Semi-supervised Semantic Role Labeling Using the Latent Words Language Model

Koen Deschacht
Department of computer science
K.U.Leuven, Belgium
koen.deschacht@cs.kuleuven.be

Marie-Francine Moens
Department of computer science
K.U.Leuven, Belgium
sien.moens@cs.kuleuven.be

Abstract

Semantic Role Labeling (SRL) has proved to be a valuable tool for performing automatic analysis of natural language texts. Currently however, most systems rely on a large training set, which is manually annotated, an effort that needs to be repeated whenever different languages or a different set of semantic roles is used in a certain application. A possible solution for this problem is semi-supervised learning, where a small set of training examples is automatically expanded using unlabeled texts. We present the Latent Words Language Model, which is a language model that learns word similarities from unlabeled texts. We use these similarities for different semi-supervised SRL methods as additional features or to automatically expand a small training set. We evaluate the methods on the PropBank dataset and find that for small training sizes our best performing system achieves an error reduction of 33.27% F1-measure compared to a state-of-the-art supervised baseline.

1 Introduction

Automatic analysis of natural language is still a very hard task to perform for a computer. Although some successful applications have been developed (see for instance (Chinchor, 1998)), implementing an automatic text analysis system is still a labour and time intensive task. Many applications would benefit from an intermediate representation of texts, where an automatic analysis is already performed which is sufficiently general to be useful in a wide range of applications.

Syntactic analysis of texts (such as Part-Of-Speech tagging and syntactic parsing) is an example of such a generic analysis, and has proved useful in applications ranging from machine translation (Marcu et al., 2006) to text mining in the bio-medical domain (Cohen and Hersh, 2005). A syntactic parse is however a representation that is very closely tied with the surface-form of natural language, in contrast to Semantic Role Labeling (SRL) which adds a layer of predicate-argument information that generalizes across different syntactic alternations (Palmer et al., 2005). SRL has received a lot of attention in the research community, and many systems have been developed (see section 2). Most of these systems rely on a large dataset for training that is manually annotated. In this paper we investigate whether we can develop a system that achieves state-of-the-art semantic role labeling without relying on a large number of labeled examples. We aim to do so by employing the Latent Words Language Model that learns *latent words* from a large unlabeled corpus. Latent words are words that (unlike observed words) did not occur at a particular position in a text, but given semantic and syntactic constraints from the context could have occurred at that particular position.

In section 2 we revise existing work on SRL and on semi-supervised learning. Section 3 outlines our supervised classifier for SRL and section 4 discusses the Latent Words Language Model. In section 5 we will combine the two models for semi-supervised role labeling. We will test the model on the standard PropBank dataset and compare it with state-of-the-art semi-supervised SRL systems in section 6 and finally in section 7 we draw conclusions and outline future work.

2 Related work

Gildea and Jurafsky (2002) were the first to describe a statistical system trained on the data from the FrameNet project to automatically assign semantic roles. This approach was soon followed by other researchers (Surdeanu et al., 2003; Pradhan et al., 2004; Xue and Palmer, 2004), focus-

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 21–29,
Singapore, 6-7 August 2009.

ing on improved sets of features, improved machine learning methods or both, and SRL became a shared task at the CoNLL 2004, 2005 and 2008 conferences[1]. The best system (Johansson and Nugues, 2008) in CoNLL 2008 achieved an F1-measure of 81.65% on the workshop's evaluation corpus.

Semi-supervised learning has been suggested by many researchers as a solution to the annotation bottleneck (see (Chapelle et al., 2006; Zhu, 2005) for an overview), and has been applied successfully on a number of natural language processing tasks. Mann and McCallum (2007) apply Expectation Regularization to Named Entity Recognition and Part-Of-Speech tagging, achieving improved performance when compared to supervised methods, especially on small numbers of training data. Koo et al. (2008) present an algorithm for dependency parsing that uses clusters of semantically related words, which were learned in an unsupervised manner. There has been little research on semi-supervised learning for SRL. We refer to He and Gildea (2006) who tested active learning and co-training methods, but found little or no gain from semi-supervised learning, and to Swier and Stevenson (2004), who achieved good results using semi-supervised methods, but tested their methods on a small number of VerbNet roles, which have not been used by other SRL systems. To the best of our knowledge no system was able to reproduce the successful results of (Swier and Stevenson, 2004) on the PropBank roleset. Our approach most closely resembles the work of Fürstenau and Lapata (2009) who automatically expand a small training set using an automatic dependency alignment of unlabeled sentences. This method was tested on the FrameNet corpus and improved results when compared to a fully-supervised classifier. We will discuss their method in detail in section 5.

3 Semantic role labeling

Fillmore (1968) introduced semantic structures called semantic frames, describing abstract actions or common situations (frames) with common roles and themes (semantic roles). Inspired by this idea different resources were constructed, including FrameNet (Baker et al., 1998) and PropBank (Palmer et al., 2005). An alternative approach to semantic role labeling is the framework developed by Halliday (1994) and implemented by Mehay et al. (2005). PropBank has thus far received the most attention of the research community, and is used in our work.

3.1 PropBank

The goal of the PropBank project is to add semantic information to the syntactic nodes in the English Penn Treebank. The main motivation for this annotation is the preservation of semantic roles across different syntactic realizations. Take for instance the sentences

1. The window broke.
2. John broke the window.

In both sentences the constituent "the window" is broken, although it occurs at different syntactic positions. The PropBank project defines for a large collection of verbs (excluding auxiliary verbs such as "will", "can", ...) a set of senses, that reflect the different meanings and syntactic alternations of this verb. Every sense has a number of expected roles, numbered from Arg0 to Arg5. A small number of arguments are shared among all senses of all verbs, such as temporals (Arg-TMP), locatives (Arg-LOC) and directionals (Arg-DIR). Additional to the frame definitions, PropBank has annotated a large training corpus containing approximately 113.000 annotated verbs. An example of an annotated sentence is

$$[\text{John}\ _{Arg0}][\text{broke}\ _{BREAK.01}]\ [\text{the window}\ _{Arg1}].$$

Here *BREAK*.01 is the first sense of the "break" verb. Note that (1) although roles are defined for every frame separately, in reality roles with identical names are identical or very similar for all frames, a fact that is exploited to train accurate role classifiers and (2) semantic role labeling systems typically assume that a frame is fully expressed in a single sentence and thus do not try to instantiate roles across sentence boundaries. Although the original PropBank corpus assigned semantic roles to syntactic phrases (such as noun phrases), we use the CoNLL dataset, where the PropBank corpus was converted to a dependency representation, assigning semantic roles to single (head) words.

3.2 Features

In this section we discuss the features used in the semantic role labeling system. All features but the

[1] See http://www.cnts.ua.ac.be/conll/ for an overview.

Split path feature are taken from existing semantic role labeling systems, see for example (Gildea and Jurafsky, 2002; Lim et al., 2004; Thompson et al., 2006). The number in brackets denotes the number of unique features for that type.

Word We split every sentence in (unigram) word tokens, including punctuation. (37079)

Stem We reduce the word tokens to their stem, e.g. "walks" -> "walk". (28690)

POS The part-of-speech tag for every word, e.g. "NNP" (for a singular proper noun). (77)

Neighbor POS's The concatenated part-of-speech tags of the word before and the word just after the current word, e.g. "RBS_JJR". (1787)

Path This important feature describes the path through the dependency tree from the current word to the position of the predicate, e.g. "coord↑obj↑adv↑root↓dep↓nmod↓pmod", where '↑' indicates going up a constituent and '↓' going down one constituent. (829642)

Split Path Because of the nature of the path feature, an explosion of unique features is found in a given data set. We reduce this by splitting the path in different parts and using every part as a distinct feature. We split, for example, the previous path in 6 different features: "coord", "↑obj", "↑adv", "↑root", "↓dep", "↓nmod", "↓pmod". Note that the split path feature includes the POS feature, since the first component of the path is the POS tag for the current word. This feature has not been used previously for semantic role detection. (155)

For every word w_i in the training and test set we construct the feature vector $\mathbf{f}(w_i)$, where at every position in this vector 1 indicates the presence for the corresponding feature and 0 the absence of that feature.

3.3 Discriminative model

Discriminative models have been found to outperform generative models for many different tasks including SRL (Lim et al., 2004). For this reason we also employ discriminative models here. The structure of the model was inspired by a similar (although generative) model in (Thompson et al., 2006) where it was used for semantic frame classification. The model (fig. 1) assumes that the role label r_{ij} for the word w_i is conditioned on the features $\mathbf{f}_i$ and on the role label $r_{i-1\,j}$ of the previous word and that the predicate label p_j for word w_j is conditioned on the role labels $\mathbf{R}^j$ and on the features $\mathbf{f}_j$. This model can be seen as an extension of the standard Maximum Entropy Markov Model (MEMM, see (Ratnaparkhi, 1996)) with an extra dependency on the predicate label, we will henceforth refer to this model as *MEMM+pred.*

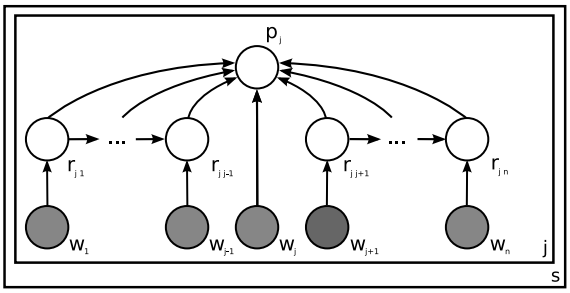

Figure 1: Discriminative model for SRL. Grey circles represent observed variables, white circles hidden variables and arrows directed dependencies. *s* ranges over all sentences in the corpus and *j* over the *n* words in the sentence.

To estimate the parameters of the *MEMM+pred* model we turn to the successful Maximum Entropy (Berger et al., 1996) parameter estimation method. The Maximum Entropy principle states that the best model given the training data is the model such that the conditional distribution defined by the model has maximum entropy subject to the constraints represented by the training examples. There is no closed form solution to find this maximum and we thus turn to an iterative method. In this work we use Generalized Iterative Scaling[2], but other methods such as (quasi-) Newton optimization could also have been used.

4 Latent Words Language Model

4.1 Rationale

As discussed in sections 1 and 3 most SRL systems are trained today on a large set of manually annotated examples. PropBank for example contains approximately 50000 sentences. This manual annotation is both time and labour-intensive, and needs to be repeated for new languages or

[2]We use the *maxent* package available on http://maxent.sourceforge.net/

for new domains requiring a different set of roles. One approach that can help to solve this problem is semi-supervised learning, where a small set of annotated examples is used together with a large set of unlabeled examples when training a SRL model.

Manual inspection of the results of the supervised model discussed in the previous section showed that the main source of errors was incorrect labeling of a word because the word token did not occur, or occurred only a small number of times in the training set. We hypothesize that knowledge of semantic similar words could overcome this problem by associating words that occurred infrequently in the training set to similar words that occurred more frequently. Furthermore, we would like to learn these similarities automatically, to be independent of knowledge sources that might not be available for all languages or domains.

The Distributional Hypothesis, supported by theoretical linguists such as Harris (1954), states that words that occur in the same contexts tend to have similar meanings. This suggests that one can learn the similarity between two words automatically by comparing their relative contexts in a large unlabeled corpus, which was confirmed by different researchers (e.g. (Lin, 1998; McDonald and Ramscar, 2001; Grefenstette, 1994)). Different methods for computing word similarities have been proposed, differing between methods to represent the context (using dependency relationship or a window of words) and between methods that, given a set of contexts, compute the similarity between different words (ranging from cosine similarity to more complex metrics such as the Jaccard index). We refer to (Lin, 1998) for a comparison of the different similarity metrics.

In the next section we propose a novel method to learn word similarities, the Latent Words Language Model (LWLM) (Deschacht and Moens, 2009). This model learns similar words and learns the a distribution over the contexts in which certain types of words occur typically.

4.2 Definition

The LWLM introduces for a text $\mathbf{T} = w_1 ... w_N$ of length N for every observed word w_i at position i a hidden variable h_i. The model is a generative model for natural language, in which the latent variable h_i is generated by its context $C(h_i)$ and the observed word w_i is generated by the latent variable h_i. In the current model we assume that the context is $C(h_i) = \mathbf{h}_{i-2}^{i-1}\mathbf{h}_{i+1}^{i+2}$ where $\mathbf{h}_{i-2}^{i-1} = h_{i-2}h_{i-1}$ is the two previous words and $\mathbf{h}_{i+1}^{i+2} = h_{i+1}h_{i+2}$ is the two next words. The observed w_i has a value from the vocabulary V, while the hidden variable h_i is unknown, and is modeled as a probability distribution over all words of V. We will see in the next section how this distribution is estimated from a large unlabeled training corpus. The aim of this model is to estimate, at every position i, a distribution for h_i, assigning high probabilities to words that are similar to w_i, given the context of this word $C(h_i)$, and low probabilities to words that are not similar to w_i in this context.

A possible interpretation of this model states that every hidden variable h_i models the "meaning" for a particular word in a particular context. In this probabilistic model, when generating a sentence, we generate the meaning of a word (which is an unobserved representation) with a certain probability, and then we generate a certain observation by writing down one of the possible words that express this meaning.

Creating a representation that models the meaning of a word is an interesting (and controversial) topic in its own right, but in this work we make the assumption that the meaning of a particular word can be modeled using other words. Modeling the meaning of a word with other words is not an unreasonable one, since it is already employed in practice by humans (e.g. by using dictionaries and thesauri) and machines (e.g. relying on a lexical resource such as WordNet) in word sense disambiguation tasks.

4.3 Parameter estimation

As we will further see the LWLM model has three probability distributions: $P(w_i|h_i)$, the probability of the observed word w_j given the latent variable h_j, $P(h_i|\mathbf{h}_{i-2}^{i-1})$, the probability of the hidden word h_j given the previous variables h_{j-2} and h_{j-1}, and $P(h_i|\mathbf{h}_{i+1}^{i+2})$, the probability of the hidden word h_j given the next variables h_{j+1} and h_{j+2}. These distributions need to be learned from a training text $\mathbf{T}_{train} =< w_0 ... w_z >$ of length Z.

4.3.1 The Baum-Welch algorithm

The attentive reader will have noticed the similarity between the proposed model and a standard second-order Hidden Markov Model (HMM) where the hidden state is dependent on the two

previous states. However, we are not able to use the standard Baum-Welch (or forward-backward) algorithm, because the hidden variable h_i is modeled as a probability distribution over all words in the vocabulary V. The Baum-Welch algorithm would result in an execution time of $O(|V|^3 NG)$ where $|V|$ is the size of the vocabulary, N is the length of the training text and G is the number of iterations needed to converge. Since in our dataset the vocabulary size is more than $30K$ words (see section 3.2), using this algorithm is not possible. Instead we use techniques of approximate inference, i.e. Gibbs sampling.

4.3.2 Initialization

Gibbs sampling starts from a random initialization for the hidden variables and then improves the estimates in subsequent iterations. In preliminary experiments it was found that a pure random initialization results in a very long burn-in-period and a poor performance of the final model. For this reason we initially set the distributions for the hidden words equal to the distribution of words as given by a standard language model[3].

4.3.3 Gibbs sampling

We store the initial estimate of the hidden variables in $\mathbf{M}^0_{train} =< h_0 ... h_Z >$, where h_i generates w_i at every position i. Gibbs sampling is a Markov Chain Monte Carlo method that updates the estimates of the hidden variables in a number of iterations. $\mathbf{M}^{\tau}_{train}$ denotes the estimate of the hidden variables in iteration τ. In every iteration a new estimate $\mathbf{M}^{\tau+1}_{train}$ is generated from the previous estimate $\mathbf{M}^{\tau}_{train}$ by selecting a random position j and updating the value of the hidden variable at that position. The probability distributions $P^{\tau}(w_j|h_j)$, $P^{\tau}(h_j|\mathbf{h}^{j-1}_{j-2})$ and $P^{\tau}(h_j|\mathbf{h}^{j+2}_{j+1})$ are constructed by collecting the counts from all positions $i \neq j$. The hidden variable h_j is dependent on h_{j-2}, h_{j-1}, h_{j+1}, h_{j+2} and w_j and we can compute the distribution of possible values for the variable h_j as

$$P^{\tau}(h_j|w_j, \mathbf{h}^{j-1}_0, \mathbf{h}^{Z}_{j+1}) = \frac{P^{\tau}(w_j|h_j)P^{\tau}(h_j|\mathbf{h}^{j-1}_{j-2}\mathbf{h}^{j+2}_{j+1})}{\sum_{h_i} P^{\tau}(w_i|h_i)P^{\tau}(h_j|\mathbf{h}^{j-1}_{j-2}\mathbf{h}^{j+2}_{j+1})}$$

We set $P(h_j|\mathbf{h}^{j-1}_{j-2}\mathbf{h}^{j+2}_{j+1}) = P(h_j|\mathbf{h}^{j-1}_{j-2}) \cdot P(h_j|\mathbf{h}^{j+2}_{j+1})$ which can be easily computed given the above distributions. We select a new value for the hidden variable according to $P^{\tau}(h_j|w_j, \mathbf{h}^{j-1}_0, \mathbf{h}^{Z}_{j+1})$ and place it at position j in $\mathbf{M}^{\tau+1}_{train}$. The current estimate for all other unobserved words remains the same. After performing this iteration a large number of times ($|V| * 10$ in this experiment), the distribution approaches the true maximum likelihood distribution. Gibbs sampling however samples this distribution, and thus will never reach it exactly. A number of iterations ($|V| * 100$) is then performed in which Gibbs sampling oscillates around the correct distribution. We collect independent samples of this distribution every $|V| * 10$ iterations, which are then used to construct the final model.

[3] We used the interpolated Kneser-Ney model as described in (Goodman, 2001).

4.4 Evaluation of the Language Model

A first evaluation of the quality of the automatically learned latent words is by translation of this model into a sequential language model and by measuring its perplexity on previously unseen texts. In (Deschacht and Moens, 2009) we perform a number of experiments, comparing different corpora (news texts from Reuters and from Associated Press, and articles from Wikipedia) and n-gram sizes (3-gram and 4-gram). We also compared the proposed model with two state-of-the-art language models, Interpolated Kneser-Ney smoothing and *fullibmpredict* (Goodman, 2001), and found that LWLM outperformed both models on all corpora, with a perplexity reduction ranging between 12.40% and 5.87%. These results show that the estimated distributions over latent words are of a high quality and lead us to believe they could be used to improve automatic text analysis, like SRL.

5 Role labeling using latent words

The previous section discussed how the LWLM learns similar words and how these similarities improved the perplexity on an unseen text of the language model derived from this model. In this section we will see how we integrate the latent words model in two novel semi-supervised SRL models and compare these with two state-of-the-art semi-supervised models for SRL and dependency parsing.

Latent words as additional features

In a first approach we estimate the distribution of latent words for every word for both the training and test set. We then use the latent words at every

position as additional probabilistic features for the discriminative model. More specifically, we append $|V|$ extra values to the feature vector $\mathbf{f}(w_j)$, containing the probability distribution over the $|V|$ possible words for the hidden variable h_i[4]. We call this the *LWFeatures* method.

This method has the advantage that it is simple to implement and that many existing SRL systems can be easily extended by adding additional features. We also expect that this method can be employed almost effortless in other information extraction tasks, such as Named Entity Recognition or Part-Of-Speech labeling.

We compare this approach to the semi-supervised method in Koo et al. (2008) who employ clusters of related words constructed by the Brown clustering algorithm (Brown et al., 1992) for syntactic processing of texts. Interestingly, this clustering algorithm has a similar objective as LWLM since it tries to optimize a class-based language model in terms of perplexity on an unseen test text. We employ a slightly different clustering method here, the *fullibmpredict* method discussed in (Goodman, 2001). This method was shown to outperform the class based model proposed in (Brown et al., 1992) and can thus be expected to discover better clusters of words. We append the feature vector $\mathbf{f}(w_j)$ with c extra values (where c is the number of clusters), respectively set to 1 if the word w_i belongs to the corresponding cluster or to 0 otherwise. We call this method the *ClusterFeatures* method.

Automatic expansion of the training set using predicate argument alignment

We compare our approach with a method proposed by Fürstenau and Lapata (2009). This approach is more tailored to the specific case of SRL and is summarized here.

Given a set of labeled seed verbs with annotated semantic roles, for every annotated verb a number of occurrences of this verb is found in unlabeled texts where the context is similar to the context of the annotated example. The context is defined here as all words in the sentence that are direct dependents of this verb, given the syntactic dependency tree. The similarity between two occurrences of a particular verb is measured by finding all different alignments $\sigma : M_\sigma \rightarrow \{1...n\}$ $(M_\sigma \subset \{1,...,m\})$ between the m dependents of the first occurrence and the n dependents of the second occurrence. Every alignment σ is assigned a score given by

$$\sum_{i \in M_\sigma} \left(A \cdot syn(g_i, g_{\sigma(i)}) + sem(w_i, w_{\sigma(i)}) - B \right)$$

where $syn(g_i, g_{\sigma(i)})$ denotes the syntactic similarity between grammatical role[5] g_i of word w_i and grammatical role $g_{\sigma(i)}$ of word $w_{\sigma(i)}$, and $sem(w_i, w_{\sigma(i)})$ measures the semantic similarity between words w_i and $w_{\sigma(i)}$. A is a constant weighting the importance of the syntactic similarity compared to semantic similarity, and B can be interpreted as the lowest similarity value for which an alignment between two arguments is possible. The syntactic similarity $syn(g_i, g_{\sigma(i)})$ is defined as 1 if the dependency relations are identical, $0 < a < 1$ if the relations are of the same type but of a different subtype[6] and 0 otherwise. The semantic similarity $sem(w_i, w_{\sigma(i)})$ is automatically estimated as the cosine similarity between the contexts of w_i and $w_{\sigma(i)}$ in a large text corpus. For details we refer to (Fürstenau and Lapata, 2009).

For every verb in the annotated training set we find the k occurrences of that verb in the unlabeled texts where the contexts are most similar given the best alignment. We then expand the training set with these examples, automatically generating an annotation using the discovered alignments. The variable k controls the trade-off between annotation confidence and expansion size. The final model is then learned by running the supervised training method on the expanded training set. We call this method *AutomaticExpansionCOS*[7]. The values for k, a, A and B are optimized automatically in every experiment on a held-out set (disjoint from both training and test set).
We adapt this approach by employing a different method for measuring semantic similarity. Given two words w_i and $w_{\sigma(i)}$ we estimate the distribution of latent words, respectively $L(h_i)$ and

[4] Probabilities smaller than $1e10^{-4}$ were set to 0 for efficiency reasons.

[5] Note that this is a syntactic role, not a semantic role as the ones discussed in this article.

[6] Subtypes are fine-grained distinctions made by the parser such as the underlying grammatical roles in passive constructions.

[7] The only major differences with (Fürstenau and Lapata, 2009) are the dependency parser which was used (the MALT parser (Nivre et al., 2006) instead of the RASP parser (Briscoe et al., 2006)) and the corpus employed to learn semantic similarities (the Reuters corpus instead of the British National Corpus). We expect that these differences will only influence the results minimally.

	5%	20%	50%	100%
Supervised	40.49%	67.23%	74.93%	78.65%
LWFeatures	**60.29%**	**72.88%**	**76.42%**	**80.98%**
ClusterFeatures	59.51%	66.70%	70.15%	72.62%
AutomaticExpansionCOS	47.05%	53.72%	64.51%	70.52%
AutomaticExpansionLW	45.40%	53.82%	65.39%	72.66%

Table 1: Results (in F1-measure) on the CoNLL 2008 test set for the different methods, comparing the supervised method (*Supervised*) with the semi-supervised methods *LWFeatures*, *ClusterFeatures*, *AutomaticExpansionCOS* and *AutomaticExpansionLW*. See section 5 for details on the different methods. Best results are in bold.

$L(h_{\sigma(i)})$. We then compute the semantic similarity measure as the Jensen-Shannon (Lin, 1997) divergence

$$JS(L(h_i)||L(h_{\sigma(i)})) = \frac{1}{2}\left[D(L(h_i)||avg) + D\left(L(h_{\sigma(i)})||avg\right)\right]$$

where $avg = (L(h_i) + L(h_{\sigma(i)}))/2$ is the average between the two distributions and $D(L(h_i)||avg)$ is the Kullback–Leiber divergence (Cover and Thomas, 2006).

Although this change might appear only a slight deviation from the original model discussed in (Fürstenau and Lapata, 2009) it is potentially an important one, since an accurate semantic similarity measure will greatly influence the accuracy of the alignments, and thus of the accuracy of the automatic expansion. We call this method *AutomaticExpansionLW*.

6 Experiments

We perform a number of experiments where we compare the fully supervised model with the semi-supervised models proposed in the previous section. We first train the LWLM model on an unlabeled 5 million word *Reuters* corpus[8].

We perform different experiments for the supervised and the four different semi-supervised methods (see previous section). Table 1 shows the results of the different methods on the test set of the CoNLL 2008 shared task. We experimented with different sizes for the training set, ranging from 5% to 100%. When using a subset of the full training set, we run 10 different experiments with random subsets and average the results.

We see that the *LWFeatures* method performs better than the other methods across all training sizes. Furthermore, these improvements are larger for smaller training sets, showing that the approach can be applied successfully in a setting where only a small number of training examples is available.

When comparing the *LWFeatures* method with the *ClusterFeatures* method we see that, although the *ClusterFeatures* method has a similar performance for small training sizes, this performance drops for larger training sizes. A possible explanation for this result is the use of the clusters employed in the *ClusterFeatures* method. By definition the clusters merge many words into one cluster, which might lead to good generalization (more important for small training sizes) but can potentially hurt precision (more important for larger training sizes).

A third observation that can be made from table 1 is that, although both automatic expansion methods (*AutomaticExpansionCOS* and *AutomaticExpansionCOS*) outperform the supervised method for the smallest training size, for other sizes of the training set they perform relatively poorly. An informal inspection showed that for some examples in the training set, little or no correct similar occurrences were found in the unlabeled text. The algorithm described in section 5 adds the most similar k occurrences to the training set for every annotated example, also for these examples where little or no similar occurrences were found. Often the automatic alignment fails to generate correct labels for these occurrences and introduces errors in the training set. In the future we would like to perform experiments that determine dynamically (for instance based on the similarity measure between occurrences) for every annotated example how many training examples to add.

[8] See http://www.daviddlewis.com/resources

7 Conclusions and future work

We have presented the Latent Words Language Model and showed how it learns, from unlabeled texts, latent words that capture the meaning of a certain word, depending on the context. We then experimented with different methods to incorporate the latent words for Semantic Role Labeling, and tested different methods on the PropBank dataset. Our best performing method showed a significant improvement over the supervised model and over methods previously proposed in the literature. On the full training set the best method performed 2.33% better than the fully supervised model, which is a 10.91% error reduction. Using only 5% of the training data the best semi-supervised model still achieved 60.29%, compared to 40.49% by the supervised model, which is an error reduction of 33.27%. These results demonstrate that the latent words learned by the LWLM help for this complex information extraction task. Furthermore we have shown that the latent words are simple to incorporate in an existing classifier by adding additional features. We would like to perform experiments on employing this model in other information extraction tasks, such as Word Sense Disambiguation or Named Entity Recognition. The current model uses the context in a very straightforward way, i.e. the two words left and right of the current word, but in the future we would like to explore more advanced methods to improve the similarity estimates. Lin (1998) for example discusses a method where a syntactic parse of the text is performed and the context of a word is modeled using dependency triples.

The other semi-supervised methods proposed here were less successful, although all improved on the supervised model for small training sizes. In the future we would like to improve the described automatic expansion methods, since we feel that their full potential has not yet been reached. More specifically we plan to experiment with more advanced methods to decide whether some automatically generated examples should be added to the training set.

Acknowledgments

The work reported in this paper was supported by the EU-IST project CLASS (Cognitive-Level Annotation using Latent Statistical Structure, IST-027978) and the IWT-SBO project AMASS++ (IWT-SBO-060051). We thank the anonymous reviewers for their helpful comments and Dennis N. Mehay for his help on clarifying the linguistic motivation of our models.

References

C.F. Baker, C.J. Fillmore, and J.B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, volume 98. Montreal, Canada.

A.L. Berger, V.J. Della Pietra, and S.A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

T. Briscoe, J. Carroll, and R. Watson. 2006. The second release of the RASP system. In *Proceedings of the Interactive Demo Session of COLING/ACL*, volume 6.

P.F. Brown, R.L. Mercer, V.J. Della Pietra, and J.C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

O. Chapelle, B. Schölkopf, and A. Zien, editors. 2006. *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

N.A. Chinchor. 1998. Overview of MUC-7/MET-2. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, volume 1.

A.M. Cohen and W.R. Hersh. 2005. A survey of current work in biomedical text mining. *Briefings in Bioinformatics*, 6(1):57–71.

T.M. Cover and J.A. Thomas. 2006. *Elements of Information Theory*. Wiley-Interscience.

Koen Deschacht and Marie-Francine Moens. 2009. The Latent Words Language Model. In *Proceedings of the 18th Annual Belgian-Dutch Conference on Machine Learning*.

C. J. Fillmore. 1968. The case for case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*. Rinehart & Winston.

Hagen Fürstenau and Mirella Lapata. 2009. Semi-supervised semantic role labeling. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 220–228, Athens, Greece. Association for Computational Linguistics.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Joshua T. Goodman. 2001. A bit of progress in language modeling, extended version. Technical report, Microsoft Research.

G. Grefenstette. 1994. *Explorations in automatic thesaurus discovery*. Springer.

M.A.K. Halliday. 1994. *An Introduction to Functional Grammar (second edition)*. Edward Arnold, London.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

S. He and D. Gildea. 2006. Self-training and Co-training for Semantic Role Labeling: Primary Report. Technical report. TR 891.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with propbank and nombank. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 183–187, Manchester, England, August. Coling 2008 Organizing Committee.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 595–603.

J.-H. Lim, Y.-S. Hwang, S.-Y. Park, and H.-C. Rim. 2004. Semantic role labeling using maximum entropy model. In *Proceedings of the Eighth Conference on Computational Natural Language Learning*, pages 122–125, Boston, Massachusetts, USA. ACL.

D. Lin. 1997. Using syntactic dependency as local context to resolve word sense ambiguity. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, volume 35, pages 64–71. ACL.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational Linguistics*, pages 768–774. Association for Computational Linguistics Morristown, NJ, USA.

G.S. Mann and A. McCallum. 2007. Simple, robust, scalable semi-supervised learning via expectation regularization. In *Proceedings of the 24th International Conference on Machine Learning*, pages 593–600. ACM Press New York, USA.

D. Marcu, W. Wang, A. Echihabi, and K. Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing*, pages 44–52.

S. McDonald and M. Ramscar. 2001. Testing the distributional hypothesis: The influence of context on judgements of semantic similarity. In *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, pages 611–616.

Dennis Mehay, Rik De Busser, and Marie-Francine Moens. 2005. Labeling generic semantic roles. In *Proceedings of the Sixth International Workshop on Computational Semantics*.

J. Nivre, J. Hall, and J. Nilsson. 2006. MaltParser: A datadriven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 2216–2219.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

S. Pradhan, W. Ward, K. Hacioglu, J. Martin, and D. Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of the Human Language Technology Conference/North American chapter of the Association of Computational Linguistics*, Boston, MA.

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142. Association for Computational Linguistics.

M. Surdeanu, S. Harabagiu, J. Williams, and P. Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 8–15.

R.S. Swier and S. Stevenson. 2004. Unsupervised semantic role labelling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 95–102.

C. Thompson, R. Levy, and C. Manning. 2006. A generative model for FrameNet semantic role labeling . In *Proceedings of the 14th European Conference on Machine Learning*, Cavtat-Dubrovnik, Croatia.

N. Xue and M. Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 4.

X. Zhu. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.