

Christopher D. Manning, 1993. *Automatic Acquisition of a Large Subcategorization Dictionary.* ACL '93.

Lluìs Màrquez, Xavier Carreras, Kenneth C. Litkowski and Suzanne Stevenson, 2008. *Semantic Role Labeling: An introdution to the Special Issue.* Computational Linguistics, 34(2):145–159

Lluìs Màrquez, Jesus Gimènez Pere Comas and Neus Català, 2005. *Semantic Role Labeling as Sequential Tagging.* CoNLL '05.

Lluìs Màrquez, Lluis Villarejo, M. A. Martì and Mariona Taulè, 2007. *SemEval–2007 Task 09: Multilevel Semantic Annotation of Catalan and Spanish.* The 4th international workshop on Semantic Evaluations (SemEval '07).

Gabriele Musillo and Paula Merlo, 2006. *Accurate Parsing of the proposition bank.* HLT-NAACL '06.

Martha Palmer, Daniel Gildea and Paul Kingsbury, 2005. *The Proposition Bank: A Corpus Annotated with Semantic Roles.* Computational Linguistics, 31(1):71–106.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin and Daniel Jurafsky, 2005. *Support Vector Learning for Semantic Argument Classification.* Machine Learning, 60(1):11–39.

Sameer Pradhan, Wayne Ward, James H. Martin, 2008. *Towards Robust Semantic Role Labeling.* Computational Linguistics, 34(2):289–310.

Adwait Ratnaparkhi, 1996. *Maximum Entropy Part-Of-Speech Tagger.* EMNLP '96.

Helmut Schmid, 1994. *Probabilistic Part-of-Speech Tagging Using Decision Trees* International Conference on New Methods in Language Processing.

Yoav Seginer, 2007. *Fast Unsupervised Incremental Parsing.* ACL '07.

Noah A. Smith and Jason Eisner, 2006. *Annealing Structural Bias in Multilingual Weighted Grammar Induction.* ACL '06.

Robert S. Swier and Suzanne Stevenson, 2004. *Unsupervised Semantic Role Labeling.* EMNLP '04.

Robert S. Swier and Suzanne Stevenson, 2005. *Exploiting a Verb Lexicon in Automatic Semantic Role Labelling.* EMNLP '05.

Erik F. Tjong Kim Sang and Hervé Déjean, 2001. *Introduction to the CoNLL-2001 Shared Task: Clause Identification.* CoNLL '01.

Nianwen Xue and Martha Palmer, 2004. *Calibrating Features for Semantic Role Labeling.* EMNLP '04.

Nianwen Xue, 2008. *Labeling Chinese Predicates with Semantic Roles.* Computational Linguistics, 34(2):225–255.