Syntax-aware Semantic Role Labeling without Parsing

Rui Cai and **Mirella Lapata**

Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
Rui.Cai@ed.ac.uk mlap@inf.ed.ac.uk

Abstract

In this paper we focus on learning dependency aware representations for semantic role labeling without recourse to an external parser. The backbone of our model is an LSTM-based semantic role labeler jointly trained with two *auxiliary* tasks: predicting the dependency label of a word and whether there exists an arc linking it to the predicate. The auxiliary tasks provide syntactic information that is specific to semantic role labeling and are learned from training data (dependency annotations) without relying on existing dependency parsers, which can be noisy (e.g., on out-of-domain data or infrequent constructions). Experimental results on the CoNLL-2009 benchmark dataset show that our model outperforms the state of the art in English, and consistently improves performance in other languages, including Chinese, German, and Spanish.

1 Introduction

Semantic role labeling (SRL) aims to identify the arguments of semantic predicates in a sentence and label them with a set of predefined relations (e.g., ''who'' did ''what'' to ''whom,'' ''when,'' and ''where''). Semantic roles capture basic predicate-argument structure while abstracting over surface syntactic configurations and have been shown to benefit a wide spectrum of applications ranging from machine translation (Aziz et al., 2011; Marcheggiani et al., 2018) to information extraction (Christensen et al., 2011) and summarization (Khan et al., 2015).

The successful application of neural networks to a variety of NLP tasks (Bahdanau et al., 2015; Vinyals et al., 2015) has provided strong impetus to develop deep end-to-end models for SRL that forego the need for extensive feature engineering. Recently proposed models (Zhou and Xu, 2015; He et al., 2017; Marcheggiani et al., 2017) largely rely on bi-directional recurrent neural networks (Hochreiter and Schmidhuber, 1997) and predict semantic roles from textual input. They achieve competitive results while being syntax agnostic, thereby challenging conventional wisdom that parse trees provide a better form of representation for assigning semantic role labels (Johansson and Nugues, 2008).

There are, however, good reasons why syntax ought to help semantic role labeling. First and foremost, SRL systems are trained on datasets whose semantic role annotations have been produced on top of treebanked corpora, and as a result are closely tied to syntactic information. An example sentence with roles labeled in the style of PropBank (Palmer et al., 2005) is shown in Figure 1. Here, many arcs in the syntactic dependency graph are mirrored in the semantic dependency graph, suggesting that syntactic dependencies could provide useful information to the SRL task. Secondly, predicates are typically associated with a *standard* linking, that is, a deterministic mapping from syntactic roles to semantic ones (Lang and Lapata, 2010; Surdeanu et al., 2008). For example, subject (SBJ) is commonly mapped onto A0, whereas A1 is often realized as object (OBJ). Even in cases where there is no canonical mapping, dependency labels are still closely related to certain semantic roles, like the syntactic function TMP and the semantic role AM-TMP.

The question of how to effectively incorporate syntactic information into sequential neural network models has met with different answers in the literature. Marcheggiani and Titov (2017) make use of graph convolutional networks (GCNs; Duvenaud et al., 2015; Kearnes et al., 2016; Kipf and Welling, 2017) as a means to represent syntax in neural models. GCNs are used to encode syntactic dependency trees in combination with encoders based on long short-term memory units

Transactions of the Association for Computational Linguistics, vol. 7, pp. 343–356, 2019. Action Editor: Alessandro Moschitti.
Submission batch: 10/2018; Revision batch: 1/2019; Published 6/2019.

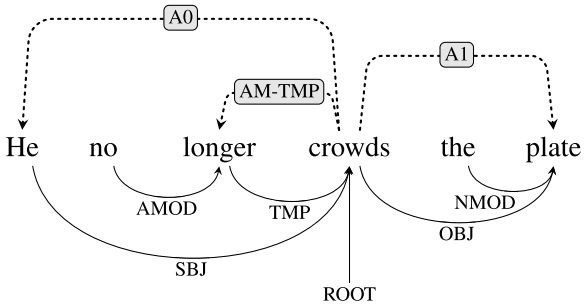

Figure 1: Example sentence from the CoNLL-2009 English dataset annotated with syntactic dependencies (bottom) and semantic roles (top).

(LSTMs). He et al. (2018) emphasize the role of syntax in argument identification rather than role labeling. Specifically, they develop an argument pruning algorithm that operates over dependency structures and selects argument candidates subject to a parameter determining their distance from the predicate. The predicate and its arguments are then encoded with an LSTM similar to Marcheggiani and Titov (2017). He et al. (2017) incorporate syntax at decoding time, in the form of constraints on the output structure (e.g., consistency with a parse tree is enforced by rejecting or penalizing arguments that are not constituents), whereas Strubell et al. (2018) incorporate syntactic information in a multi-task neural network model that simultaneously performs part-of-speech tagging, dependency parsing, predicate detection, and SRL.

In this paper we argue that syntactic information is important for semantic role labeling and syntactic parsing is not. Despite recent advances in dependency parsing (Dozat and Manning, 2016; Kiperwasser and Goldberg, 2016), the use of an external parser often leads to pipeline-style architectures where errors propagate to later processing stages, affecting model performance. To mitigate such errors, Marcheggiani and Titov (2017) calculate a scalar gate for each edge in the dependency tree. And perhaps unsurprisingly, the performance of their system decreases when more than one GCN layer is stacked, as the effect of noisy information is amplified. Our key insight is to focus on dependency labels, which provide important information for semantic roles without requiring access to a full-blown syntactic representation of the sentence. Our model concentrates on the dependency structures pertaining to the predicate in a given sentence rather than capturing information relating to every arc in the dependency tree. The majority of arguments (approximately 68%) in the CoNLL-2009 English development set are directly linked to the predicate or are predicates themselves.

Our work focuses on learning dependency-aware representations without recourse to an external parser. The backbone of our model is a semantic role labeler jointly trained with a dependency information extractor with two *auxiliary* tasks: predicting the dependency label of a word and whether there exists an arc linking it to the predicate. The two auxiliary tasks provide dependency information that is specific to the SRL task and is learned from training data (dependency annotations) without ever utilizing an external parser. Our model falls under the general paradigm of multi-task learning (Caruana, 1993) which aims to improve a main task by jointly learning one or more related auxiliary tasks. Multi-task learning has been successfully applied to various sequence-prediction tasks including chunking, tagging (Collobert et al., 2011b; Bjerva et al., 2016; Plank, 2016; Søgaard and Goldberg, 2016; Hashimoto et al., 2017), name error detection (Cheng et al., 2015), machine translation (Luong et al., 2016), supersense tagging (Bingel and Søgaard, 2017), entailment (Hashimoto et al., 2017), and semantic role labeling (Collobert et al., 2011b; Strubell et al., 2018).

Experimental results on the CoNLL-2009 benchmark dataset show that our model is able to outperform the state of the art in English, and to improve SRL performance in other languages, including Chinese, German, and Spanish.

2 Model Description

Most supervised semantic role labeling systems adopt an architecture consisting of the following steps: (a) predicate identification and disambiguation (e.g., *crowds* in Figure 1 is a predicate with sense *crowd.02*); (b) argument identification (e.g., the arguments of the predicate *crowds* in Figure 1 are *He* and *plate*); and (c) argument classification (e.g., the semantic roles for *He* and *plate* are A0 and A1, respectively). In this paper we focus solely on identifying arguments and labeling them with semantic roles using an off-the-shelf disambiguation model for the first step (Björkelund et al., 2010; Roth and Lapata, 2016).

Our semantic role labeler is built on top of the syntax-agnostic model of Marcheggiani et al.

(2017), which achieves good performance on the CoNLL-2009 English dataset without making use of a parser. Figure 2 provides a schematic overview of our model, which has two main components, namely, a dependency information extractor, and a semantic role predictor. The aim of the dependency extractor is to learn syntactic information for each word which subsequently serves as input (combined with word representations) to the semantic role labeler. The dependency extractor consists of:

- a word representation component (which boils down to a simple embedding look-up);
- a K-layer bidirectional LSTM (BiLSTM) encoder that takes as input the representation of each word in a sentence and produces context-dependent embeddings;
- two multilayer perceptron (MLP) networks that predict the dependency label and type of arc between a word and a predicate.

The semantic role predictor consists of:

- a word representation component that encapsulates predicate-specific dependency information;
- a J-layer BiLSTM encoder that takes as input the representation of each word in a sentence;
- a classifier that takes as input the BiLSTM representations of predicates and their arguments and assigns semantic roles to the latter.

In the following sections we describe these two components more formally.

2.1 Dependency Information Extractor

The dependency information extractor (bottom block in Figure 2) operates on sentences after predicate identification (and disambiguation) has taken place. It learns important syntactic information (i.e., the dependency relation between a predicate and its candidate arguments), which is subsequently used by the semantic role labeler. In the description below we assume that predicates are known.

Sentence Encoder We represent words as the concatenation of three vectors: a randomly initialized word embedding $x'_{re} \in R^{d_w}$, a pre-trained word embedding $x'_{pe} \in R^{d_w}$ estimated on an external text collection, and a character embedding x_i^{ce} learned by convolutional neural network (CNN) with bidirectional LSTM (BiLSTM). The final word representation is given by $x = x'_{re} \circ x'_{pe} \circ x'_{ce}$, where $\circ$ represents the concatenation operator.

Following Marcheggiani et al. (2017), sentences are represented using a bi-directional recurrent neural network with LSTMs (Hochreiter and Schmidhuber, 1997). A bidirectional LSTM receives at time step t a representation x for each word and recursively computes two hidden states, one for the forward pass ($\overrightarrow{h_t}$), and another one for the backward pass ($\overleftarrow{h_t}$). Each word is the concatenation of its forward and backward LSTM state vectors $h_t = \overrightarrow{h_t} \circ \overleftarrow{h_t}$.

Dependency Label Prediction Our model focuses on predicting the dependency labels of predicates as opposed to all words in a dependency tree. For each arc (w, p) consisting of predicate p and modifier w, our model assigns the dependency label l with the highest score according to a multilayer perceptron (MLP):

$$label(w, p) = \arg\max_{l \in labels} \text{MLP}_{\mathcal{LBL}}(h_w \circ h_p)[l] \quad (1)$$

where l are pre-defined dependency labels (e.g., SUBJ, OBJ), and h_w and h_p are the hidden states of the bidirectional sentence encoder representing word w and predicate p, respectively (see the bottom BiLSTM in Figure 2).

The inner structure of the MLP is shown in Figure 3. Dependency label scores for arc (w, p) are calculated as follows:

$$\text{MLP}_{\mathcal{LBL}} = W_{\mathcal{LBL}} \tanh(W_w h_w + W_p h_p) \quad (2)$$

where W_w, W_p, and $W_{\mathcal{LBL}}$ are parameter matrices. In our experiments we use a two-layered BiLSTM encoder following Kiperwasser and Goldberg (2016), who show that placing a dependency classifier on top of two BiLSTM layers achieves best results for labeled dependency parsing.

Link Type Prediction Our aim is to capture how semantic predicates are linked to adjacent words in a sentence. Specifically, we are interested in predicting whether they are linked, and, if they are, what type of link they have. Again, we only

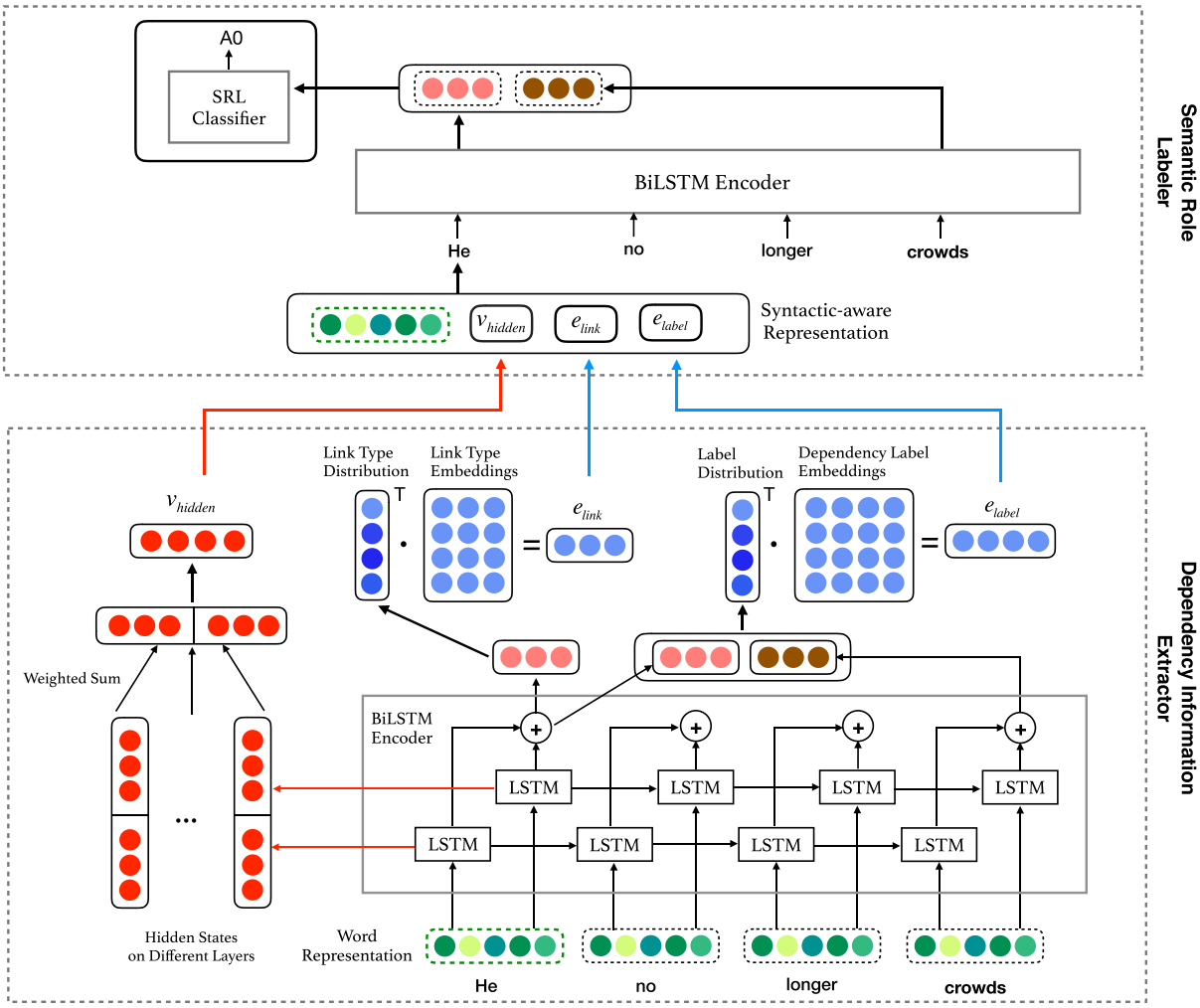

Figure 2: Model overview: Dependency information extractor (bottom) and a semantic role labeler (top). Colored lines are syntax-aware representations for the word *He* and are shared between the two components.

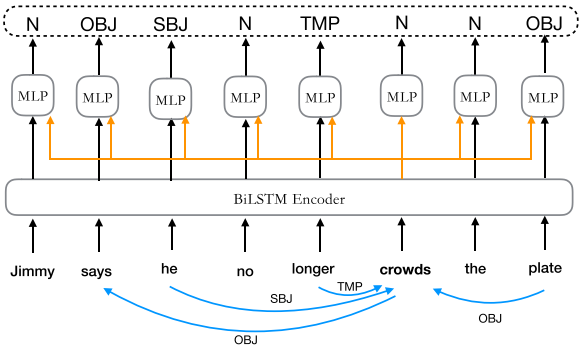

Figure 3: Dependency label prediction. Blue lines denote dependency arcs between words in the sentence and the predicate *crowds*.

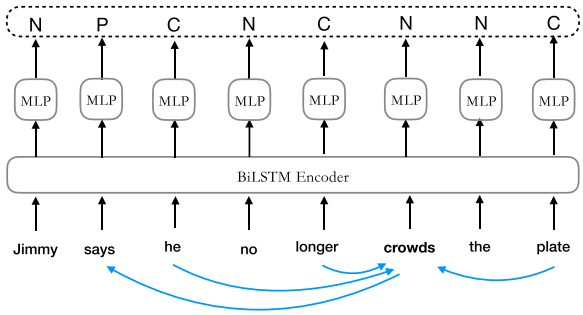

Figure 4: Predicting the link type of each word in a sentence. Blue lines denote dependency arcs between words in the sentence and the predicate *crowds*.

focus on syntactic arcs pertaining to the semantic predicate, rather than *all* arcs in the dependency tree, and assign each word a label representing its link type in relation to the predicate. Tag N indicates there is no arc between a word and the predicate, whereas tags C and P represent child and parent nodes of the predicate, respectively. Figure 4 shows an example of how these labels are processed by our model. We extract predicate linking information from dependency

tree annotations, and use a MLP predictor to identify link type information for word w_t :

$$\text{MLP}_{\mathcal{LNK}} = W_{\mathcal{LNK}} \tanh(W_L h_t) \quad (3)$$

A key difference between the dependency label classifier and the link type predictor is that the latter does not explicitly use h_p, namely, predicate-specific information. By doing this, we force the model to learn the linking between long-distance word pair. For a sentence with more than one predicates, the dependency information extractor will produce different results for each predicate.

2.2 Syntax-aware Semantic Role Labeler

Our semantic role labeler (upper block in Figure 2) estimates the probability of role r given the hidden states of candidate argument word i and predicate word p:

$$p(r|t_i, t_p, l) \propto exp(W_{l,r}(t_i \circ t_p)), \quad (4)$$

where t_i and t_p are representations for word i and predicate p, respectively, and l is the lemma of predicate p; symbol $\circ$ denotes concatenation and $\propto$ signifies proportionality. Following Marcheggiani and Titov (2017), matrix $W_{l,r}$ is the joint embedding of role r and the predicate lemma l using a non-linear transformation:

$$W_{l,r} = \text{ReLU}(U(e_l \circ e_r)) \quad (5)$$

where U is a parameter matrix, and $e_l \in R^{d_l'}$ and $e_r \in R^{d_r}$ are randomly initialized embeddings of predicate lemmas and roles. This way, each role prediction is predicate-specific, and a good representation for roles associated with infrequent predicates can be learned.

The model's training objective $\mathcal{L}$ is the weighted sum of objectives for the SRL task and the two auxiliary tasks. Formally,

$$\mathcal{L} = \mathcal{L}_{\mathcal{SRL}} + \alpha(\mathcal{L}_{\mathcal{LBL}} + \mathcal{L}_{\mathcal{LNK}}), \quad (6)$$

where $\mathcal{L}_{\mathcal{SRL}}$, $\mathcal{L}_{\mathcal{LBL}}$, and $\mathcal{L}_{\mathcal{LNK}}$ are the categorical cross-entropy of SRL, dependency label prediction, and link type prediction, respectively. α is a scalar weight for the auxiliary tasks whose value is tuned experimentally on the development dataset.

We will next discuss how the hidden states t_i and t_p are obtained taking into account the the dependency extractor introduced earlier.

Input Layer Representations Given a sentence with words $(w_1, \ldots, w_N)$, we form a syntax-agnostic word representation x for each word using randomly initialized word embedding $x_{re} \in R^{d_w}$, pre-trained word embedding $x_{pe} \in R^{d_w}$ estimated on an external text collection, randomly initialized part-of-speech tag embedding $x_{pos} \in R^{d_{pos}}$, and randomly initialized lemma embedding $x^{le} \in R^{d_l}$ (active only if the word is a predicate). The word representation is thus given by $x = x_{re} \circ x_{pe} \circ x_{pos} \circ x_{le}$, where $\circ$ represents the concatenation operator.

The parameters of the pre-trained word embeddings x_{pe} are shared with the word embeddings $x_{pe}^{'}$ used for our dependency information extractor, and are updated during training. In order to obtain more syntax-aware representations, we utilize hidden-layer representations v_{hidden} and dependency embeddings (e_{label} and e_{link}). The final representation R, which serves as input to the SRL, is the concatenation of three syntactically informed representations:

$$R = x \circ v_{hidden} \circ e_{label} \circ e_{link} \quad (7)$$

Hidden-layer Representations In order to compute v_{hidden}, we draw inspiration from ELMo (Peters et al., 2018), a recently proposed model for generating word representations based on bidirectional LSTMs trained with a coupled language model objective. Unlike more traditional word embeddings (Mikolov et al., 2013), ELMo representations are deep, essentially a linear combination of the representations learned at all layers of the LSTM instead of just the final layer.

We also utilize the combination of the intermediate layer representations in the dependency information extractor. Given sentence $(w_1, \ldots, w_N)$, a BiLSTM encoder with L layers computes for each word a set of $2L$ representations:

$$\begin{aligned} S &= \{\vec{h}_j, \overleftarrow{h}_j | j = 1, \ldots, L\} \\ &= \{h_j | j = 1, \ldots, L\} \end{aligned} \quad (8)$$

where $h_j = [\vec{h}_j; \overleftarrow{h}_j]$ for each hidden layer in the BiLSTM encoder.

In order to make use of all layers in S for our SRL task, we collapse them into a single vector. Although we could simply concatenate these representations or select the top layer, we compute

vector v_{hidden} as a weighting of the BiLSTM layers, followed by a non-linear projection:

$$v_{hidden} = \text{ReLU}(W_{hidden}(\gamma \sum_{j=1}^{j=L} \beta_j h_j)) \quad (9)$$

where β are softmax-normalized weights for h_j, and the scalar parameter γ is of practical importance for optimization, as it allows the model to scale the weighted hidden-layer representations (Peters et al., 2018); both β and γ are updated during training.

Dependency Embeddings An obvious way to take advantage of the dependency label predictions (see Section 2.1) would be to use the embedding e_l of the label l with the highest score. However, this would place too much emphasis on high confidence labels, which can be noisy. Instead, we use the weighted composition of *all* dependency label embeddings e_{label}, which is calculated as:

$$e_{label} = \sum_{l \in labels} \text{softmax}(\text{MLP}_{\mathcal{LBL}})[l] * e_l \quad (10)$$

where the weight of each label embedding is the normalized probability given by the label classifier. Analogously, we represent dependency link information e_{link} as:

$$e_{link} = \sum_{l \in \{N,C,P\}} \text{softmax}(MLP_{\mathcal{LNK}})[l] * e_l \quad (11)$$

3 Experiments

We implemented our model in PyTorch[1] and evaluated it on the English, Chinese, German, and Spanish CoNLL-2009 benchmark datasets following the standard training, testing, and development set splits. The datasets contain gold-standard dependency annotations, and also gold lemmas, part-of-speech tags, and morphological features. Data for the different languages was generated by merging various language specific treebanks such as the Penn Treebank (Parcus et al., 1993) and Brown corpus (Francis and Kucera, 1979) for English, the Prague Dependency Treebank for Czech (Hajičová et al., 1999), the Chinese Treebank (Xue et al., 2005), and Proposition Bank (Xue and Palmer, 2009) for Chinese, and so on (we refer the interested reader to Hajic et al. [2009] for details on individual languages and their annotations).

Hyperparameter	value
d_w (English word embeddings)	100
d_w (other languages word embeddings)	300
d_c (character embeddings)	300
d_{pos} (POS embeddings)	16
d_l (lemma embeddings)	100
d_h (LSTM hidden states)	300
d_{hidden} (hidden layer representation)	200
d_{output} (output label embeddings)	32
d_r (role representation)	128
$d_l^{'}$ (output lemma representation)	128
K (BiLSTM depth)	4
J (BiLSTM depth)	2
batch size	30
input layer dropout rate	0.3
hidden layer dropout rate	0.3
learning rate	0.001
auxiliary tasks loss weight α	0.5

Table 1: Hyperparameter values.

For experiments on English, we used the embeddings of Dyer et al. (2015), which were learned using the structured skip n-gram approach of Ling et al. (2015). In a few experiments we also used English character embeddings following He et al. (2018). These were pre-trained with a CNN-BiLSTM model (Peters et al., 2018) on the 1 Billion Word Benchmark,[2] which is publicly released as part of the AllenNLP toolkit.[3] Embeddings[4] for Chinese, Spanish, and German were pre-trained on Wikipedia using `fastText` (Bojanowski et al., 2017).

The dropout mechanism was applied to the input layer and the top hidden layer of the BiLSTM encoders. We used the Adam optimizer (Kingma and Ba, 2014) to train our models. We performed hyperparameter tuning and model selection on the English development set; optimal hyperparameter values (for all languages) are shown in Table 1. The BiLSTM for the dependency extractor had two layers, and the BiLSTM for the semantic role labeler had four.

Predicted POS tags were provided by the CoNLL-2009 shared-task organizers. For all language, we used the same predicate disambiguator

[1] Our code is available at `https://github.com/RuiCaiNLP/SRL_DEP`.

[2] `http://www.statmt.org/lm-benchmark/`

[3] `https://allennlp.org/elmo`

[4] Publicly available at `https://github.com/facebookresearch/fastText`

Single Models (with external parser)	P	R	F
Björkelund et al. (2010)	87.1	84.5	85.8
Lei et al. (2015)	-	-	86.6
FitzGerald et al. (2015)	-	-	86.7
Roth and Lapata (2016)	88.1	85.3	86.7
Marcheggiani and Titov (2017)	89.1	86.8	88.0
He et al. (2018)	89.7	89.3	89.5
Single Models (w/o external parser)	P	R	F
Marcheggiani et al. (2017)	88.7	86.8	87.7
He et al. (2018)	89.5	87.9	88.7
Ours (w/o ELMo)	90.5	88.6	**89.6**
Ours (with ELMo)	90.9	89.1	**90.0**
Ensemble Models	P	R	F
FitzGerald et al. (2015)	-	-	87.7
Roth and Lapata (2016)	90.3	85.7	87.9
Marcheggiani and Titov (2017)	90.5	87.7	89.1

Table 2: English results on the CoNLL-2009 in-domain (WSJ) test set.

Single Models (with external parser)	P	R	F
Björkelund et al. (2010)	75.7	72.2	73.9
Lei et al. (2015)	-	-	75.6
FitzGerald et al. (2015)	-	-	75.2
Roth and Lapata (2016)	76.9	73.8	75.3
Marcheggiani and Titov (2017)	78.5	75.9	77.2
He et al. (2018)	81.9	76.9	79.3
Single Models (w/o external parser)	P	R	F
Marcheggiani et al. (2017)	79.4	76.2	77.7
He et al. (2018)	81.7	76.1	78.8
Ours (w/o ELMo)	80.5	78.2	**79.4**
Ours (with ELMo)	80.8	78.6	**79.7**
Ensemble Models	P	R	F
FitzGerald et al. (2015)	-	-	75.5
Roth and Lapata (2016)	79.7	73.6	76.5
Marcheggiani and Titov (2017)	80.8	77.1	78.9

Table 3: English results on the CoNLL-2009 out-of domain (Brown) test set.

as in Roth and Lapata (2016) which uses a pipeline of mate-tools (Björkelund et al., 2010).

3.1 Results

Our results on the English (in-domain) test set are summarized in Table 2. We compared our system against previous models that use a dependency parser (first block in the table) and those that do not (second block). We also report the results of various ensemble SRL models (third block). For a fair comparison with He et al. (2018), we present a variant of our model with character-based ELMo embeddings. Most comparisons involve neural systems that are based on BiLSTMs (Marcheggiani et al., 2017; Marcheggiani and Titov, 2017; He et al., 2018) or use neural networks for learning SLR-specific embeddings (FitzGerald et al., 2015; Roth and Lapata, 2016). We also report the results of two strong symbolic models based on tensor factorization (Lei et al., 2015) and a pipeline of modules that carry out the tokenization, lemmatization, part-of-speech tagging, dependency parsing, and semantic role labeling (Björkelund et al., 2010).

As can be seen in Table 2, our model outperforms previous single and ensemble models, irrespective of whether they make use of a dependency parser or not. When taking into account ELMo embeddings, our model achieves 90.0% F_1, which is an absolute improvement of 0.5 percentage point over the state of the art (He et al., 2018). It is also interesting to note that the performance of He et al. (2018) drops from 89.5% to 88.7% when a dependency parser is not available, whereas our model is able to extract dependency information on its own, without relying on external syntactic parsers.

Results on the out-of-domain English test set are presented in Table 3. We include comparisons with the same models as in the in-domain case. Again, our syntax-light model outperforms previously published single and ensemble models, even when ELMo character embeddings are not taken into account (F_1 increases from 79.4% to 79.7% with ELMo). It is perhaps not surprising that our model outperforms by a wide margin semantic role labelers that rely heavily on syntactic parsers (Roth and Lapata, 2016; Marcheggiani and Titov, 2017). Their performance degrades considerably on out-of-domain data and the syntactic trees they produce are noisy, compromising the accuracy of the SRL systems that rely on them. Our model only makes use of features extracted from hidden layers and the weighted sum of output embeddings, rather than the output of any parser, and as a result is less brittle in this setting.

In Table 4 we report results from additional experiments on Chinese, German, and Spanish. Although we have not performed detailed

Chinese	P	R	F
Björkelund et al. (2010)	82.4	75.1	78.6
Roth and Lapata (2016)	83.2	75.9	79.4
Marcheggiani and Titov (2017)	84.6	80.4	82.5
He et al. (2018)	84.2	81.5	82.8
Ours	85.5	81.8	**83.6**
German	P	R	F
Björkelund et al. (2010)	81.2	78.3	79.7
Roth and Lapata (2016)	81.8	78.5	80.1
Ours	83.9	81.5	**82.7**
Spanish	P	R	F
Björkelund et al. (2010)	78.9	74.3	76.5
Roth and Lapata (2016)	83.2	77.4	80.2
Marcheggiani et al. (2017)	81.4	79.3	80.3
Ours	83.1	80.5	**81.8**

Table 4: Results on the CoNLL-2009 test sets for Chinese, German, and Spanish.

parameter selection in these languages (i.e., we used the same parameters as in English), our model achieves state-of-the-art performance across languages. Note that ELMo character embeddings are not available in Chinese and as a result differences in performance between our model and He et al. (2018) are more noticeable compared with English (our system outperforms theirs by 0.5 percentage point F_1). For German and Spanish, our model also achieves the best overall F_1-scores of 82.7% and 81.8%, respectively.

3.2 Ablation Studies and Analysis

In order to evaluate the contribution of various model components, we performed a series of ablation studies on the English development set without predicate disambiguation. We performed ablation studies without ELMo embeddings, as they could introduce external syntactic and semantic information, potentially obscuring any conclusions about the behavior of our own model.

Our ablation experiments are summarized in Table 5. The first block shows the performance of the full model. In the second block, we focus on the effect of different kinds of syntactic representations. First, we examined whether it is advantageous to share word embeddings between the semantic role labeler and the dependency extractor. We observed that a version of the model that updates pre-trained word embeddings separately performs slightly worse. Second, we observe a 0.8% drop in F_1 when not using the representations

System	P	R	F
Ours	86.6	84.8	85.7
w/o sharing word embeddings	86.1	84.7	85.4
w/o hidden-layer representation	86.0	83.9	84.9
w/o output embeddings	86.6	84.4	85.5
w/o multi-task learning	85.8	84.2	84.9
with full parser	86.3	85.0	85.6
w/o joint training	85.9	84.8	85.3

Table 5: Ablation results on the CoNLL-2009 English development set.

of the hidden states in the dependency extractor. The result indicates that features captured by hidden layers for the dependency prediction task are also helpful in semantic role labeling. Third, we see that not directly using the results of the dependency extractor slightly hurts SRL performance (we observe a 0.2 percentage point drop in F_1). This is not surprising as the semantic role labeler and dependency information extractor are trained simultaneously. At the beginning of the training process the performance of the extractor is low, so the semantic role labeler gradually learns how to utilize noisy label embeddings, instead of relying on the accuracy of extractor. This makes our model more robust in situations where the dependency extractor cannot achieve high performance, and also explains why our model performs better on the out-of-domain test set compared with other systems relying on parsers.

In the third block of Table 5, we first verify whether multi-task learning is critical to our SRL task by removing the term $(\mathcal{L}_{\mathcal{LBL}} + \mathcal{L}_{\mathcal{LNK}})$ from the training objective (see Equation (6)) and observe a 0.8 percentage point drop in F_1. In Figure 5 we compare the full model with multi-task learning against a model trained only for semantic role labeling (*SRL only*) in more detail. We group the semantic roles assigned by the two models (our full model vs. *SRL only*) by their dependency labels. As can be seen, the full model outperforms *SRL-only* on most dependency labels except OPRD and TMP, which account only for 3% of semantic roles. We observe noticeable gains for semantic roles with dependency labels NMOD, OBJ, SBJ, and ADV, which appear relatively frequently in the development set. In Table 6, we present model performance for verbal and nominal predicates, and again compare the results of the full

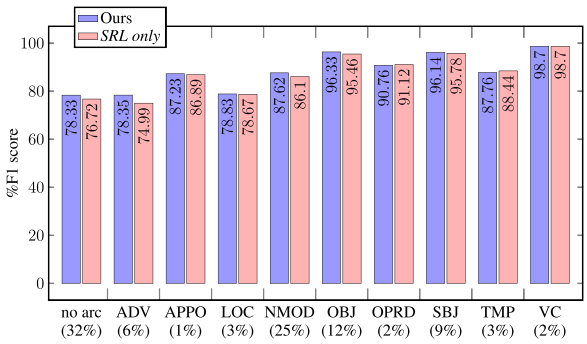

Figure 5: Semantic role labeling performance on the English CoNLL-2009 development set; roles are grouped into corresponding dependency relations whose proportional frequencies shown in parentheses (x-axis).

Verbal	Ours	*SRL only*	Frequency(%)
A0	92.9	92.2	15%
A1	93.5	92.8	21%
A2	84.3	82.8	5%
AM-*	80.8	80.1	16%
All	89.1	88.3	61%
Nominal	**Ours**	***SRL only***	**Frequency(%)**
A0	84.4	83.1	10%
A1	87.8	86.3	16%
A2	82.7	81.5	7%
AM-*	77.0	75.7	5%
All	84.5	83.2	39%

Table 6: F_1 results on the English test set broken down into verbal and nominal predicates.

model against an *SRL only* model. Both models are worse at predicting the semantic roles of nominal predicates (compared with verbal ones). However, our model is generally more accurate, especially for nominal predicates, bringing an overall improvement of 1.3 percentage point in F_1.

We next substituted the dependency extractor with a full parser, specifically, the graph-based neural model of Kiperwasser and Goldberg (2016). The parser, enhanced model achieves a performance of 85.6% in F_1, which is quite close to the model relying on the dependency extractor (see row ''with full parser'' in Table 5). This indicates that we are able to capture most of the information contained in a syntactic parser without any overhead incurred by full-blown parsing. We observed that using a full parser leads to a 0.2 percentage point F_1 increase in recall, but at the expense of precision which drops by 0.3 percentage point. As shown in Figure 5, approximately 32% of the arguments in the English development set are not directly linked to the predicate (see no arc bar). Long-range dependencies often pose problems for SRL models; in fact, special networks like GCN (Marcheggiani and Titov, 2017) and *PathLSTM* (Roth and Lapata, 2016) have been proposed to explicitly percolate information from each word in the sentence to its syntactic neighbors. However, *PathLSTM* performs worse than a vanilla BiLSTM model in the case of long-distance arguments, and the performance of an SRL model augmented with GCN also decreases when more than one GCN layer is stacked. One of the disadvantages of using an external parser are errors that then propagate through paths in the tree.

In our model, a word's dependency information solely relates to the predicate under consideration, which renders the semantic role labeler aware of the overall dependency structure of the input sentence without, however, propagating errors to other words. Although the dependency information extractor is trained to recognize arcs pertaining to the predicate, its hidden layers still capture syntactic features for long-distance arguments and share them with the semantic role labeler. As shown in the first bar of Figure 5, arguments not directly linked to the predicate are identified more accurately with the full model (F_1 improves by approximately 2 percentage point).

Finally, instead of building a dependency information extractor, we simply took the one-best outputs of the trained full parser and directly used them as word representations (i.e., replacing the e_{link} and e_{label}). This means that the full parser is pre-trained and its parameters will not be updated during the training process for SRL. We see (row ''w/o joint training'' in Table 5) that compared with the model using a full parser, removing joint training further hurts SRL performance (0.3 percentage point drop in F_1).

3.3 Dependency Annotations

Although our model does not rely on an external parser for providing information pertaining to dependency relations, it nevertheless requires gold standard dependency annotations for training the dependency extractor component (i.e., dependency label and link prediction). As manual annotations are expensive and not always available, we also examined whether it is possible to obtain competitive performance with fewer annotations.

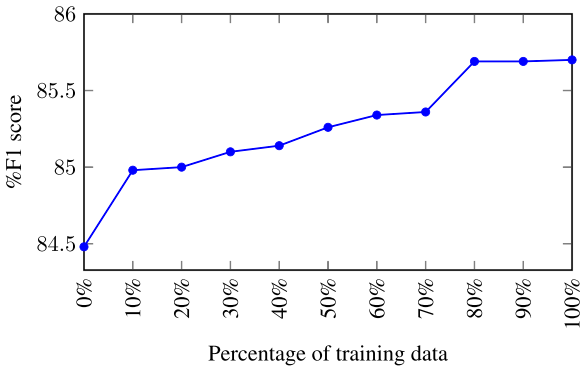

Figure 6: SRL performance on the English CoNLL-2009 development set when different proportions of dependency annotations are used for training.

Figure 6 shows how F_1 varies when the full model is trained on increasing amounts of dependency annotations. Like our previous ablation studies, we do not perform predicate disambiguation and do not use character embeddings in these experiments. We randomly choose a subset of training samples (10%, 20% etc.) with dependency annotations, and if the input sample is in the subset, we update model parameters during training according to the combined loss of the SRL and auxiliary tasks, otherwise parameters are updated for the SRL task only.

It is obvious from Figure 6 that the performance of our model increases gradually with more annotations. Interestingly, we observe a large jump in performance with only 10% of the available dependency annotations (F_1 improves from 84.5% to 85%). The model's performance becomes competitive when 80% of the annotations are used, and remains stable when more annotations are provided. In general, these results suggest that our model can also work effectively when a small number of gold dependency labels are given as a supervision signal to the dependency information extractor.

4 Related Work

Our model resonates with the recent trend of developing neural network models for semantic role labeling. It also agrees with previous work in devising ways to better take advantage of syntactic information for the SRL task within a relatively simple modeling framework based on bi-directional LSTMs (Marcheggiani et al., 2017). Previous proposals for incorporating syntactic information include the use of low-rank tensor factorizations (Lei et al., 2015), convolutional and time-domain neural networks (Foland and Martin, 2015), jointly embedded arguments and semantic roles in a shared vector space (FitzGerald et al., 2015), learning representations of shortest dependency paths between a predicate and its potential arguments (Roth and Lapata, 2016), encoding sentences with graph convolutional networks (Marcheggiani and Titov, 2017), constrained decoding (He et al., 2017), and argument pruning (He et al., 2018). In contrast to these approaches, we do not use an external dependency parser but rather incorporate syntactic information as part of the model's learning objective. Aside from assigning semantic roles, our model performs two auxiliary tasks (dependency and link type prediction), thereby learning syntactic information specific to the SRL task.

Multi-task learning (MTL; Caruana, 1993) has been a popular approach for various NLP tasks, starting with Collobert et al. (2011a) who propose a multi-task model for POS-tagging, chunking, named entity recognition, and SRL. Søgaard and Goldberg (2016) train a multi-task model for POS-tagging, syntactic chunking, and combinatory categorical grammar supertagging, while Hashimoto et al. (2017) introduce a joint many-task model together with a strategy for successively growing its depth to solve increasingly complex tasks. Zhang and Weiss (2016) propose stack-propagation using a continuous and differentiable link between POS tagging and dependency parsing, in which POS tags are utilized as a regularizer of learned representations for parsing. MTL has also been applied to semantic dependency parsing (Peng et al., 2017; Swayamdipta et al., 2017) and semantic role labeling. Strubell et al. (2018) present an end-to-end SRL model that is trained to jointly predict parts of speech and predicates, perform parsing, and attend to syntactic parse parents, while assigning semantic role labels.

Most recent MTL models (Bingel and Søgaard, 2017; Hashimoto et al., 2017) use different layers for multiple tasks with different datasets, separately optimizing each task at each epoch. In our case, the SRL task and the two auxiliary tasks share the same input, and as a result optimization for all three tasks takes place simultaneously which is more efficient. Also, in terms of model architecture, information from the auxiliary tasks is not incorporated by simply stacking layers on top of each other but rather is explored more directly by serving as input to the SRL model

itself. Like Strubell et al. (2018), we resort to multi-task learning in order to make use of linguistic information for semantic role labeling as effectively as possible. Our model is simpler in eschewing the training of a parser; it also does not predict part of speech tags or predicates, although such auxiliary tasks could be incorporated in the future. We introduce novel auxiliary tasks such as predicting the dependency label of a word and whether there exists an arc linking it to the predicate and show that they improve SRL performance for English *and* other languages.

5 Conclusions

In this paper, we proposed a multi-task model that learns dependency aware representations for semantic role labeling without using any external parser. Experimental results across languages have shown improvements over competitive baselines and state-of-the-art systems. Through several ablation studies we have also confirmed that hidden-layer representations, pre-trained word embeddings, and label embeddings all contribute in improving the performance of our SRL model. Although the dependency extractor takes a rather local view of the sentence, concentrating on the predicate and closely related neighbors, more global syntactic information is nevertheless implicitly captured. Even when dependency annotations are sparse, our model is able to encapsulate syntactic information and improve upon a syntax agnostic variant.

Directions for future work are many and varied. We would like to improve our multi-task model by determining the value of α (i.e., the loss weight for the two auxiliary tasks) dynamically. This would allow us to optimize performance for the main and auxiliary tasks at the same time. Our experiments in this work have focused exclusively on dependency-based formalisms for representing semantic predicate-argument structures (as operationalized in the CoNLL-2008 shared task). An interesting question is whether our model would work equally well for semantic role representations based on constituents (i.e., phrases or spans) such as those annotated in the CoNLL-2005 shared task (Carreras and Màrquez, 2005) or OntoNotes (Pradhan et al., 2013). Addressing this question would also allow direct comparisons with recently proposed span-based models (He et al., 2017; Strubell et al., 2018). Finally, a more ambitious goal would be to learn a semantic role labeler in a weakly supervised setting where only annotations for dependency labels are available.

Acknowledgments

We thank the anonymous reviewers for their feedback and the action editor Alessandro Moschitti for his comments. We gratefully acknowledge the support of the European Research Council (award number 681760, ‘‘Translating Multiple Modalities into Text’’).

References

Wilker Aziz, Miguel Rios, and Lucia Specia. 2011. Shallow semantic trees for SMT. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 316–322. Edinburgh.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*. San Diego, CA.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169.

Johannes Bjerva, Barbara Plank, and Johan Bos. 2016. Semantic tagging with deep residual networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3531–3541. Osaka.

Anders Björkelund, Bernd Bohnet, Love Hafdell, and Pierre Nugues. 2010. A high-performance syntactic and semantic dependency parser. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, pages 33–36.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of*

the Association for Computational Linguistics, 5:135–146.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164. Ann Arbor, MI.

Richard Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the 10th International Conference on Machine Learning*, pages 41–48.

Hao Cheng, Hao Fang, and Mari Ostendorf. 2015. Open-domain name error detection using a multi-task RNN. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 737–746. Lisbon.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the 6th International Conference on Konwledge Capture*, pages 113–119. Banff.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011a. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Ronan Collobert, Jason Weston, Michael Karlen, Léon Bottou, Koray Kavukcuoglu, and Pavel Kuksa. 2011b. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems 28*, pages 2224–2232.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 960–970. Lisbon.

William Foland and James Martin. 2015. Dependency-based semantic role labeling using convolutional neural networks. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 279–288.

Nelson Francis and Henry Kucera. 1979, Brown corpus manual. Technical report, Department of Linguistics, Brown Unviersity, Providence, RI.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CONLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18. Boulder, CO.

Eva Hajičová, Zdeněk Kirschner, and Petr Sgall. 1999. A manual for analytic layer annotation of the Prague dependency treebank (English translation), ÚFAL MFF UK, Prague, Czech Republic.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pages 473–483. Vancouver.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.

Richard Johansson and Pierre Nugues. 2008. The effect of syntactic representation on semantic role labeling. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 393–400. Manchester.

Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. 2016. Molecular graph convolutions: Moving beyond fingerprints. *Journal of Computer-Aided Molecular design*, 30(8):595–608.

Atif Khan, Naomie Salim, and Yogan Jaya Kumar. 2015. A framework for multi-document abstractive summarization based on semantic role labelling. *Applied Soft Computing*, 30:737–747.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint, arXiv:1412.6980*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*. Toulon.

Joel Lang and Mirella Lapata. 2010. Unsupervised induction of semantic roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947.

Tao Lei, Yuan Zhang, Lluís Màrquez, Alessandro Moschitti, and Regina Barzilay. 2015. High-order low-rank tensors for semantic role labeling. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1150–1160. Denver, CO.

Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304, Denver, CO.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *Proceedings of the International Conference on Learning Representations*. San Juan, PR.

Diego Marcheggiani, Joost Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of the the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*. New Orleans, LA.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 411–420. Vancouver.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515. Copenhagen.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013, Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

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

Mitchell P. Parcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2048.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Barbara Plank. 2016. Keystroke dynamics as signal for shallow syntactic parsing. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, pages 609–619. Osaka.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using Ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152. Sofia.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202. Berlin.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235. Berlin.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CONLL 2008 Shared Task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 159–177.

Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A Smith. 2017. Frame-semantic parsing with softmax-margin segmental RNNS and a syntactic scaffold. *arXiv preprint, arXiv:1706.09528*.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, pages 2773–2781. Montreal.

Nanwen Xue, Fei Xia, Fu Dong Chiou, and Martha Palmer. 2005. The Penn Chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese treebank. *Natural Language Engineering*, 15(1):143–172.

Yuan Zhang and David Weiss. 2016. Stack-propagation: Improved representation learning for syntax. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1557–1566.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137. Beijing.