Collective Semantic Role Labelling with Markov Logic

Sebastian Riedel **Ivan Meza-Ruiz**
Institute for Communicating and Collaborative Systems
School of Informatics
University of Edinburgh, Scotland
{S.R.Riedel,I.V.Meza-Ruiz}@sms.ed.ac.uk

Abstract

This paper presents our system for the Open Track of the CoNLL 2008 Shared Task (Surdeanu et al., 2008) in Joint Dependency Parsing[1] and Semantic Role Labelling. We use Markov Logic to define a joint SRL model and achieve a semantic F-score of 74.59%, the second best in the Open Track.

1 Introduction

Many SRL systems use a two-stage pipeline that first extracts possible argument candidates (argument identification) and then assigns argument labels to these candidates (argument classification) (Xue and Palmer, 2004). If we also consider the necessary previous step of identifying the predicates and their senses (predicate identification) this yields a three-stage pipeline: predicate identification, argument identification and argument classification.

Our system, on the other hand, follows a joint approach in the spirit of Toutanova et al. (2005) and performs the above steps *collectively* . We decided to use Markov Logic (ML, Richardson and Domingos, 2005), a First Order Probabilistic Language, to develop a global probabilistic model of SRL. By using ML we are able to incorporate the dependencies between the decisions of different stages in the pipeline and the well-known global correlations between the arguments of a predicate (Punyakanok et al., 2005). And since learning and inference methods were already implemented in the ML software we use, only minimal engineering efforts had to be done.

In contrast to the work of Toutanova et al. (2005) our system applies online learning to train its parameters and exact inference to predict a collective role labelling. Moreover, we jointly label the arguments of *all* predicates in a sentence. This allows us, for example, to require that certain tokens have to be an argument of some predicates in the sentence.

In this paper we also investigate the impact of different levels of interaction between the layers of the joint SRL model. We find that a probabilistic model which resembles a traditional bottom-up pipeline (though jointly trained and globally normalised) performs better than the complete joint model on the WSJ test set and worse on the Brown test set. The worst performance is observed when no interaction between SRL stages is allowed.

In terms of semantic F-score (74.59%) our submitted results are the second best in the Open Track of the Shared Task. Our error analysis indicates that a) the training regime can be improved and b) nominalizations are difficult to handle for the model as it is.

In the next sections we will first briefly introduce Markov Logic. Then we present the Markov Logic model we used in our final submission. We present and analyse our results in section 4 before we conclude in Section 5.

2 Markov Logic

Markov Logic (ML, Richardson and Domingos, 2005) is a Statistical Relational Learning language based on First Order Logic and Markov Networks. It can be seen as a formalism that extends First Order Logic to allow formulae that can be violated with some penalty. From an alternative

[1] Note that in this work we do not consider the parsing task; instead we use the provided dependencies of the open track datatsets.

CoNLL 2008: Proceedings of the 12th Conference on Computational Natural Language Learning, pages 193–197
Manchester, August 2008

point of view, it is an expressive template language that uses First Order Logic formulae to instantiate Markov Networks of repetitive structure.

Let us describe Markov Logic by considering the predicate identification task. In Markov Logic we can model this task by first introducing a set of logical predicates[2] such as *isPredicate(Token)* or *word(Token,Word)*. Then we specify a set of weighted first order formulae that define a distribution over sets of ground atoms of these predicates (or so-called *possible worlds*).

Ideally, the distribution we define with these weighted formulae assigns high probability to possible worlds where SRL predicates are correctly identified and a low probability to worlds where this is not the case. For example, a suitable set of weighted formulae would assign a high probability to the world[3]

$$\{word\,(1, Haag)\,, word(2, plays),\\ word(3, Elianti), isPredicate(2)\}$$

and a low one to

$$\{word\,(1, Haag)\,, word(2, plays),\\ word(3, Elianti), isPredicate(3)\}$$

In Markov Logic a set $M = \{(\phi, w_\phi)\}_\phi$ of weighted first order formulae is called a *Markov Logic Network* (MLN). It assigns the probability

$$p\,(\mathbf{y}) = \frac{1}{Z} \exp\left(\sum_{(\phi,w)\in M} w \sum_{\mathbf{c}\in C^{n_\phi}} f^\phi_{\mathbf{c}}\,(\mathbf{y})\right) \quad (1)$$

to the possible world $\mathbf{y}$. Here $f^\phi_{\mathbf{c}}$ is a feature function that returns 1 if in the possible world $\mathbf{y}$ the ground formula we get by replacing the free variables in ϕ by the constants in $\mathbf{c}$ is true and 0 otherwise. C^{n_ϕ} is the set of all tuples of constants we can replace the free variables in ϕ with. Z is a normalisation constant. Note that this distribution corresponds to a Markov Network where nodes represent ground atoms and factors represent ground formulae.

For example, if M contains the formula ϕ

$$word\left(x,'take'\right) \Rightarrow isPredicate\,(x)$$

then its corresponding log-linear model has, among others, a feature f^ϕ_{t1} for which x in ϕ has been replaced by the constant t_1 and that returns 1 if

$$word\left(1,'take'\right) \Rightarrow isPredicate\,(1)$$

is true in $\mathbf{y}$ and 0 otherwise.

We will refer predicates such as *word* as *observed* because they are known in advance. In contrast, *isPredicate* is *hidden* because we need to infer it at test time.

2.1 Learning

An MLN we use to model the collective SRL task is presented in section 3. We learn the weights associated this MLN using 1-best MIRA (Crammer and Singer, 2003) Online Learning method.

2.2 Inference

Assuming that we have an MLN, a set of weights and a given sentence then we need to predict the choice of predicates, frame types, arguments and role labels with maximal a posteriori probability. To this end we apply a method that is both exact and efficient: Cutting Plane Inference (CPI, Riedel, 2008) with Integer Linear Programming (ILP) as base solver. We use it for inference at test time as well as during the MIRA online learning process.

3 Model

We define five hidden predicates for the three stages of the task. For predicate identification, we use the predicates *isPredicate* and *sense*. *isPredicate(p)* indicates that the word in the position p is an SRL predicate while *sense(p,e)* signals that predicate in position p has the sense e.

For argument identification, we use the predicates *isArgument* and *hasRole*. The atom *isArgument(a)* signals that the word in the position a is a SRL argument of some (unspecified) SRL predicate while *hasRole(p,a)* indicates that the token at position a is an argument of the predicate in position p.

Finally, for the argument classification stage we define the predicate *role*. Here *role(p,a,r)* corresponds to the decision that the argument in the position a has the role r with respect to the predicate in the position p.

3.1 Local formulae

We define a set of *local* formulae. A formula is local if its groundings relate any number of observed ground atoms to exactly one hidden ground atom. For example, a grounding of the local formula

$$lemma(p, +l_1) \wedge lemma(a, +l_2) \Rightarrow hasRole(p, a)$$

[2] In the cases were is not obvious whether we refer to SRL or ML predicates we add the prefix SRL or ML, respectively.

[3] "Haag plays Elianti" is a segment of a sentence in training corpus.

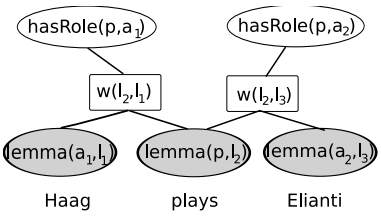

Figure 1: Factor graph for the local formula in section 3.1.

can be seen in the Markov Network of Figure 1. It connects a hidden *hasRole* ground atom to two observed *lemma* ground atoms. Note that the "+" prefix for variables indicates that there is a different weight for each possible pair of lemmas (l_1, l_2).

For the *hasRole* and *role* predicates we defined local formulae that aimed to reproduce the standard features used in previous work (Xue and Palmer, 2004). This also required us to develop dependency-based versions of the constituent-based features such as the syntactic path between predicate and argument, as proposed by Xue and Palmer (2004).

The remaining hidden predicates, *isPredicate*, *isArgument* and *sense*, have local formulae that relate their ground atoms to properties of a contextual window around the token the atom corresponds to. For this we used the information provided in the closed track training corpus of the shared task (i.e. both versions of lemma and POS tags plus a coarse version of the POS tags).

Instead of describing the local feature set in more detail we refer the reader to our MLN model files.[4] They can be used both as a reference and as input to our Markov Logic Engine[5], and thus allow the reader to easily reproduce our results. We believe that this is another advantage of explicitly separating model and algorithms by using first order probabilistic logic languages.

3.2 Global formulae

Global formulae relate several hidden ground atoms. We use them for two purposes: to ensure consistency between the decisions of all SRL stages and to capture some of our intuition about the task. We will refer to formulae that serve the first purpose as *structural constraints*.

For example, a structural constraint is given by the (deterministic) formula

$$role(p, a, r) \Rightarrow hasRole(p, a)$$

which ensures that, whenever the argument a is given a label r with respect to the predicate p, this argument must be an argument of a as denoted by *hasRole(p,a)*. Note that this formula by itself models the traditional "bottom-up" argument identification and classification pipeline: it is possible to not assign a role r to an predicate-argument pair (p, a) proposed by the identification stage; however, it is impossible to assign a role r to token pairs (p, a) that have not been proposed as potential arguments.

One example of another class of structural constraints is

$$hasRole(p, a) \Rightarrow \exists r.role(p, a, r)$$

which, by itself, models an inverted or "top-down" pipeline. In this architecture the argument classification stage can assign roles to tokens that have not been proposed by the argument identification stage. However, it must assign a label to any token pair the previous stage proposes. Figure 2 illustrates the structural formulae we use in form of a Markov Network.

The formulae we use to ensure consistency between the remaining hidden predicates are omitted for brevity as they are very similar to the bottom-up and top-down formulae we presented above.

For the SRL predicates that perform a labelling task (*role* and *sense*) we also need a structural constraint which ensures that not more than one label is assigned. For instance,

$$(role(p, a, r_1) \wedge r_1 \neq r_2 \Rightarrow \neg role(p, a, r_2))$$

forbids two different semantic roles for a pair of words.

The global formulae that capture our intuition about the task itself can be further divided into two classes. The first one uses deterministic or *hard* constraints such as

$$role\,(p, a_1, r) \wedge \neg mod\,(r) \wedge a_1 \neq a_2 \Rightarrow \neg role\,(p, a_2, r)$$

which forbids cases where distinct arguments of a predicate have the same role unless the role describes a modifier.

The second class of global formulae is *soft* or nondeterministic. For instance, the formula

$$lemma(p, +l) \wedge ppos(a, +p) \wedge hasRole(p, a) \Rightarrow sense(p, +f)$$

[4] http://thebeast.googlecode.com/svn/mlns/conll08

[5] http://thebeast.googlecode.com

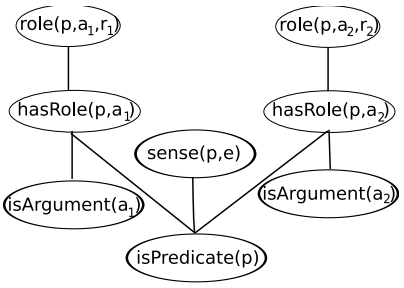

Figure 2: Markov Network that illustrates the structural constraints we use.

Model	WSJ	Brown	Train Time	Test Time
Full	75.72%	**65.38%**	25h	24m
Up	**76.96%**	63.86%	11h	14m
Down	73.48%	59.34%	22h	23m
Isolated	60.49%	48.12%	11h	14m
Structural	74.93%	64.23%	22h	33m

Table 1: F-scores for different models.

is a soft global formula. It captures the observation that the sense of a verb or noun depends on the type of its arguments. Here the type of an argument token is represented by its POS tag.

4 Results

We only submitted results for the Open Track of the Shared Task. Moreover, we focused on SRL and did not infer dependencies; instead we used the MALT dependencies parses provided in the Open Track dataset. Our submission was ranked second out of five with a semantic F1-score of 74.59%.[6]

After submission we also set up additional experiments to evaluate different types and degrees of connectivity between the decisions made by our model. To this end we created four new models: a model that omits top-down structural constraints and thus resembles a (globally trained) bottom-up pipeline (Up); a model that does not contain bottom-up structural constraints and thus resembles a top-down architecture (Down); a model in which stages are not connected at all (Isolated); and finally, a model in which additional global formulae are omitted and the only remaining global formulae are structural (Structural). The results we submitted were generated using the full model (Full).

Table 1 summarises the results for each of these models. We report the F-scores for the WSJ and Brown test corpora provided for the task. In addition we show training and test times for each system.

There are four findings we take from this. First, and somewhat surprisingly, the jointly trained bottom-up model (Up) performs substantially better than the full model on the WSJ test corpus. We will try to give an explanation for this result in the next section. Second, the bottom-up model is twice as fast compared to both the full and the top-down model. This is due to the removal of formulae with existential quantifiers that would result in large clique sizes of the ground Markov Network. Third, the isolated model performs extremely poor, particularly for argument classification. Here features defined for the *role* predicate can not make any use of the information in previous stages. Finally, the additional global formulae do improve performance, although not substantially.

4.1 Analysis

A substantial amount of errors in our submitted results (Full) can be attributed to the seemingly random assignment of the very low frequency label "R-AA" (appears once in the training set) to token pairs that should either have a different role or no role at all. Without these false positives, precision would increase by about 1%. Interestingly, this type of error completely disappears for the bottom-up model (Up) and thus seem to be crucial in order understand why this model can outperform the full model.

We believe that this type of error is an artifact of the training regime. For the full model the weights of the *role* predicate only have ensure that the right (true positive) role is the relative winner among all roles. In the bottom-up model they also have to make sure that their cumulative weight is non-negative – otherwise simply not assigning a role r for (p, a) would increase the score even if *hasRole(p,a)* is predicted with high confidence. Thus more weight is shifted towards the correct roles. This helps the right label to win more likely over the "R-AA" label, whose weights have rarely been touched and are closer to zero.

Likewise, in the bottom-up model the total weight of the *hasRole* features of a wrong (false positive) candidate token pair must be nonpositive. Otherwise picking the wrong candidate would increase overall score and no *role* features can re-

[6] While we did use information of the open dataset we do believe that it is possible to train a stacked parsing-SRL system that would perform similarily. If so, our system would have the 5th best semantic scores among the 20 participants of the closed track.

ject this decision because the corresponding structural constraints are missing. Thus more weight is shifted away from false positive candidates, resulting in a higher precision of the *hasRole* predicate. This also means that less wrong candidates are proposed, for which the "R-AA" role is more likely to be picked because its weights have hardly been touched.

However, it seems that by increasing precision in this way, we decrease recall for out-of-domain data. This leads to a lower F1 score for the bottom-up model on the Brown test set.

Another prominent type of errors appear for nominal predicates. Our system only recovers only about 80% of predicates with "NN", "NNS" and "NNP" tags (and classifies about 90% of these with the right predicate sense). Argument identification and classification performs equally bad. For example, for the "A0" argument of "VB" predicates we get an F-score of 82.00%. For the "A0" of "NN" predicates F-score is 65.92%. The features of our system are essentially taken from the work done on PropBank predicates and we did only little work to adapt these to the case of nominal predicates. Putting more effort into designing features specific to the case of nominal predicates might improve this situation.

5 Conclusion

We presented a Markov Logic Network that jointly performs predicate identification, argument identification and argument classification for SRL. This network achieves the second best semantic F-scores in the Open Track of the CoNLL shared task.

Experimentally we show that results can be further improved by using an MLN that resembles a conventional SRL bottom-up pipeline (but is still jointly trained and globally normalised) instead of a fully connected model. We hypothesise that when training this model more weight is shifted away from wrong argument candidates and more weight is shifted towards correct role labels. This results in higher precision for argument identification and better accuracy for argument classification.

Possible future work includes better treatment of nominal predicates, for which we perform quite poorly. We would also like to investigate the impact of linguistically motivated global formulae more thoroughly. So far our model benefits from them, albeit not substantially.

References

Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 2003.

V. Punyakanok, D. Roth, and W. Yih. Generalized inference with multiple semantic role labeling systems. In *Proceedings of the Annual Conference on Computational Natural Language Learning*, 2005.

Matthew Richardson and Pedro Domingos. Markov logic networks. Technical report, University of Washington, 2005.

Sebastian Riedel. Improving the accuracy and efficiency of map inference for markov logic. In *Proceedings of the Annual Conference on Uncertainty in AI*, 2008.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005.

Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In *Proceedings of the Annual Conference on Empirical Methods in Natural Language Processing*, 2004.