A Unified Syntax-aware Framework for Semantic Role Labeling

Zuchao Li[1,2,*], **Shexia He**[1,2,*], **Jiaxun Cai**[1,2], **Zhuosheng Zhang**[1,2], **Hai Zhao**[1,2,†], **Gongshen Liu**[3], **Linlin Li**[4], **Luo Si**[4]

[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]School of Cyber Security, Shanghai Jiao Tong University, China
[4]Alibaba Group, Hangzhou, China
{charlee,heshexia,caijiaxun,zhangzs}@sjtu.edu.cn,
zhaohai@cs.sjtu.edu.cn, lgshen@sjtu.edu.cn,
{linyan.lll,luo.si}@alibaba-inc.com

Abstract

Semantic role labeling (SRL) aims to recognize the predicate-argument structure of a sentence. Syntactic information has been paid a great attention over the role of enhancing SRL. However, the latest advance shows that syntax would not be so important for SRL with the emerging much smaller gap between syntax-aware and syntax-agnostic SRL. To comprehensively explore the role of syntax for SRL task, we extend existing models and propose a unified framework to investigate more effective and more diverse ways of incorporating syntax into sequential neural networks. Exploring the effect of syntactic input quality on SRL performance, we confirm that high-quality syntactic parse could still effectively enhance syntactically-driven SRL. Using empirically optimized integration strategy, we even enlarge the gap between syntax-aware and syntax-agnostic SRL. Our framework achieves state-of-the-art results on CoNLL-2009 benchmarks both for English and Chinese, substantially outperforming all previous models.

1 Introduction

The purpose of semantic role labeling (SRL) is to derive the predicate-argument structure of each predicate in a sentence. A popular formalism to represent the semantic predicate-argument structure is based on dependencies, namely dependency SRL, which annotates the heads of arguments rather than phrasal arguments. Given a sentence (in Figure 1), SRL is generally decomposed into multiple subtasks in pipeline framework, consisting of predicate identification (*makes*), predicate disambiguation (*make.02*), argument identification (e.g., *Someone*) and argument classification (*Someone* is A0 for the predicate *makes*). SRL is beneficial to a wide range of natural language processing (NLP) tasks, including machine translation (Shi et al., 2016) and question answering (Berant et al., 2013; Yih et al., 2016).

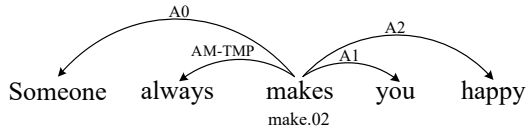

Figure 1: An example of dependency-based SRL.

Most traditional SRL methods rely heavily on feature templates that struggle to capture sufficient discriminative information, while neural models are capable of extracting features automatically. In particular, recent works (Zhou and Xu, 2015; He et al., 2017; Marcheggiani et al., 2017) propose syntax-agnostic models for SRL and achieve favorable results, which seems to be in conflict with the belief that syntactic information is an absolutely necessary prerequisite for high-performance SRL (Gildea and Palmer, 2002). Despite the success of these models, the main reasons for putting syntax aside are two-fold. First, it is still challenging to effectively incorporate syntactic information into neural SRL models, due to the sophisticated tree structure of syntactic relation. Second, the syntactic parsers are unreliable on account of the risk of erroneous syntactic input, which may lead to error propagation and an unsatisfactory SRL performance.

However, syntactic information is considered closely related to semantic relation and plays an essential role in SRL task (Punyakanok et al., 2008). Recently, Marcheggiani and Titov (2017)

[*] These authors made equal contribution.[†] Corresponding author. This paper was partially supported by National Key Research and Development Program of China (No. 2017YFB0304100), National Natural Science Foundation of China (No. 61672343 and No. 61733011), Key Project of National Society Science Foundation of China (No. 15-ZDA041), The Art and Science Interdisciplinary Funds of Shanghai Jiao Tong University (No. 14JCRZ04) and the joint research project with Youtu Lab of Tencent.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 2401–2411
Brussels, Belgium, October 31 - November 4, 2018.

proposed a syntactic graph convolutional networks (GCNs) based SRL model and further improved the SRL performance with relatively better syntactic parser as input. Since syntax can provide rich structure and information for SRL, we seek to effectively model complex syntactic tree structure for incorporating syntax into neural SRL.

In this paper, we present a general framework[1] for SRL, which enables us to integrate syntax into SRL in diverse ways. Following Marcheggiani and Titov (2017), we focus on argument labeling and formulate SRL as sequence labeling problem. However, we differ by (1) leveraging enhanced word representation, (2) applying recent advances in recurrent neural networks (RNNs), such as highway connections (Srivastava et al., 2015), (3) using deep encoder with residual connections (He et al., 2016), (4) further extending Syntax Aware Long Short-Term Memory (SA-LSTM) (Qian et al., 2017) for SRL, and (5) introducing the Tree-Structured Long Short-Term Memory (Tree-LSTM) (Tai et al., 2015) to model syntactic information for SRL.

In addition, as pointed out by He et al. (2017) for span SRL, the worse syntactic input will hurt performance if the syntactically-driven SRL model trusts syntactic information too much, and high-quality syntax can still make a large impact on SRL, which motivates us to investigate the effect of syntactic quality on dependency SRL. In summary, our major contributions are as follows:

- We propose a unified neural framework for dependency SRL to more effectively integrate syntactic information with multiple methods.
- Our SRL framework incorporated with syntax achieves the new state-of-the-art results on both English and Chinese CoNLL-2009 benchmarks.
- We explore the impact of different quality of syntactic input on SRL performance, showing that high quality syntactic parse may indeed improve syntax-aware SRL.

2 A Unified SRL Framework

In order to explore the effectiveness of the syntactic feature from various perspectives, we propose a unified neural framework that is capable of optionally accommodating various types of syntactic encoders for syntax-based SRL.

Since the CoNLL-2009 shared task (Hajič et al., 2009) have beforehand indicated the predicate positions, we need to identify and label all arguments for each predicate, which is a typical sequence tagging problem. In this work, we construct a general SRL framework for argument labeling. As shown in Figure 2, our SRL framework includes three main modules, (1) BiLSTM encoder that directly takes sequential inputs, (2) MLP with highway connections for softmax output layer, and (3) an optional syntactic encoder that receives the outputs of the BiLSTM encoder and then let its own outputs integrate with the BiLSTM outputs through residual connections.

Note that when the syntactic encoder is completely removed, MLP only takes inputs directly from the BiLSTM encoder, which let our framework become a syntax-agnostic labeler.

2.1 Sentence Encoder

Word representation Given a sentence and known predicate, we consider predicate-specific word representation, following previous work (Marcheggiani and Titov, 2017). Specifically, each word embedding representation e_i of input sentence is the concatenation of several features, a randomly initialized word embedding e_i^r, a pretrained word embedding e_i^p, a randomly initialized lemma embedding e_i^l, a randomly initialized POS tag embedding e_i^{pos}, and a predicate-specific feature e_i^f, which is a binary flag set 0 or 1 indicating whether the current word is the given predicate.

To further enhance the word representation, we leverage an external embedding ELMo (Embeddings from Language Models) proposed by Peters et al. (2018). ELMo is obtained by deep bidirectional language model that takes characters as input, enriching subword information and contextual information, which has expressive representation power. Eventually, the resulting word representation is concatenated as $e_i = [e_i^r, e_i^p, e_i^l, e_i^{pos}, e_i^f, \text{ELMo}_i]$.

BiLSTM encoder We use bi-directional Long Short-term Memory neural network (BiLSTM) (Hochreiter and Schmidhuber, 1997) as the sentence encoder to model sequential inputs. Given an input sequence $(e_1, \ldots, e_n)$, the BiLSTM processes these embedding vectors sequentially from both directions to obtain two separated hidden states, $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$ respectively. By concatenating the two states, we get a contextual representation $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$, which will be taken by the next

[1] Our code is available here: https://github.com/bcmi220/unified_syn_srl.

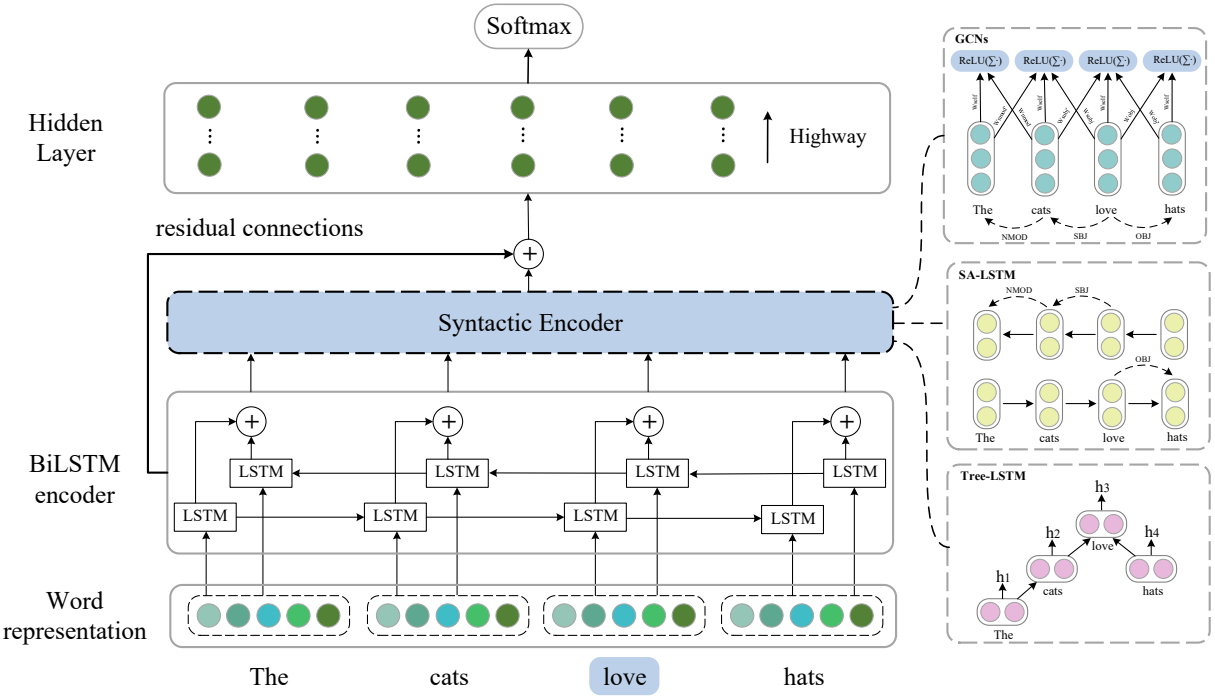

Figure 2: The unified syntax-based SRL framework

BiLSTM layer as input. In this work, we stack four layers of BiLSTM.

2.2 Role Labeler

We adopt a Multi-Layer Perceptron (MLP) with highway connections (Srivastava et al., 2015) on the top of our deep encoder, which takesthe concatenated representation as input. The MLP consists of 10 layers and we employ *ReLU* activations for the hidden layer. To get the final predicted semantic roles, we use a softmax layer over the outputs to maximize the likelihood of labels. The MLP part takes inputs from both the BiLSTM encoder and syntactic encoder, which are joint through a residual connection (He et al., 2016) as shown in Figure 2. It is worth noting that our deep encoder is different from the one of Marcheggiani and Titov (2017), which directly applies a softmax transformation over the syntactic representation and predicts the role label for each word. That is, their syntactic encoder outputs are directly taken as the input of hidden layer.

3 Syntactic Encoder

To integrate the syntactic information into sequential neural networks, we employ a syntactic encoder on top of the BiLSTM encoder.

Specifically, given a syntactic dependency tree T, for each node n_k in T, let $C(k)$ denote the syntactic children set of n_k, $H(k)$ denote the syntactic head of n_k, and $L(k,\cdot)$ be the dependency relation between node n_k and those have a direct arc from or to n_k. Then we formulate the syntactic encoder as a transformation f^τ over the node n_k, which may take some of $C(k)$, $H(k)$, or $L(k,\cdot)$ as input, and compute a syntactic representation v_k for node n_k, namely, $v_k = f^\tau(C(k), H(k), L(k,\cdot), x_k)$. When not otherwise specified, x_k denotes the input feature representation of n_k which may be either the word representation e_k or the output of BiLSTM h_k, σ denotes the logistic sigmoid function, and $\odot$ denotes the element-wise multiplication.

In practice, the transformation f^τ can be any syntax encoding method. In this paper, we will consider three types of syntactic encoders, syntactic graph convolutional network (Syntactic GCN) (in Section 3.1), syntax aware LSTM (SA-LSTM) (in Section 3.2), tree-structured LSTM (Tree-LSTM) (in Section 3.3). Then, we will provide a brief introduction in subsequent subsections.

3.1 Syntactic GCN

GCN (Kipf and Welling, 2017) is proposed to induce the representations of nodes in a graph based on the properties of their neighbors. Given its effectiveness, Marcheggiani and Titov (2017) in-

troduce a generalized version for the SRL task, namely syntactic GCN, and shows that syntactic GCN is effective in incorporating syntactic information into neural models.

Syntactic GCN captures syntactic information flows in two directions, one from heads to dependents (along), the other from dependents to heads (opposite). Besides, it also models the information flows from a node to itself, namely, it assumes that a syntactic graph contains self-loop for each node. Thus, the syntactic GCN transformation of a node n_k is defined on its neighborhood $N(k) = C(k) \cup H(k) \cup \{n_k\}$. For each edge connects n_k and its neighbor n_j, we can compute a vector representation for it,

$$u_{k,j} = W^{dir(k,j)} x_j + b^{L(k,j)},$$

where $dir(k, j)$ denotes the direction type (along, opposite or self-loop) of the edge from n_k to n_j, $W^{dir(k,j)}$ is direction type specific parameter, $b^{L(k,j)}$ is label specific parameter. Considering that syntactic information from all the neighboring nodes may make different contribution to semantic role labeling, syntactic GCN introduces an additional edge-wise gating for each node pair (n_k, n_j) as

$$g_{k,j} = \sigma(W_g^{dir(k,j)} x_k + b_g^{L(k,j)}).$$

The syntactic representation v_k for a node n_k can be then computed as:

$$v_k = \mathit{ReLU}(\sum_{j \in N(k)} g_{k,j} \odot u_{k,j}).$$

3.2 SA-LSTM

SA-LSTM (Qian et al., 2017) is an extension of the standard BiLSTM architecture, which aims to simultaneously encode the syntactic and contextual information for a given word as shown in Figure 2. On one hand, the SA-LSTM calculates the hidden state in sequence timestep order like the standard LSTM,

$$\begin{aligned} i_g &= \sigma(W^{(i)} x_k + U^{(i)} h_{k-1} + b^{(i)}), \\ f_g &= \sigma(W^{(f)} x_k + U^{(f)} h_{k-1} + b^{(f)}), \\ o_g &= \sigma(W^{(o)} x_k + U^{(o)} h_{k-1} + b^{(o)}), \\ u &= f(W^{(u)} x_k + U^{(u)} h_{k-1} + b^{(u)}), \\ c_k &= i_g \odot u + f_g \odot c_{k-1}. \end{aligned}$$

On the other hand, it further incorporates the syntactic information into the representation of each word by introducing an additional gate,

$$\begin{aligned} s_g &= \sigma(W^{(s)} x_k + U^{(s)} h_{k-1} + b^{(s)}), \\ h_k &= o_g \odot f(c_k) + s_g \odot \tilde{h}_k. \end{aligned}$$

where $\tilde{h}_k = f(\sum_{t_j < t_k} \alpha_j \times h_j)$ is the weighted sum of all hidden state vectors h_j which come from previous node (word) n_j, the weight factor α_j is actually a trainable weight related to the dependency relation $L(k, \cdot)$ when there exists a directed edge from n_j to n_k.

Note that $\tilde{h}_k$ is always the hidden state vector of the syntactic head of n_k according to the definition of α_j. Since a word will be assigned a single syntactic head, such a strict constraint prevents the SA-LSTM from incorporating complex syntactic structures. Inspire by the idea of GCN, we relax the directed constraint of α_j, whenever there is an edge between n_j and n_k.

After the SA-LSTM transformation, the outputs of the SA-LSTM layer from both directions are concatenated and taken as the syntactic representation of each word n_k, i.e., $v_k = [\overrightarrow{h_k}, \overleftarrow{h_k}]$. Different from the syntactic GCN, SA-LSTM encoding both syntactic and contextual information in a single vector v_k.

3.3 Tree-LSTM

Tree-LSTM (Tai et al., 2015) can be considered as an extension of the standard LSTM, which aims to model the tree-structured topologies. At each timestep, it composes an input vector and the hidden states from arbitrarily many child units. Specifically, the main difference between Tree-LSTM unit and the standard one is that the memory cell updating and the calculation of gating vectors are depended on multiple child units. A Tree-LSTM unit can be connected to arbitrary number of child units and assigns a single forget gate for each child unit. This provides Tree-LSTM the flexibility to incorporate or drop the information from each child unit.

Given a syntactic tree, the Tree-LSTM transformation is defined on node n_k and its children set $C(k)$, which can be formulated as follows (Tai

et al., 2015):

$$\tilde{h}_k = \sum_{j \in C(k)} h_k, \quad (1)$$

$$i_g = \sigma(W^{(i)} x_k + U^{(i)} \tilde{h}_k + b^{(i)}),$$
$$f_g^{k,j} = \sigma(W^{(f)} x_k + U^{(f)} h_j + b^{(f)}), \quad (2)$$
$$o_g = \sigma(W^{(o)} x_k + U^{(o)} \tilde{h}_k + b^{(o)}),$$
$$u = \tanh(W^{(u)} x_k + U^{(u)} \tilde{h}_k + b^{(u)}),$$
$$c_k = i_g \odot u + \sum_{j \in C(k)} f_g^{k,j} \odot c_j,$$
$$h_k = o_g \odot \tanh(c_k).$$

where $j \in C(k)$, h_j is the hidden state of the j-th child node, c_k is the memory cell of the head node k, and h_k is the hidden state of node k. Note that in Eq.(2), a single forget gate $f_g^{k,j}$ is computed for each hidden state h_j.

However, the primitive form of Tree-LSTM does not take the dependency relations into consideration. Given the importance of dependency relations in SRL task, we further extend the Tree-LSTM by adding an additional gate r_g and reformulate the Eq. (1),

$$r_g^{k,j} = \sigma(W^{(r)} x_k + U^{(r)} h_j + b^{L(k,j)}),$$
$$\tilde{h}_k = \sum_{j \in C(k)} r_g^{k,j} \odot h_j.$$

where $b^{L(k,j)}$ is a relation label specific bias term. After the Tree-LSTM transformation, the hidden state of each node in dependency tree is taken as its syntactic representation, i.e., $v_k = h_k$.

4 Experiments

We evaluate our models performance of syntactic GCN (henceforth Syn-GCN), SA-LSTM and Tree-LSTM on CoNLL-2009 datasets both for English and Chinese with standard training, development and test splits. For predicate disambiguation, we follow previous work (Marcheggiani and Titov, 2017), using the off-the-shelf disambiguator from Roth and Lapata (2016). For syntactic dependency tree, we parse the corpus with Biaffine Parser (Dozat and Manning, 2017).

4.1 Experimental Settings

In our experiments, the pre-trained word embeddings for English are 100-dimensional GloVe vectors (Pennington et al., 2014). For Chinese, we exploit Wikipedia documents to train the same dimensional Word2Vec embeddings (Mikolov et al., 2013). All other vectors are randomly initialized, the dimension of lemma embeddings is 100, and the dimension of POS tag embedding is 32. In addition, we use 300-dimensional ELMo embedding for English[2].

During training, we use the categorical cross-entropy as objective, with Adam optimizer (Kingma and Ba, 2015) the learning rate 0.001, and the batch size is set to 64. The BiLSTM encoder consists of 4-layer BiLSTM with 512-dimensional hidden units. We apply dropout for BiLSTM with a 90% keeping probability between time-steps and layers. We train models for a maximum of 20 epochs and obtain the nearly best model based on English development results.

System	P	R	F_1
Local model			
Lei et al. (2015)	–	–	86.6
FitzGerald et al. (2015)	–	–	86.7
Roth and Lapata (2016)	88.1	85.3	86.7
Marcheggiani et al. (2017)	88.7	86.8	87.7
Marcheggiani and Titov (2017)	89.1	86.8	88.0
He et al. (2018)	89.7	89.3	89.5
Cai et al. (2018)	89.9	89.2	89.6
Ours (Syn-GCN)	90.3	**89.3**	**89.8**
Ours (SA-LSTM)	**90.8**	88.6	89.7
Ours (Tree-LSTM)	90.0	88.8	89.4
Global model			
Björkelund et al. (2010)	88.6	85.2	86.9
FitzGerald et al. (2015)	–	–	87.3
Roth and Lapata (2016)	**90.0**	**85.5**	**87.7**
Ensemble model			
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 1: Results on the English in-domain test set.

4.2 Results

We compare our models of Syn-GCN, SA-LSTM and Tree-LSTM with previous approaches for dependency SRL on both English and Chinese. Noteworthily, our model is local (argument identification and classification decisions are conditionally independent) and single without reranking, which neither includes global inference nor com-

[2]For Chinese, we do not use pre-trained ELMo whose weights are only available for English.

System	P	R	F_1
Local model			
Zhao et al. (2009a)	80.4	75.2	77.7
Marcheggiani et al. (2017)	83.4	79.1	81.2
Marcheggiani and Titov (2017)	84.6	80.4	82.5
He et al. (2018)	84.2	81.5	82.8
Cai et al. (2018)	84.7	**84.0**	**84.3**
Ours (Syn-GCN)	84.8	81.2	83.0
Ours (SA-LSTM)	**85.2**	80.5	82.8
Ours (Tree-LSTM)	84.5	80.7	82.6
Global model			
Björkelund et al. (2009)	82.4	75.1	78.6
Roth and Lapata (2016)	**83.2**	**75.9**	**79.4**

Table 2: Results on the Chinese test set.

bines multiple models. The experimental results on the in-domain English and Chinese test sets are summarized in Tables 1 and 2, respectively.

For English, our models of Syn-GCN, SA-LSTM and Tree-LSTM overwhelmingly surpass most previously published single models, achieving state-of-the-art results of 89.8%, 89.7% and 89.4% in F_1 scores respectively. In comparison to ensemble models, our Syn-GCN even performs better than the previous model (Marcheggiani and Titov, 2017) with a margin of 0.7% F_1.

From Table 1, we also see that our Syn-GCN model provides the best recall and F_1 score, while our SA-LSTM model yields the competitive performance with higher precision at the expense of recall, which shows that SA-LSTM is better at classifying arguments. Overall, the Tree-LSTM gives slightly weaker performance, which may be attributed to tree-structured network topology. More specifically, Tree-LSTM only considers information from arbitrary child units so that each node lacks of the information from parent. However, our Syn-GCN and SA-LSTM combine bidirectional information, both head-to-dependent and dependent-to-head.

For Chinese (Table 2), even though we use the same parameters as for English, our models are still comparable with the best reported results.

Table 3 presents the results on English out-of-domain test set. Our models outperform the highest records achieved by He et al. (2018), with absolute improvements of 0.2-0.5% in F_1 scores. These favorable results on both in-domain and out-of-domain data demonstrate the effectiveness and robustness of our proposed unified framework.

System	P	R	F_1
Local model			
Lei et al. (2015)	–	–	75.6
FitzGerald et al. (2015)	–	–	75.2
Roth and Lapata (2016)	76.9	73.8	75.3
Marcheggiani et al. (2017)	79.4	76.2	77.7
Marcheggiani and Titov (2017)	78.5	75.9	77.2
He et al. (2018)	81.9	76.9	79.3
Cai et al. (2018)	79.8	78.3	79.0
Ours (Syn-GCN)	80.6	**79.0**	**79.8**
Ours (SA-LSTM)	**81.0**	78.2	79.6
Ours (Tree-LSTM)	80.4	78.7	79.5
Global model			
Björkelund et al. (2010)	77.9	73.6	75.7
FitzGerald et al. (2015)	–	–	75.2
Roth and Lapata (2016)	**78.6**	**73.8**	**76.1**
Ensemble model			
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: Results on the English out-of-domain test set.

Our system	P	R	F_1
Syn-GCN	89.2	87.6	88.4
w/o POS tag	88.5	87.5	88.0
w/o ELMo embedding	87.7	86.7	87.2

Table 4: Ablation on the English development set.

4.3 Ablation and Analysis

To investigate the contributions of word representation and deep encoder in our method, we conduct a series of ablation studies on the English development set, unless otherwise stated.

Effect of word representation In order to better understand how the enhanced word representation influences our model performance, we train our Syn-GCN model with different settings in input word embeddings. Table 4 shows results for our system when we remove POS tag and ELMo embedding respectively. Interestingly, the impact of POS tag embedding (about 0.4% F_1) is less compared to the previous works, which allows us to build an accuracy model even when the POS tag is unavailable. We also observe that effect of ELMo embedding is somewhat surprising (1.2% F_1 performance degradation). Experimental results indicate that a combination of these features could enhance the word representation, leading to SRL performance improvement.

System	syntax-agnostic	syntax-aware	Δ F_1
M&T (2017)	87.7	88.0	0.3
He et al. (2018)	88.7	89.5	0.8
Cai et al. (2018)	89.6	89.6	$\approx$0.0
Our model	88.7	89.8	1.1

Table 5: Comparison of our Syn-GCN model with (Marcheggiani and Titov, 2017), (He et al., 2018) and (Cai et al., 2018) on the English test set. Δ F_1 shows the absolute performance gap between syntax-agnostic and syntax-aware settings.

Our system	P	R	F_1
Baseline (syntax-agnostic)	89.5	87.9	88.7
Syn-GCN	90.3	89.3	**89.8**
SA-LSTM	90.8	88.6	89.7
Tree-LSTM	90.0	88.8	89.4
Syn-GCN (M&T encoder)	89.2	88.0	88.6
SA-LSTM (M&T encoder)	89.8	88.8	**89.3**
Tree-LSTM (M&T encoder)	90.0	87.8	88.9

Table 6: Comparison of models with deep encoder and M&T encoder (Marcheggiani and Titov, 2017) on the English test set.

Effect of deep encoder Table 5 reports F_1 scores of our Syn-GCN model, Marcheggiani and Titov (2017), He et al. (2018) and Cai et al. (2018) on English test set in both syntax-agnostic and syntax-aware settings. The comparison shows that our framework is more effective for incorporating syntactic information by giving more performance improvement through introducing syntax over syntax-agnostic SRL than previous state-of-the-art systems did.

To further investigate the impact of deep encoder, we perform our Syn-GCN, SA-LSTM and Tree-LSTM models with another alternative configuration, using the same encoder as (Marcheggiani and Titov, 2017) (M&T encoder for short), which removes the residual connections from our framework. The corresponding results of our models are also summarized in Table 6 for comparison. Note that the first row is the results of our syntax-agnostic model. Surprisingly, we observe a dramatical performance decline of 1.2% F_1 for our Syn-GCN model with M&T encoder. A less significant performance loss for our SA-LSTM (-0.4%) and Tree-LSTM (-0.5%) models shows that the Syn-GCN is more sensitive to contextual information. Nevertheless, the overall results show that applying deep encoder could receive higher gains.

4.4 Syntactic Role

As mentioned before, syntactic parsers are unreliable due to the risk of erroneous syntactic input, especially on out-of-domain data. This section thus attempts to explore the impact of different quality of syntactic input on SRL performance. To this end, we further carry out experiments on English test data with different syntactic inputs based on our Syn-GCN model.

Syntactic Input

Four types of syntactic inputs are used to explore the role of syntax in our unified framework, (1) the automatically predicted parse provided by CoNLL-2009 shared task, (2) the parsing results of the CoNLL-2009 data by state-of-the-art syntactic parser, the Biaffine Parser (used in our previous experiments), (3) corresponding results from another parser, the BIST Parser (Kiperwasser and Goldberg, 2016), which is also adopted by Marcheggiani and Titov (2017), (4) the gold syntax available from the official data set.

Evaluation Metric

It is worth noting that for SRL task, the standard evaluation metric is the semantic labeled F_1 score (Sem-F_1), and we use the labeled attachment score (LAS) to quantify the quality of syntactic input. In addition, the ratio between labeled F_1 score for semantic dependencies and the LAS for syntactic dependencies (Sem-F_1/LAS) proposed by CoNLL-2008 shared task[3] (Surdeanu et al., 2008), are also given for reference. To a certain extent, the ratio Sem-F_1/LAS could normalize the semantic score relative to syntactic parse, impartially estimating the true performance of SRL, independent of the performance of the input syntactic parser.

Comparison and Discussion

Table 7 presents the comprehensive results of our Syn-GCN model on the four syntactic inputs aforementioned of different quality together with previous SRL models. A number of observations can be made from these results. First, our model gives quite stable SRL performance no matter the syntactic input quality varies in a broad range, ob-

[3]CoNLL-2008 is an English-only task, while CoNLL-2009 extends to a multilingual one. Their main difference is that predicates have been pre-identified for the latter.

System	LAS	P	R	Sem-F_1	Sem-F_1/LAS
Zhao et al. (2009c) [SRL-only]	86.0	–	–	85.4	99.3
Zhao et al. (2009a) [Joint]	89.2	–	–	86.2	96.6
Björkelund et al. (2010)	89.8	87.1	84.5	85.8	95.6
Lei et al. (2015)	90.4	–	–	86.6	95.8
Roth and Lapata (2016)	89.8	88.1	85.3	86.7	96.5
Marcheggiani and Titov (2017)	90.34*	89.1	86.8	88.0	97.41
He et al. (2018) [CoNLL-2009 predicted]	86.0	89.7	89.3	89.5	104.0
He et al. (2018) [Gold syntax]	100	91.0	89.7	90.3	90.3
Our Syn-GCN (CoNLL-2009 predicted)	86.0	90.5	88.5	89.5	104.07
Our Syn-GCN (Biaffine Parser)	90.22	90.3	89.3	89.8	99.53
Our Syn-GCN (BIST Parser)	90.05	90.3	89.1	89.7	99.61
Our Syn-GCN (Gold syntax)	100.0	91.0	90.0	90.5	90.50

Table 7: Results on English test set, in terms of labeled attachment score for syntactic dependencies (LAS), semantic precision (P), semantic recall (R), semantic labeled F_1 score (Sem-F_1), the ratio Sem-F_1/LAS. All numbers are in percent. A superscript * indicates LAS results from our personal communication with the authors.

taining overall higher scores compared to previous state-of-the-arts. Second, It is interesting to note that the Sem-F_1/LAS score of our model becomes relatively smaller as the syntactic input becomes better. Though not so surprised, these results show that our SRL component is even relatively stronger. Third, when we adopt a syntactic parser with higher parsing accuracy, our SRL system will achieve a better performance. Notably, our model yields a Sem-F_1 of 90.5% taking gold syntax as input. It suggests that high-quality syntactic parse may indeed enhance SRL, which is consistent with the conclusion in (He et al., 2017).

5 Related Work

Semantic role labeling was pioneered by Gildea and Jurafsky (2002), also known as shallow semantic parsing. In early works of SRL, considerable attention has been paid to feature engineering (Pradhan et al., 2005; Zhao and Kit, 2008; Zhao et al., 2009a,b,c; Li et al., 2009; Björkelund et al., 2009; Zhao et al., 2013). Along with the the impressive success of deep neural networks (Zhang et al., 2016; Cai and Zhao, 2016; Qin et al., 2016; Wang et al., 2016b,a; Zhang et al., 2018; Li et al., 2018; Huang et al., 2018), a series of neural SRL systems have been proposed. For instance, Foland and Martin (2015) presented a semantic role labeler using convolutional and time-domain neural networks. FitzGerald et al. (2015) exploited neural network to jointly embed arguments and semantic roles, akin to the work (Lei et al., 2015), which induced a compact feature representation applying tensor-based approach.

Recently, people have attempted to build end-to-end systems for span SRL without syntactic input (Zhou and Xu, 2015; He et al., 2017; Tan et al., 2018). Similarly, Marcheggiani et al. (2017) also proposed a syntax-agnostic model for dependency SRL and obtained favorable results. Despite the success of syntax-agnostic models, there are several works focus on leveraging the advantages of syntax. Roth and Lapata (2016) employed dependency path embedding to model syntactic information and exhibited a notable success. Marcheggiani and Titov (2017) leveraged the graph convolutional network to incorporate syntax into a neural SRL model. Qian et al. (2017) proposed SA-LSTM to model the whole tree structure of dependency relation in an architecture engineering way.

Besides, syntax encoding has also successfully promoted other NLP tasks. Tree-LSTM (Tai et al., 2015) is a variant of the standard LSTM that can encode a dependency tree with arbitrary branching factors, which has shown effectiveness on semantic relatedness and the sentiment classification tasks. In this work, we extend the Tree-LSTM with a relation specific gate and employ it to recursively encode the syntactic dependency tree for SRL. RCNN (Zhu et al., 2015) is an extension of the recursive neural network (Socher et al., 2010) which has been popularly used to encode trees with fixed branching factors. The RCNN is able to encode a tree structure with arbitrary number of factors and is useful in a re-ranking model for dependency parsing (Zhu et al., 2015).

In our experiments, we simplify and reformulate the RCNN model. However, the simplified model performs poorly on the development and the test sets. The reason might be that the RCNN model with a single global composition parameter is too simple to cover all types of syntactic relation in a dependency tree. Because of the poor performance of the modified RCNN, we do not include it in this work. Considering there might be other approach to incorporate the recursive network in SRL model, we leave it as our future work and just provide a brief discussion here.

In this work, we extend existing methods and introduce Tree-LSTM for incorporating syntax into SRL. Rather than proposing completely new model, we synthesize these techniques and present a unified framework to take genuine superiority of syntactic information.

6 Conclusion

This paper presents a unified neural framework for dependency-based SRL, effectively incorporating syntactic information by directly modeling syntax based on syntactic parse tree. Rather than proposing completely new model, we extend existing models and apply tree-structured LSTM for SRL. Our approach significantly outperforms all previous models, achieving state-of-the-art results on the CoNLL-2009 benchmarks for both English and Chinese.

Our experiments specially show that giving an enlarged performance gap from syntax-agnostic to syntax-aware setting, SRL can be further promoted with the help of deep enhanced representation and effective methods of integrating syntax. Furthermore, we explore the impact of the quality of syntactic input. The relevant results indicate that high-quality syntactic parse is more favorable to semantic role labeling.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1533–1544, Seattle, Washington, USA.

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

Anders Björkelund, Love Hafdell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 43–48, Boulder, Colorado.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for Chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 409–420.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 2753–2765.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

Nicholas FitzGerald, Oscar Tckstrm, 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 (EMNLP)*, pages 960–970.

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

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

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 239–246, Philadelphia, Pennsylvania, USA.

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, Colorado.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

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 (ACL)*, pages 473–483, Vancouver, Canada.

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)*, volume 1, pages 2061–2071.

Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yafang Huang, Zuchao Li, Zhuosheng Zhang, and Hai Zhao. 2018. Moon ime: neural-based Chinese pinyin aided input method with customizable association. *Proceedings of ACL 2018, System Demonstrations*, pages 140–145.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

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 (ICLR)*.

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 (NAACL: HLT)*, pages 1150–1160.

Junhui Li, Guodong Zhou, Hai Zhao, Qiaoming Zhu, and Peide Qian. 2009. Improving nominal srl in Chinese language with verbal srl information and automatic predicate recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1280–1288.

Zuchao Li, Jiaxun Cai, Shexia He, and Hai Zhao. 2018. Seq2seq dependency parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3203–3214.

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, Canada.

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 (EMNLP)*, pages 1506–1515, Copenhagen, Denmark.

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 (NIPS)*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

Matthew E. 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 (NAACL: HLT)*, New Orleans, Louisiana.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 581–588, Ann Arbor, Michigan.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Feng Qian, Lei Sha, Baobao Chang, Lu Chen Liu, and Ming Zhang. 2017. Syntax aware LSTM model for semantic role labeling. In *The Workshop on Structured Prediction for Natural Language Processing*, pages 27–32.

Lianhui Qin, Zhisong Zhang, and Hai Zhao. 2016. Implicit discourse relation recognition with context-aware character-enhanced embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1914–1924.

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 (ACL)*, pages 1192–1202, Berlin, Germany.

Chen Shi, Shujie Liu, Shuo Ren, Shi Feng, Mu Li, Ming Zhou, Xu Sun, and Houfeng Wang. 2016. Knowledge-based semantic embedding for machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2245–2254, Berlin, Germany.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1–9.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in neural information processing systems*, pages 2377–2385.

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 *Proceedings of the Twelfth Conference on Computational Natural Language Learning - Shared Task (CoNLL)*, pages 159–177, Manchester, England.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 1556–1566, Beijing, China.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Rui Wang, Hai Zhao, Bao-Liang Lu, Masao Utiyama, and Eiichiro Sumita. 2016a. Connecting phrase based statistical machine translation adaptation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3135–3145.

Rui Wang, Hai Zhao, Sabine Ploux, Bao-Liang Lu, and Masao Utiyama. 2016b. A bilingual graph-based semantic model for statistical machine translation. In *IJCAI*, pages 2950–2956.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 201–206, Berlin, Germany.

Zhisong Zhang, Hai Zhao, and Lianhui Qin. 2016. Probabilistic graph-based dependency parsing with convolutional neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1382–1392.

Zhuosheng Zhang, Yafang Huang, and Hai Zhao. 2018. Subword-augmented embedding for cloze reading comprehension. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1802–1814.

Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009a. Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - Shared Task (CoNLL)*, pages 61–66, Boulder, Colorado.

Hai Zhao, Wenliang Chen, and Chunyu Kit. 2009b. Semantic dependency parsing of nombank and propbank: An efficient integrated approach via a large-scale feature selection. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 30–39.

Hai Zhao, Wenliang Chen, and Guodong Zhou. 2009c. Multilingual dependency learning: A huge feature engineering method to semantic dependency parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - Shared Task (CoNLL)*, pages 55–60, Boulder, Colorado.

Hai Zhao and Chunyu Kit. 2008. Parsing syntactic and semantic dependencies with two single-stage maximum entropy models. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL)*, pages 203–207.

Hai Zhao, Xiaotian Zhang, and Chunyu Kit. 2013. Integrative semantic dependency parsing via efficient large-scale feature selection. *Journal of Artificial Intelligence Research*, 46:203–233.

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 (ACL)*, pages 1127–1137, Beijing, China.

Chenxi Zhu, Xipeng Qiu, Xinchi Chen, and Xuanjing Huang. 2015. A re-ranking model for dependency parser with recursive convolutional neural network. 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)*, volume 1, pages 1159–1168.