A Simple and Accurate Syntax-Agnostic Neural Model for Dependency-based Semantic Role Labeling

Citation for published version:

Digital Object Identifier (DOI):
10.18653/v1/K17-1041

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
A Simple and Accurate Syntax-Agnostic Neural Model for Dependency-based Semantic Role Labeling

Diego Marcheggiani\textsuperscript{1}, Anton Frolov\textsuperscript{2}, Ivan Titov\textsuperscript{1,3}
\textsuperscript{1}ILLC, University of Amsterdam
\textsuperscript{2}Machine Intelligence Department, Yandex
\textsuperscript{3}ILCC, School of Informatics, University of Edinburgh
marcheggiani@uva.nl
anton-fr@yandex-team.ru
ititov@inf.ed.ac.uk

Abstract

We introduce a simple and accurate neural model for dependency-based semantic role labeling. Our model predicts predicate-argument dependencies relying on states of a bidirectional LSTM encoder. The semantic role labeler achieves competitive performance on English, even without any kind of syntactic information and only using local inference. However, when automatically predicted part-of-speech tags are provided as input, it substantially outperforms all previous local models and approaches the best reported results on the English CoNLL-2009 dataset. We also consider Chinese, Czech and Spanish where our approach also achieves competitive results. Syntactic parsers are unreliable on out-of-domain data, so standard (i.e., syntactically-informed) SRL models are hindered when tested in this setting. Our syntax-agnostic model appears more robust, resulting in the best reported results on standard out-of-domain test sets.

1 Introduction

The task of semantic role labeling (SRL), pioneered by Gildea and Jurafsky (2002), involves the prediction of predicate argument structure, i.e., both identification of arguments as well as their assignment to an underlying semantic role. These representations have been shown to be beneficial in many NLP applications, including question answering (Shen and Lapata, 2007) and information extraction (Christensen et al., 2011). Semantic banks (e.g., PropBank (Palmer et al., 2005)) often represent arguments as syntactic constituents or, more generally, text spans (Baker et al., 1998). In contrast, CoNLL-2008 and 2009 shared tasks (Surdeanu et al., 2008; Hajic et al., 2009) popularized dependency-based semantic role labeling where the goal is to identify syntactic heads of arguments rather than entire constituents. Figure 1 shows an example of such a dependency-based representation: node labels are senses of predicates (e.g., “01” indicates that the first sense from the PropBank sense repository is used for predicate makes in this sentence) and edge labels are semantic roles (e.g., A0 is a proto-agent, ‘doer’).

Until recently, state-of-the-art SRL systems relied on complex sets of lexico-syntactic features (Pradhan et al., 2005) as well as declarative constraints (Punyakanok et al., 2008; Roth and Yih, 2005). Neural SRL models instead exploited feature induction capabilities of neural networks, largely eliminating the need for complex hand-crafted features. Initially achieving state-of-the-art results only in the multilingual setting, where careful feature engineering is not practical (Gesmundo et al., 2009; Titov et al., 2009), neural SRL models now also outperform their traditional counterparts on standard benchmarks for English (FitzGerald et al., 2015; Roth and Lapata, 2016; Swayamdipta et al., 2016; Foland and Martin, 2015).

Recently, it has been shown that an accurate span-based SRL model can be constructed without relying on syntactic features (Zhou and Xu, 2015).
Nevertheless, the situation with dependency-based SRL has not changed: even recent state-of-the-art methods for this task heavily rely on syntactic features (Roth and Lapata, 2016; FitzGerald et al., 2015; Lei et al., 2015; Roth and Woodsend, 2014; Swayamdipta et al., 2016). In particular, Roth and Lapata (2016) argue that syntactic features are necessary for the dependency-based SRL and show that performance of their model degrades dramatically if syntactic paths between arguments and predicates are not provided as an input. In this work, we are the first to show that it is possible to construct a very accurate dependency-based semantic role labeler which either does not use any kind of syntactic information or uses very little (automatically predicted part-of-speech tags). This suggests that our LSTM model can largely implicitly capture syntactic information, and this information can, to a large extent, substitute treebank syntax.

Similarly to the span-based model of Zhou and Xu (2015) we use bidirectional LSTMs to encode sentences and rely on their states when predicting arguments of each predicate. We predict semantic dependency edges between predicates and arguments relying on LSTM states corresponding to the predicate and the argument positions (i.e. both edge endpoints). As semantic roles are often specific to predicates or even predicate senses (e.g., in PropBank (Palmer et al., 2005)), instead of predicting the role label (e.g., A0 for Sequa in our example), we predict predicate-specific roles (e.g., make-A0) using a compositional model. Both these aspects (predicting edges and compositional embeddings of roles) contrast our approach with that of Zhou and Xu (2015) who essentially treat the SRL task as a generic sequence labeling task. We empirically show that using these two ideas is crucial for achieving competitive performance on dependency SRL (+1.0% semantic F1 in our ablation studies on English). Also, unlike the span-based version, we observe that using automatically predicted POS tags is also important (+0.7% F1).

The resulting SRL model is very simple. Not only do we not rely on syntax, our model is also local, i.e., we do not globally score or constrain sets of arguments. On the standard English in-domain CoNLL-2009 benchmark we achieve 87.7 F1 which compares favorable to the best local model (86.7% F1 for PathLSTM (Roth and Lapata, 2016)) and approaches the best results overall (87.9% for an ensemble of 3 PathLSTM models with a reranker on top). When we experiment with Chinese, Czech and Spanish portions of the CoNLL-2009 dataset, we also achieve competitive results, even without any extra hyper-parameter tuning.

Moreover, as syntactic parsers are not reliable when used out-of-domain, standard (i.e., syntactically-informed) dependency SRL models are crippled when applied to such data. In contrast, our syntax-agnostic model appears to be considerably more robust: we achieve the best result so far on the English and Czech out-of-domain test set (77.7% and 87.2% F1, respectively). For English, this constitutes a 2.4% absolute improvement over the comparable previous model (75.3% for the local PathLSTM) and substantially outperforms any previous method (76.5% for the ensemble of 3 PathLSTMs). We believe that out-of-domain performance may in fact be more important than in-domain one: in practice linguistic tools are rarely, if ever, used in-domain.

The key contributions can be summarized as follows:

- we propose the first effective syntax-agnostic model for dependency-based SRL;
- it achieves the best results among local models on the English, Chinese and Czech in-domain test sets;
- it substantially outperforms all previous methods on the out-of-domain test set on both English and Czech.

Despite the effectiveness of our syntax-agnostic version, we believe that both integration of treebank syntax and global inference are promising directions and leave them for future work. In fact, the proposed SRL model, given its simplicity and efficiency, may be used as a natural building block for future global and syntactically-informed SRL models.2

2 Our Model

The focus of this paper is on argument identification and labeling, as these are the steps which have

---

1 In the CoNLL-2009 benchmark, predicates do not need to be identified: their positions are provided as input at test time. Consequently, as standard for dependency SRL, we ignore this subtask in further discussion.

2 The code is available at https://github.com/diegma/neural-dep-srl.
been previously believed to require syntactic information. For the predicate disambiguation subtask we use models from previous work.

In order to identify and classify arguments, we propose a model composed of three components:

- a word representation component that from a word \( w_i \) in a sentence \( w \) build a word representation \( x_i \);

- a Bidirectional LSTM (BiLSTM) encoder which takes as input the word representation \( x_i \) and provide a dynamic representation of the word and its context in a sentence;

- a classifier which takes as an input the BiLSTM representation of the candidate argument and the BiLSTM representation of the predicate to predict the role associated to the candidate argument.

2.1 Word representation

We represent each word \( w \) as the concatenation of four vectors: a randomly initialized word embedding \( x^{re} \in \mathbb{R}^{d_w} \), a pre-trained word embedding \( x^{pe} \in \mathbb{R}^{d_w} \), a randomly initialized part-of-speech tag embedding \( x^{pos} \in \mathbb{R}^{d_l} \) and a randomly initialized lemma embedding \( x^{le} \in \mathbb{R}^{d_l} \) that is only active if the word is one of the predicates. The randomly initialized embeddings \( x^{re}, x^{pos}, \) and \( x^{le} \) are fine-tuned during training, while the pre-trained ones are kept fixed, as in Dyer et al. (2015). The final word representation is given by \( x = x^{re} \circ x^{pe} \circ x^{pos} \circ x^{le} \), where \( \circ \) represents the concatenation operator.

2.2 Bidirectional LSTM encoder

One of the most effective ways to model sequences are recurrent neural networks (RNN) (Elman, 1990), more precisely their gated versions, for example, Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997).

Formally, we can define an LSTM as a function \( LSTM_{\theta}(x_{1:i}) \) that takes as input the sequence \( x_{1:i} \) and returns a hidden state \( h_i \in \mathbb{R}^{d_h} \). This state can be regarded as a representation of the sentence from the start to the position \( i \), or, in other words, it encodes the word at position \( i \) along with its left context. Bidirectional LSTMs make use of two LSTMs: one for the forward pass, and another for the backward pass, \( LSTM_F \) and \( LSTM_B \), respectively. In this way the concatenation of forward and backward LSTM states encodes both left and right contexts of a word, \( BiLSTM(x_{1:n}, i) = LSTM_F(x_{1:i}) \circ LSTM_B(x_{n:i}) \). In this work we stack \( k \) layers of bidirectional LSTMs, each layer takes the lower layer as its input.

2.3 Predicate-specific encoding

As we will show in the ablation studies in Section 3, encoding a sentence with a bidirectional LSTM in one shot and using it to predict the entire semantic dependency graph does not result in competitive SRL performance. Instead, similarly to Zhou and Xu (2015), we produce predicate-specific encodings of a sentence and use them to predict arguments of the corresponding predicate. This contrasts with most other applications of LSTM encoders (for example, in syntactic parsing (Kiperwasser and Goldberg, 2016; Cross and Huang, 2016) or machine translation (Sutskever et al., 2014)), where sentences are typically encoded once and then used to predict the entire structured output (e.g., a syntactic tree or a target sentence). Specifically, when identifying arguments of a given predicate, we add a predicate-specific feature to the representation of each word in the sentence by concatenating a binary flag to the word representation of Section 2.1. The flag is set to 1 for the word corresponding to the currently considered predicate, it is set to 0 otherwise. In this way, sentences with more than one predicate will be re-encoded by bidirectional LSTMs multiple times.
2.4 Role classifier

Our goal is to predict and label arguments for a given predicate. This can be accomplished by labeling each word in a sentence with a role, including the special ‘NULL’ role to indicate that it is not an argument of the predicate. We start with explaining the basic role classifier and then discuss two extensions, which we will later show to be crucial for achieving competitive performance.

2.4.1 Basic role classifier

The basic role classifier takes the hidden state of the top-layer bidirectional LSTM corresponding to the considered word at position $i$ and uses it to estimate the probability of the role $r$. Though we experimented with multilayer perceptrons, we obtained the best results with a simple log-linear model:

$$p(r|v_i, p) \propto \exp(W_r v_i),$$

where $v_i$ is the hidden state calculated by $BiLSTM(x_{1:n}, i)$, $p$ refers to the predicate and the symbol $\propto$ signifies proportionality. This is essentially equivalent to the approach used in Zhou and Xu (2015) for span-based SRL.\(^3\)

2.4.2 Incorporating predicate state

Since the context of a predicate in the sentence is highly informative for deciding if a word is its argument and for choosing its semantic role, we provide the predicate’s hidden state ($v_p$) as another input to the classifier (as in Figure 2):

$$p(r|v_i, v_p) \propto \exp(W_r (v_i \circ v_p)),$$

where, as before, $\circ$ denotes concatenation. Note that we are effectively predicting an edge between words $i$ and $p$ in the sentence, so it is quite natural to exploit hidden states corresponding to both endpoints.\(^4\)

Since we use predicate information within the classifier, it may seem that predicate-specific sentence encoding (Section 2.3) is not needed anymore. Moreover, predicting dependency edges relying on LSTM states of endpoints was shown effective in the context of syntactic dependency parsing without any form of re-encoding (Kiperwasser and Goldberg, 2016). Nevertheless, in our ablation studies we observed that foregoing predicate-specific encoding results in large performance degradation (-6.2\% $F_1$ on English). Though this dramatic drop in performance seems indeed surprising, the nature of the semantic dependencies, especially for nominal predicates, is different from general syntactic dependencies, with many arguments being far away from the predicates. Relations of these arguments to the predicate may be hard to encode with this simpler mechanism.

The two ways of encoding predicate information, using predicate-specific encoding and incorporating the predicate state in the classifier, turn out to be complementary.

2.4.3 Compositional modeling of roles

Instead of using a matrix $W_r$, we found it beneficial to jointly embed the role $r$ and predicate lemma $l$ using a non-linear transformation:

$$p(r|v_i, v_p, l) \propto \exp(W_{l,r} (v_i \circ v_p)), \quad \text{(3)}$$

$$W_{l,r} = ReLU(U(u_l \circ v_r)), \quad \text{(4)}$$

where $ReLU$ is the rectilinear activation function, $U$ is a parameter matrix, whereas $u_l \in \mathbb{R}^{d_l}$ and $v_r \in \mathbb{R}^{d_r}$ are randomly initialized embeddings of predicate lemmas and roles. In this way each role prediction is predicate-specific, and at the same time we expect to learn a good representation for roles associated to infrequent predicates. This form of compositional embedding is similar to the one used in FitzGerald et al. (2015).

3 Experiments

We applied our model to the English, Chinese, Czech and Spanish CoNLL-2009 datasets with the standard split into training, test and development sets. For English, we used external embeddings of Dyer et al. (2015) learned using the structured skip n-gram approach of Ling et al. (2015), for Chinese, we used external embeddings produced with the neural language model of Bengio et al. (2003). For Czech and Spanish, we used embeddings created with the model proposed by Bojanowski et al. (2016).

Similarly to Kiperwasser and Goldberg (2016) we used word dropout (Iyyer et al., 2015); we replaced a word with the $UNK$ token with probability $\frac{\alpha}{f_r[w]+\alpha}$, where $\alpha$ is an hyper-parameter and...
$fr(w)$ is the frequency of the word $w$. The predicted POS tags were provided by the CoNLL-2009 shared-task organizers. We used the same predicate disambiguator as in Roth and Lapata (2016) for English, the one used in Zhao et al. (2009) for Czech and Spanish, and the one used in Björkelund et al. (2009) for Chinese. The training objective was the categorical cross-entropy, and we optimized it with Adam (Kingma and Ba, 2015). The hyperparameter tuning and all model selection was performed on the English development set; the chosen values are shown in Table 1.

### Table 1: Hyperparameter values.

<table>
<thead>
<tr>
<th>Semantic role labeler</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_w$ (English word embeddings)</td>
<td>100</td>
</tr>
<tr>
<td>$d_w$ (Chinese word embeddings)</td>
<td>128</td>
</tr>
<tr>
<td>$d_w$ (Czech word embeddings)</td>
<td>300</td>
</tr>
<tr>
<td>$d_w$ (Spanish word embeddings)</td>
<td>300</td>
</tr>
<tr>
<td>$d_{pos}$ (POS embeddings)</td>
<td>16</td>
</tr>
<tr>
<td>$d_l$ (lemma embeddings)</td>
<td>100</td>
</tr>
<tr>
<td>$d_h$ (LSTM hidden states)</td>
<td>512</td>
</tr>
<tr>
<td>$d_r$ (role representation)</td>
<td>128</td>
</tr>
<tr>
<td>$d'_r$ (output lemma representation)</td>
<td>128</td>
</tr>
<tr>
<td>$k$ (BiLSTM depth)</td>
<td>4</td>
</tr>
<tr>
<td>$\alpha$ (word dropout)</td>
<td>0.25</td>
</tr>
<tr>
<td>learning rate</td>
<td>0.01</td>
</tr>
</tbody>
</table>

### 3.1 Results

We compared our full model (with POS tags and the classifier defined in Section 2.4.3) against state-of-the-art models for dependency-based SRL on English, Chinese, Czech and Spanish. For English, our model significantly outperformed all the local counter-parts (i.e., models which do not perform global inference) on the in-domain tests (see Table 2) with 87.6% F$_1$ for our model vs. 86.7% for PathLSTM (Roth and Lapata, 2016). When compared with global models, our model performed on-par with the state-of-the-art global version of PathLSTM.

Though we had not done any parameter selection for other languages (i.e., used the same parameters as for English), our model performed competitively across all languages we considered.

For Chinese (Table 4), the proposed model outperformed the best previous model (PathLSTM) with an improvement of 1.8% F$_1$.

For Czech (Table 5), our model, even though unlike previous work it does not use any kind of morphological features explicitly,\(^5\) was able to outperform the system that achieved the best score in the CoNLL-2009 shared task. The improvement is 0.8% F$_1$.

Finally, for Spanish (Table 6), our system, though again achieved competitive results, did not outperform the best CoNLL-2009 model and yielded results very similar to those of PathLSTM. One possible reason for this slightly weaker performance is the relatively small size of the Spanish training set (less than half of the English one). This suggests that our model, tuned on English, is likely over-parametrized or under-regularized for Spanish.

The results are especially strong on out-of-domain data. As shown in Table 3, our approach outperformed even ensemble models on the out-of-domain English data (77.7% vs. 76.5% for\(^5\) However, character level information is encoded in the external embeddings, see (Bojanowski et al., 2016).
the ensemble of PathLSTMs). Similarly, it performed very well on the out-of-domain Czech dataset scoring 87.2% F$_1$, with a 1.8% F$_1$ improvement over the best CoNLL-2009 participant (see Table 5, bottom). The favorable results on out-of-domain test sets are not surprising, as syntactic parsers, even the most accurate ones, usually struggle on domains different from the ones they have been trained on. This means that the syntactic trees they produce are unreliable and compromise the accuracy of SRL systems which rely on them. The error propagation can in principle be mitigated by exploiting a distribution over parse trees (e.g., encoded in a parse forest) rather than using a single ("Viterbi") parse. However, this is rarely feasible in practice. Since our model does not use predicted parse trees and instead relies on the ability of LSTMs to capture long distance dependencies and syntactic phenomena (Linzen et al., 2016), it is less brittle in this setting.

### 3.2 Ablation studies and analysis

In order to show the contribution of the modeling choices we made, we performed an ablation study on the English development set (Table 7). In these experiments we made individual changes to the model (one by one) and measured their influence on the model performance.

First, we observed that POS tag information is highly beneficial for obtaining competitive performance.

### Table 4: Results on the Chinese test set.

<table>
<thead>
<tr>
<th>System</th>
<th>P</th>
<th>R</th>
<th>F$_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Björkelund et al. (2009)</td>
<td>82.4</td>
<td>75.1</td>
<td>78.6</td>
</tr>
<tr>
<td>Zhao et al. (2009)</td>
<td>80.4</td>
<td>75.2</td>
<td>77.7</td>
</tr>
<tr>
<td>Roth and Lapata (2016)</td>
<td>83.2</td>
<td>75.9</td>
<td>79.4</td>
</tr>
<tr>
<td>Ours</td>
<td>83.4</td>
<td>79.1</td>
<td>81.2</td>
</tr>
</tbody>
</table>

### Table 5: Results on the Czech test sets.

<table>
<thead>
<tr>
<th>System</th>
<th>P</th>
<th>R</th>
<th>F$_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Björkelund et al. (2009)</td>
<td>88.1</td>
<td>82.9</td>
<td>85.4</td>
</tr>
<tr>
<td>Zhao et al. (2009)</td>
<td>88.2</td>
<td>82.4</td>
<td>85.2</td>
</tr>
<tr>
<td>Ours</td>
<td>86.6</td>
<td>85.4</td>
<td>86.0</td>
</tr>
</tbody>
</table>

### Table 6: Results on the Spanish test set.

<table>
<thead>
<tr>
<th>System</th>
<th>P</th>
<th>R</th>
<th>F$_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Björkelund et al. (2009)</td>
<td>82.9</td>
<td>74.3</td>
<td>76.5</td>
</tr>
<tr>
<td>Roth and Lapata (2016)</td>
<td>83.2</td>
<td>77.4</td>
<td>80.2</td>
</tr>
<tr>
<td>Ours</td>
<td>81.4</td>
<td>79.3</td>
<td>80.3</td>
</tr>
</tbody>
</table>

### Table 7: Ablation study on the English development set.

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>R</th>
<th>F$_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours (local)</td>
<td>87.7</td>
<td>85.5</td>
<td>86.6</td>
</tr>
<tr>
<td>w/o POS tags</td>
<td>87.3</td>
<td>84.5</td>
<td>85.9</td>
</tr>
<tr>
<td>w/o predicate-specific encoding</td>
<td>80.9</td>
<td>79.8</td>
<td>80.4</td>
</tr>
<tr>
<td>with basic classifier</td>
<td>86.7</td>
<td>84.5</td>
<td>85.6</td>
</tr>
</tbody>
</table>

Not using predicate-specific encoding (Section 2.3), or, in other words, doing one-pass encoding with no predicate flags, hurts the performance even more badly (6% drop in F$_1$ on the development set). This is somewhat surprising given that one-pass LSTM encoders performed competitively for syntactic dependencies (Kiperwasser and Goldberg, 2016; Cross and Huang, 2016) and suggests that major differences between the two problems require the use of different modeling approaches.

We also observed a 1.0% drop in F$_1$ when we follow Zhou and Xu (2015) and use the basic role classifier (Section 2.4.1). These results show that both predicate-specific encoding (Section 2.3) and exploiting predicate information in the classifier (Sections 2.4.2-2.4.3) are complementary.

We also studied how performance varies depending on the distance between a predicate and an argument (Figure 3). We compared our approach to the global PathLSTM model: PathLSTM is a natural reference point as it is the most accurate previous model, exploits similar modeling and representation techniques (e.g., word embeddings, LSTMs) but, unlike our approach, relies on predicted syntax. Contrary to our expectations, syntactically-driven and global PathLSTM was weaker for longer distances. We may speculate that syntactic paths for arguments further away from the predicate become unreliable. Though LSTMs are likely to be affected by a similar trend, their states may be able to capture the uncertainty about the structure and thus let the
Figure 3: $F_1$ as function of word distance. Percentages indicate the amount of arguments at a specific distance from a predicate.

Table 8: $F_1$ results on the English test set broken down into verbal and nominal predicates.

<table>
<thead>
<tr>
<th></th>
<th>Ours</th>
<th>PathLSTM</th>
<th>Freq. (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verbal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A0</td>
<td>90.5</td>
<td>90.4</td>
<td>15%</td>
</tr>
<tr>
<td>A1</td>
<td>92.0</td>
<td>91.8</td>
<td>21%</td>
</tr>
<tr>
<td>A2</td>
<td>80.3</td>
<td>80.2</td>
<td>5%</td>
</tr>
<tr>
<td>AM-*</td>
<td>77.9</td>
<td>77.0</td>
<td>16%</td>
</tr>
<tr>
<td>All</td>
<td>86.4</td>
<td>86.1</td>
<td>61%</td>
</tr>
<tr>
<td>Nominal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A0</td>
<td>81.8</td>
<td>81.5</td>
<td>10%</td>
</tr>
<tr>
<td>A1</td>
<td>85.1</td>
<td>85.5</td>
<td>16%</td>
</tr>
<tr>
<td>A2</td>
<td>78.5</td>
<td>79.8</td>
<td>7%</td>
</tr>
<tr>
<td>AM-*</td>
<td>72.5</td>
<td>73.2</td>
<td>5%</td>
</tr>
<tr>
<td>All</td>
<td>81.1</td>
<td>81.8</td>
<td>39%</td>
</tr>
</tbody>
</table>

In Table 8, we break down $F_1$ results on the English test set into verbal and nominal predicates, and again compare our results with PathLSTM. First, as expected, we observe that both models are less accurate in predicting semantic roles of nominal predicates. For verbal predicates, our model slightly outperformed PathLSTM in core roles (A0-2) and performed much better (0.9% $F_1$) in predicting modifiers (AM-*). This is very surprising as some information about modifiers is actually explicitly encoded in syntactic dependencies exploited by PathLSTM (e.g., the syntactic dependency TMP is predictive of the modifier role AM-TMP). Note though that the syntactic parser was trained on the same sentences (both data originates from WSJ sections 02-22 of Penn Treebank), and this can explain why these syntactic dependencies (e.g., TMP) may convey little beneficial information to the semantic role labeler. For nominal predicates, PathLSTM was more accurate than our model for all roles excluding A0. To get a better idea for what is happening, we plotted the $F_1$ scores as a function of the length of the shortest path between nominal predicates and their arguments. On one hand, Figure 4 shows that PathLSTM is more accurate on roles one syntactic arc away from the nominal predicate. Note that these are the majority (78%) of arguments. On the other hand, our model appears to be more accurate for arguments syntactically far from nominal predicates. This again suggests that PathLSTM struggles with harder cases.

Table 9: Argument recognition results broken down into verbal and nominal predicates.

<table>
<thead>
<tr>
<th></th>
<th>System</th>
<th>P</th>
<th>R</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verbal</td>
<td>PathLSTM</td>
<td>93.4</td>
<td>87.8</td>
<td>90.5</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>92.7</td>
<td>89.8</td>
<td>91.2</td>
</tr>
<tr>
<td>Nominal</td>
<td>PathLSTM</td>
<td>92.0</td>
<td>83.9</td>
<td>87.8</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>89.4</td>
<td>86.6</td>
<td>88.0</td>
</tr>
</tbody>
</table>

Figure 4: $F_1$ as function of syntactic distance for nominal predicates. Percentages indicate the amount of arguments at a specific distance from a nominal predicate.
Manual

Most of the stock \[ A2 \] selling \[ A2 \] pressure came \[ A0 \] from \[ A0 \] Wall Street professionals.

PathLSTM

Most of the stock \[ A2 \] selling \[ A2 \] pressure came from Wall Street professionals.

Ours

Most of the stock \[ A0 \] selling \[ A2 \] pressure came \[ A0 \] from \[ A0 \] Wall Street professionals.

<table>
<thead>
<tr>
<th>System</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td>Most of the stock [ A2 ] selling [ A2 ] pressure came [ A0 ] from [ A0 ] Wall Street professionals.</td>
</tr>
<tr>
<td>PathLSTM</td>
<td>Most of the stock [ A2 ] selling [ A2 ] pressure came from Wall Street professionals.</td>
</tr>
<tr>
<td>Ours</td>
<td>Most of the stock [ A0 ] selling [ A2 ] pressure came [ A0 ] from [ A0 ] Wall Street professionals.</td>
</tr>
</tbody>
</table>

Table 10: Example of errors for the nominal predicate pressure: A0 is a presser (proto-agent) and A2 is a goal.

Unlike verbal predicates, syntactic structure is less predictive of semantic roles for nominals (e.g., many arguments are noun modifiers). Consequently, we hypothesized that our model should be weaker than PathLSTM in recognizing arguments but should be on par with PathLSTM in assigning their roles. To test this, we looked into argument identification performance (i.e., ignored labels). Table 9 shows the accuracy of both models in recognizing arguments of nominal and verbal predicates. Our model appears more accurate in recognizing arguments of both nominal (88.0% vs 87.8% F1) and verbal predicates (91.2% vs. 90.5% F1). This, when taken together with weaker labeled F1 of our model for nominal predicates (Table 8), implies that, contrary to our expectations, it is the role labeling performance for nominals which is problematic for our model. Examples of this behavior can be seen in Table 10: all arguments of the predicate pressure are correctly recognized by our model but the role for the argument selling is not predicted correctly. In contrast, PathLSTM does not make any mistake with the labeling of the argument selling but fails to recognize from as an argument.

4 Related Work

Earlier approaches to SRL heavily relied on complex sets of lexico-syntactic features (Gildea and Jurafsky, 2002). Pradhan et al. (2005) used a support vector machine classifier and relied on two syntactic views (obtained with two different parsers), for feature extraction. In addition to hand-crafted features, Roth and Yih (2005) enriched CRFs with an integer linear programming inference procedure in order to encode non-local constraints in SRL; Toutanova et al. (2008) employed a global reranker for dealing with structural constraint; while Surdeanu et al. (2007) studied several combination strategies of local and global features obtained from several independent SRL models.

In the last years there has been a flurry of work that employed neural network approaches for SRL. FitzGerald et al. (2015) used hand-crafted features within an MLP for calculating potentials of a CRF model; Roth and Lapata (2016) extended the features of a non-neural SRL model with LSTM representations of syntactic paths between arguments and predicates; Lei et al. (2015) relied on low-rank tensor factorization that captured interactions between arguments, predicate, their syntactic path and semantic roles; while Collobert et al. (2011) and Foland and Martin (2015) used convolutional networks as sentence encoder and a CRF as a role classifier, both approaches employed a rich set of features as input of the convolutional encoder. Finally, Swayamdipta et al. (2016) jointly modeled syntactic and semantic structures; they extended one of the earliest neural approaches for SRL (Henderson et al., 2008; Titov et al., 2009; Gesmundo et al., 2009), with more sophisticated modeling techniques, for example, using LSTMs instead of vanilla RNNs.

Another related line of work (Naradowsky et al., 2012; Gormley et al., 2014), instead of relying on treebank syntax, integrated grammar induction as a sub-component into their statistical model. In this way, similarly to us, they do not use treebank syntax but rather rely on the ability of their joint model to induce syntax appropriate for SRL. Their focus was primarily on the low resource setting (where syntactic annotation is not available), whereas in standard set-ups their performance was not as strong. It would be interesting to see if explicit modeling of latent syntax is also beneficial when used in conjunction with LSTMs.

5 Conclusions

We proposed a neural syntax-agnostic method for dependency-based SRL. Our model is simple and fast, and surpasses comparable approaches (no
system combination, local inference) on the standard in-domain CoNLL-2009 benchmark for English, Chinese, Czech and Spanish. Moreover, it outperforms all previous methods (including ensembles) in the arguably more realistic out-of-domain setting in both English and Czech. In the future, we will consider integration of syntactic information and joint inference.

Acknowledgments

The project was supported by the European Research Council (ERC StG BroadSem 678254), the Dutch National Science Foundation (NWO VIDI 639.022.518) and an Amazon Web Services (AWS) grant. The authors would like to thank Michael Roth for his helpful suggestions.

References


James Cross and Liang Huang. 2016. Incremental parsing with minimal features using bi-directional LSTM. In Proceedings of ACL.


Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In Proceedings of EMNLP.


Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. TACL.

Tao Lei, Yuan Zhang, Lluís Márquez, Alessandro Moschitti, and Regina Barzilay. 2015. High-order low-rank tensors for semantic role labeling. In Proceedings of NAACL.


Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In Proceedings of ACL.

Michael Roth and Kristian Woodsend. 2014. Composition of word representations improves semantic role labelling. In Proceedings of EMNLP.


