Syntax for Semantic Role Labeling, To Be, Or Not To Be

Shexia He[1,2,*], **Zuchao Li**[1,2,*], **Hai Zhao**[1,2,†], **Hongxiao Bai**[1,2], **Gongshen Liu**[3]
[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
{heshexia, charlee}@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn,
{baippa, lgshen}@sjtu.edu.cn

Abstract

Semantic role labeling (SRL) is dedicated to recognizing the predicate-argument structure of a sentence. Previous studies have shown syntactic information has a remarkable contribution to SRL performance. However, such perception was challenged by a few recent neural SRL models which give impressive performance without a syntactic backbone. This paper intends to quantify the importance of syntactic information to dependency SRL in deep learning framework. We propose an enhanced argument labeling model companying with an extended k-order argument pruning algorithm for effectively exploiting syntactic information. Our model achieves state-of-the-art results on the CoNLL-2008, 2009 benchmarks for both English and Chinese, showing the quantitative significance of syntax to neural SRL together with a thorough empirical survey over existing models.

1 Introduction

Semantic role labeling (SRL), namely semantic parsing, is a shallow semantic parsing task, which aims to recognize the predicate-argument structure of each predicate in a sentence, such as *who* did *what* to *whom*, *where* and *when*, etc. Specifically, we seek to identify arguments and label their semantic roles given a predicate. SRL is an important method to obtain semantic information beneficial to a wide range of natural language processing (NLP) tasks, including machine translation (Shi et al., 2016), question answering (Berant et al., 2013; Yih et al., 2016) and discourse relation sense classification (Mihaylov and Frank, 2016).

There are two formulizations for semantic predicate-argument structures, one is based on constituents (i.e., phrase or span), the other is based on dependencies. The latter proposed by the CoNLL-2008 shared task (Surdeanu et al., 2008) is also called semantic dependency parsing, which annotates the heads of arguments rather than phrasal arguments. Generally, SRL is decomposed into multi-step classification subtasks in pipeline systems, consisting of predicate identification and disambiguation, argument identification and classification.

In prior work of SRL, considerable attention has been paid to feature engineering that struggles to capture sufficient discriminative information, while neural network models are capable of extracting features automatically. In particular, syntactic information, including syntactic tree feature, has been show extremely beneficial to SRL since a larger scale of empirical verification of Punyakanok et al. (2008). However, all the work had to take the risk of erroneous syntactic input, leading to an unsatisfactory performance.

To alleviate the above issues, Marcheggiani et al. (2017) propose a simple but effective model for dependency SRL without syntactic input. It seems that neural SRL does not have to rely on syntactic features, contradicting with the belief that syntax is a necessary prerequisite for SRL as early as Gildea and Palmer (2002). This dramatic contradiction motivates us to make a thorough exploration on syntactic contribution to SRL.

This paper will focus on semantic dependency parsing and formulate SRL as one or two se-

* 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).

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 2061–2071
Melbourne, Australia, July 15 - 20, 2018.

quence tagging tasks with predicate-specific encoding. With the help of the proposed k-order argument pruning algorithm over syntactic tree, our model obtains state-of-the-art scores on the CoNLL benchmarks for both English and Chinese.

In order to quantitatively evaluate the contribution of syntax to SRL, we adopt the ratio between labeled F_1 score for semantic dependencies (Sem-F_1) and the labeled attachment score (LAS) for syntactic dependencies introduced by CoNLL-2008 Shared Task[1] as evaluation metric. Considering that various syntactic parsers contribute different syntactic inputs with various range of quality levels, the ratio provides a fairer comparison between syntactically-driven SRL systems, which will be surveyed by our empirical study.

2 Model

To fully disclose the predicate-argument structure, typical SRL systems have to step by step perform four subtasks. Since the predicates in CoNLL-2009 (Hajič et al., 2009) corpus have been pre-identified, we need to tackle three other subtasks, which are formulized into two-step pipeline in this work, predicate disambiguation and argument labeling. Namely, we do the work of argument identification and classification in one model.

Argument structure for each known predicate will be disclosed by our argument labeler over a sequence including possible arguments (candidates). There are two ways to determine the sequence, one is to simply input the entire sentence as a syntax-agnostic SRL system does, the other is to select words according to syntactic parse tree around the predicate as most previous SRL systems did. The latter strategy usually works through a syntactic tree based argument pruning algorithm. We will use the proposed k-order argument pruning algorithm (Section 2.1) to get a sequence $w = (w_1, \ldots, w_n)$ for each predicate. Then, we represent each word $w_i \in w$ as x_i (Section 2.2). Eventually, we obtain contextual features with sequence encoder (Section 2.3). The overall role labeling model is depicted in Figure 1.

2.1 Argument Pruning

As pointed out by Punyakanok et al. (2008), syntactic information is most relevant in identifying

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

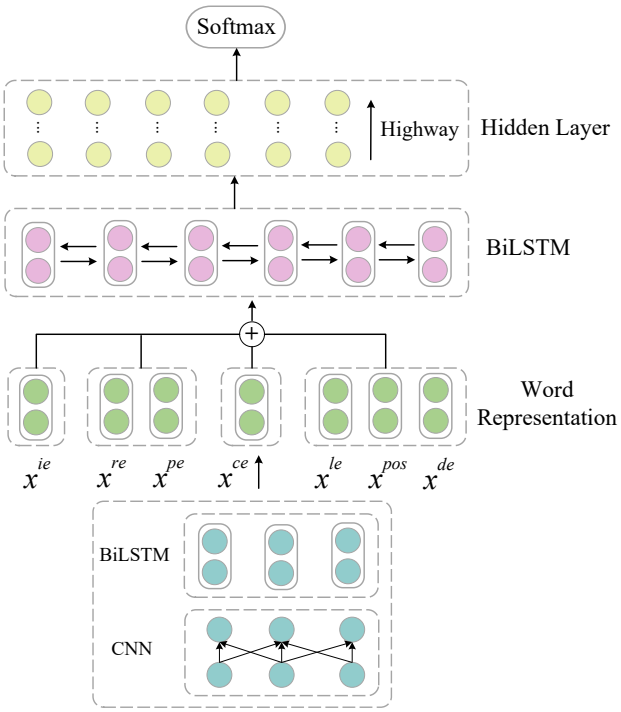

Figure 1: The Argument Labeling Model

the arguments, and the most crucial contribution of full parsing is in the pruning stage. In this paper, we propose a k-order argument pruning algorithm inspired by Zhao et al. (2009b). First of all, for node n and its descendant n_d in a syntactic dependency tree, we define the *order* to be the *distance* between the two nodes, denoted as $\mathcal{D}(n, n_d)$. Then we define k-order descendants of given node satisfying $\mathcal{D}(n, n_d) = k$, and k-order traversal that visits each node from the given node to its descendant nodes within k-th order. Note that the definition of k-order traversal is somewhat different from tree traversal in terminology.

A brief description of the proposed k-order pruning algorithm is given as follow. Initially, we set a given predicate as the current node in a syntactic dependency tree. Then, collect all its argument candidates by the strategy of k-order traversal. Afterwards, reset the current node to its syntactic head and repeat the previous step till the root of the tree. Finally, collect the root and stop. The k-order argument algorithm is presented in Algorithm 1 in detail. An example of a syntactic dependency tree for sentence *She began to trade the art for money* is shown in Figure 2.

The main reasons for applying the extended k-order argument pruning algorithm are two-fold.

Algorithm 1 k-order argument pruning algorithm

Input: A predicate p, the root node r given a syntactic dependency tree T, the order k

Output: The set of argument candidates S

initialization set p as current node c, $c = p$
for each descendant n_i of c in T **do**
  if $\mathcal{D}(c, n_i) \leq k$ and $n_i \notin S$ **then**
    $S = S + n_i$
  end if
end for
find the syntactic head c_h of c, and let $c = c_h$
if $c = r$ **then**
  $S = S + r$
else
  goto step 2
end if
return argument candidates set S

First, previous standard pruning algorithm may hurt the argument coverage too much, even though indeed arguments usually tend to surround their predicate in a close distance. As a sequence tagging model has been applied, it can effectively handle the imbalanced distribution between arguments and non-arguments, which is hardly tackled by early argument classification models that commonly adopt the standard pruning algorithm. Second, the extended pruning algorithm provides a better trade-off between computational cost and performance by carefully tuning k.

2.2 Word Representation

We produce a predicate-specific word representation x_i for each word w_i, where i stands for the word position in an input sequence, following Marcheggiani et al. (2017). However, we differ by (1) leveraging a predicate-specific indicator embedding, (2) using deeper refined representation, including character and dependency relation embeddings, and (3) applying recent advances in RNNs, such as highway connections (Srivastava et al., 2015).

In this work, word representation x_i is the concatenation of four types of features: predicate-specific feature, character-level, word-level and linguistic features. Unlike previous work, we leverage a predicate-specific indicator embedding x_i^{ie} rather than directly using a binary flag either 0 or 1. At character level, we exploit convolutional neural network (CNN) with bidirectional LSTM (BiLSTM) to learn character embedding

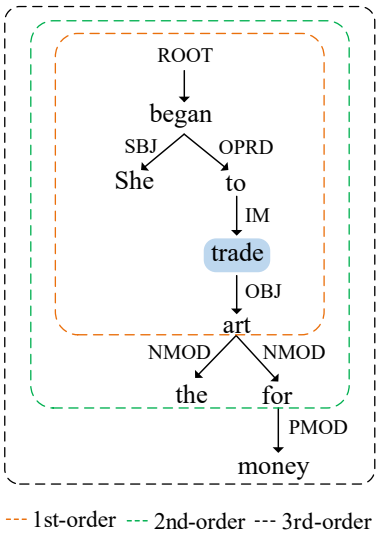

Figure 2: An example of *first*-order, *second*-order and *third*-order argument pruning. Shadow part indicates the given predicate.

x_i^{ce}. As shown in Figure 1, the representation calculated by the CNN is fed as input to BiLSTM. At word level, we use a randomly initialized word embedding x_i^{re} and a pre-trained word embedding x_i^{pe}. For linguistic features, we employ a randomly initialized lemma embedding x_i^{le} and a randomly initialized POS tag embedding x_i^{pos}. In order to incorporate more syntactic information, we adopt an additional feature, the dependency relation to syntactic head. Likewise, it is a randomly initialized embedding x_i^{de}. The resulting word representation is concatenated as $x_i = [x_i^{ie}, x_i^{ce}, x_i^{re}, x_i^{pe}, x_i^{le}, x_i^{pos}, x_i^{de}]$.

2.3 Sequence Encoder

As Long short-term memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) have shown significant representational effectiveness to NLP tasks, we thus use BiLSTM as the sentence encorder. Given an input sequence $x = (x_1, \ldots, x_n)$, BiLSTM processes the sequence in both forward and backward direction to obtain two separated hidden states, $\overrightarrow{h}_i$ which handles data from x_1 to x_i and $\overleftarrow{h}_i$ which tackles data from x_n to x_i for each word representation. Finally, we get a contextual representation $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$ by concatenating the states of BiLSTM networks.

To get the final predicted semantic roles, we exploit a multi-layer perceptron (MLP) with highway connections on the top of BiLSTM networks, which takes as input the hidden representation h_i

Hyperparameter	values
d^{ie} (indicator embedding)	16
d^{pe} (pre-trained embedding)	100
d^{ce} (character embedding)	300
d^{re} (word embedding)	100
d^{le} (lemma embedding)	100
d^{pos} (POS tag embedding)	32
d^{de} (dependency label embedding)	64
LSTM hidden sizes	512
BiLSTM layers	4
Hidden layers	10
Learning rate	0.001
Word dropout	0.1

Table 1: Hyperparameter values.

of all time steps. The MLP network consists of 10 layers with highway connections and we employ *ReLU* activations for the hidden layers. Finally, we use a softmax layer over the outputs to maximize the likelihood of labels.

2.4 Predicate Disambiguation

Although predicates have been identified given a sentence, predicate disambiguation is an indispensable task, which aims to determine the predicate-argument structure for an identified predicate in a particular context. Here, we also use the identical model (BiLSTM composed with MLP) for predicate disambiguation, in which the only difference is that we remove the syntactic dependency relation feature in corresponding word representation (Section 2.2). Exactly, given a predicate p, the resulting word representation is $p_i = [p_i^{ie}, p_i^{ce}, p_i^{re}, p_i^{pe}, p_i^{le}, p_i^{pos}]$.

3 Experiments

Our model[2] is evaluated on the CoNLL-2009 shared task both for English and Chinese datasets, following the standard training, development and test splits. The hyperparameters in our model were selected based on the development set, and are summarized in Table 1. Note that the parameters of predicate model are the same as these in argument model. All real vectors are randomly initialized, and the pre-trained word embeddings for English are GloVe vectors (Pennington et al., 2014). For Chinese, we exploit Wikipedia documents to train Word2Vec embeddings (Mikolov et al., 2013). During training procedures, we use the categorical cross-entropy as objective, with Adam optimizer (Kingma and Ba, 2015). We train models for a maximum of 20 epochs and obtain the nearly best model based on development results. For argument labeling, we preprocess corpus with k-order argument pruning algorithm. In addition, we use four CNN layers with single-layer BiLSTM to induce character representations derived from sentences. For English[3], to further enhance the representation, we adopt CNN-BiLSTM character embedding structure from AllenNLP toolkit (Peters et al., 2018).

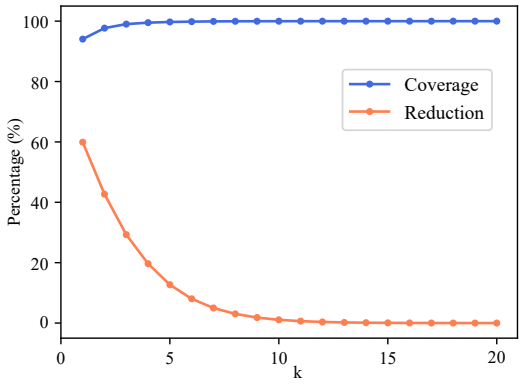

Figure 3: Changing curves of coverage and reduction with different k value on English training set. The coverage rate is the proportion of true arguments in pruning output, while the reduction is the one of pruned argument candidates in total tokens.

3.1 Preprocessing

During the pruning of argument candidates, we use the officially predicted syntactic parses provided by CoNLL-2009 shared-task organizers on both English and Chinese. Figure 3 shows changing curves of coverage and reduction following k on the English train set. According to our statistics, the number of non-arguments is ten times more than that of arguments, where the data distribution is fairly unbalanced. However, a proper pruning strategy could alleviate this problem. Accordingly, the first-order pruning reduces more than 50% candidates at the cost of missing 5.5% true ones on average, and the second-order prunes about 40% candidates with nearly 2.0% loss. The coverage of third-order has achieved 99% and it reduces approximately 1/3 corpus size.

It is worth noting that as k is larger than 19,

[2]The code is available at https://github.com/bcmi220/srl_syn_pruning.

[3]For Chinese, we do not use character embedding.

System (syntax-aware)	P	R	F_1
Single model			
Zhao et al. (2009a)	–	–	86.2
Zhao et al. (2009c)	–	–	85.4
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
Ours	**89.7**	**89.3**	**89.5**
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
System (syntax-agnostic)	P	R	F_1
Marcheggiani et al. (2017)	88.7	86.8	87.7
Ours	**89.5**	**87.9**	**88.7**

Table 2: Results on the English test set (WSJ).

System (syntax-aware)	P	R	F_1
Single model			
Zhao et al. (2009a)	–	–	74.6
Zhao et al. (2009c)	–	–	73.3
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
Ours	**81.9**	**76.9**	**79.3**
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
System (syntax-agnostic)	P	R	F_1
Marcheggiani et al. (2017)	79.4	76.2	77.7
Ours	**81.7**	**76.1**	**78.8**

Table 3: Results on English out-of-domain test set (Brown).

there will come full coverage on all argument candidates for English training set, which let our high order pruning algorithm degrade into a syntax-agnostic setting. In this work, we use the tenth-order pruning for pursuing the best performance.

3.2 Results

Our system performance is measured with the official script from CoNLL-2009 benchmarks, combining the output of our predicate disambiguation with our semantic role labeling. Our predicate disambiguation model achieves the accuracy of 95.01% and 95.58%[4] on development and test sets, respectively. We compare our model performance with the state-of-the-art models for dependency SRL.[5] Noteworthily, our model is local and single without reranking, which neither includes global inference nor combines multiple models. The experimental results on the English in-domain (WSJ) and out-of-domain (Brown) test sets are shown in Tables 2 and 3, respectively.

For English, our syntax-aware model outperforms previously published best single model, scoring 89.5% F_1 with 1.5% absolute improvement on the in-domain (WSJ) test data. Compared with ensemble models, our single model even provides better performance ($+0.4\%$ F_1) than the system (Marcheggiani and Titov, 2017), and significantly surpasses all the rest models. In the syntax-agnostic setting (without pruning and dependency relation embedding), we also reach the new state-of-the-art, achieving a performance gain of 1% F_1.

System (syntax-aware)	P	R	F_1
Zhao et al. (2009a)	80.4	75.2	77.7
Björkelund et al. (2009)	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
Ours	**84.2**	**81.5**	**82.8**
System (syntax-agnostic)	P	R	F_1
Marcheggiani et al. (2017)	83.4	79.1	81.2
Ours	**84.5**	**79.3**	**81.8**

Table 4: Results on the Chinese test set.

On the out-of-domain (Brown) test set, we achieve the new best results of 79.3% (syntax-aware) and 78.8% (syntax-agnostic) in F_1 scores. Moreover, our syntax-aware model performs better than the syntax-agnostic one.

Table 4 presents the results on Chinese test set. Even though we use the same parameters as for English, our model also outperforms the best reported results by 0.3% (syntax-aware) and 0.6% (syntax-agnostic) in F_1 scores.

[4]Note that we give a slightly better predicate model than Roth and Lapata (2016), with 94.77% and 95.47% accuracy on development and test sets, respectively.

[5]Here, we do not compare against span-based SRL models, which annotate roles for entire argument spans instead of semantic dependencies.

System(without predicate sense)	P	R	F_1
1st-order	84.4	82.6	83.5
2nd-order	84.8	83.0	83.9
3rd-order	85.1	83.3	84.2
Marcheggiani and Titov (2017)	85.2	81.6	83.3

Table 5: SRL results without predicate sense.

Our system	P	R	F_1
BiLSTM	86.5	85.1	85.8
basic model	86.3	85.7	86.0
+ indicator embedding	86.8	85.8	86.3
+ character embedding	87.2	86.6	86.9
+ both	87.7	87.0	87.3
BiLSTM + both	87.3	86.7	87.0

Table 6: Ablation on development set. The "+" denotes a specific version over the basic model.

3.3 Analysis

To evaluate the contributions of key factors in our method, a series of ablation studies are performed on the English development set.

In order to demonstrate the effectiveness of our k-order pruning algorithm, we report the SRL performance excluding predicate senses in evaluation, eliminating the performance gain from predicate disambiguation. Table 5 shows the results from our syntax-aware model with lower order argument pruning. Compared to the best previous model, our system still yields an increment in recall by more than 1%, leading to improvements in F_1 score. It demonstrates that refining syntactic parser tree based candidate pruning does help in argument recognition.

Table 6 presents the performance of our syntax-agnostic SRL system with a basic configuration, which removes components, including indicator and character embeddings. Note that the first row is the results of BiLSTM (removing MLP from basic model), whose encoding is the same as Marcheggiani et al. (2017). Experiments show that both enhanced representations improve over our basic model, and our adopted labeling model is superior to the simple BiLSTM.

Figure 4 shows F_1 scores in different k-order pruning together with our syntax-agnostic model. It also indicates that the least first-order pruning fails to give satisfactory performance, the best performing setting coming from a moderate setting of $k = 10$, and the largest k shows that our argument pruning falls back to syntax-agnostic type. Meanwhile, from the best k setting to the lower order pruning, we receive a much faster performance drop, compared to the higher order pruning until the complete syntax-agnostic case. The proposed k-order pruning algorithm always works even it reaches the syntax-agnostic setting, which empirically explains why the current syntax-aware and syntax-agnostic SRL models hold little performance difference, as maximum k-order pruning actually removes few words just like syntax-agnostic model.

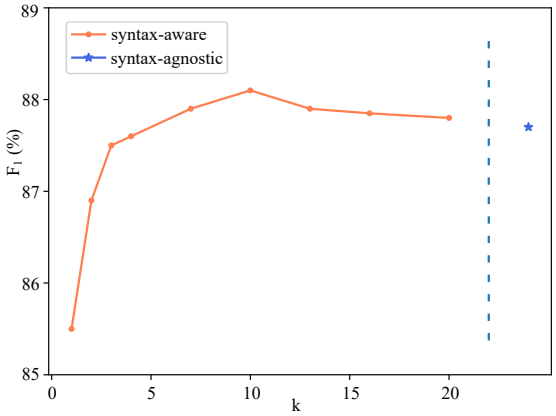

Figure 4: F_1 scores by k-order pruning and the syntax-agnostic result on English development set.

3.4 End-to-end SRL

In this work, we consider additional model that integrates predicate disambiguation and argument labeling into one sequence labeling model. In order to implement an end-to-end model, we introduce a virtual root (VR) for predicate disambiguation similar to Zhao et al. (2013) who handled the entire SRL task as word pair classification. Concretely, we add a predicate sense feature to the input sequence by concatenating a VR. The word representation of VR is randomly initialized during training. In Figure 5, we give an example sequence with the labels for the given sentence.

We also report results of our end-to-end model on CoNLL-2009 test set with syntax-aware and syntax-agnostic settings. As shown in Table 7, our end-to-end model yields slightly weaker performance compared with our pipeline. A reasonable account for performance degradation is that the training data has completely different genre distributions over predicate senses and argument roles, which may be somewhat confusing for integrative model to make classification decisions.

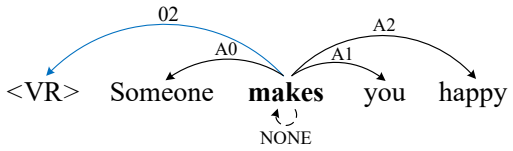

Figure 5: An example sequence with labels of end-to-end model (*makes* is the given predicate).

Our system	P	R	F_1
syntax-aware (end-to-end)	89.3	88.7	89.0
syntax-aware (pipeline)	89.7	89.3	89.5
syntax-agnostic (end-to-end)	88.9	87.9	88.4
syntax-agnostic (pipeline)	89.5	87.9	88.7

Table 7: Comparison of results on CoNLL-2009 data between our end-to-end and pipeline models.

3.5 CoNLL-2008 SRL Setting

For a full SRL task, the predicate identification subtask is also indispensable, which has been included in CoNLL-2008 shared task. We thus evaluate our model in terms of data and setting of the CoNLL-2008 benchmark (WSJ).

To identify predicates, we train the BiLSTM-MLP sequence labeling model with same parameters in Section 2.4 to tackle the predicate identification and disambiguation subtasks in one shot, and the only difference is that we remove the predicate-specific indicator feature. The F_1 score of our predicate labeling model is 90.53% on in-domain (WSJ) data. Compared with the best reported results, we observe absolute improvements in semantic F_1 of 0.8% (in Table 8). Note that as predicate identification is introduced, our same model shows about 6% performance loss for either syntax-agnostic or syntax-aware case, which indicates that predicate identification should be carefully handled, as it is very needed in a complete practical SRL system.

4 Syntactic Contribution

Syntactic information plays an informative role in semantic role labeling. However, few studies were done to quantitatively evaluate the syntactic contribution to SRL. Furthermore, we observe that most of the above compared neural SRL systems took the syntactic parser of (Björkelund et al., 2010) as syntactic inputs instead of the one from CoNLL-2009 shared task, which adopted a much weaker syntactic parser. Especially (Marcheggiani and Titov, 2017), adopted an external syntactic

System	LAS	Sem-F_1
Johansson and Nugues (2008)	90.13	81.75
Zhao and Kit (2008)	87.52	77.67
Zhao et al. (2009b)	88.39	82.1 (80.53)
	89.28	82.5 (80.94)
Zhao et al. (2013)	88.39	82.5 (80.91)
	89.28	82.4 (80.88)
Ours (syntax-agnostic)	–	82.9
Ours (syntax-aware)	86.0	83.3

Table 8: Results on the CoNLL-2008 in-domain (WSJ) test set. The results in parenthesis are on WSJ + Brown test set.

parser with even higher parsing accuracy. Contrarily, our SRL model is based on the automatically predicted parse with moderate performance provided by CoNLL-2009 shared task, but outperforms their models.

This section thus attempts to explore how much syntax contributes to dependency-based SRL in deep learning framework and how to effectively evaluate relative performance of syntax-based SRL. To this end, we conduct experiments for empirical analysis with different syntactic inputs.

Syntactic Input In order to obtain different syntactic inputs, we design a faulty syntactic tree generator (refer to STG hereafter), which is able to produce random errors in the output parse tree like a true parser does. To simplify implementation, we construct a new syntactic tree based on the gold standard parse tree. Given an input error probability distribution estimated from a true parser output, our algorithm presented in Algorithm 2 stochastically modifies the syntactic heads of nodes on the premise of a valid tree.

Evaluation Measure For SRL task, the primary evaluation measure is the semantic labeled F_1 score. However, the score is influenced by the quality of syntactic input to some extent, leading to unfaithfully reflecting the competence of syntax-based SRL system. Namely, this is not the outcome of a true and fair quantitative comparison for these types of SRL models. To normalize the semantic score relative to syntactic parse, we take into account additional evaluation measure to estimate the actual overall performance of SRL. Here, we use the ratio between labeled F_1 score for semantic dependencies (Sem-F_1) and the labeled attachment score (LAS) for syntactic dependencies

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)	90.3*	89.1	86.8	**88.0**	97.5
Ours + CoNLL-2009 predicted	86.0	89.7	89.3	89.5	104.0
Ours + Auto syntax	90.0	90.5	89.3	**89.9**	99.9
Ours + Gold syntax	100	91.0	89.7	90.3	90.3

Table 9: 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. A superscript * indicates LAS results from our personal communication with the authors.

Algorithm 2 Faulty Syntactic Tree Generator

Input: A gold standard syntactic tree GT, the specific error probability p

Output: The new generative syntactic tree NT

N denotes the number of nodes in GT
for each node $n \in GT$ **do**
  $r =$ random(0, 1), a random number
  if $r < p$ **then**
    $h =$ random(0, N), a random integer
    find the syntactic head n_h of n in GT
    modify $n_h = h$, and get a new tree NT
    if NT is a valid tree **then**
      break
    else
      goto step 5
    end if
  end if
end for
return the new generative tree NT

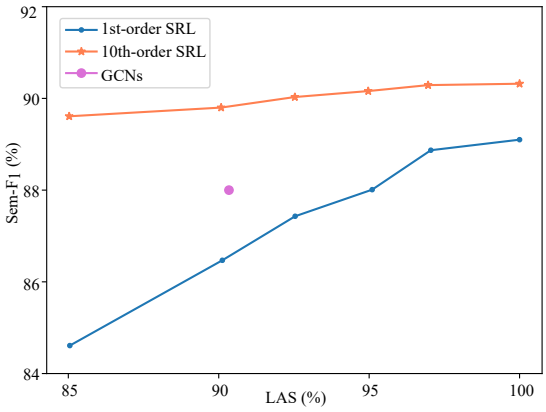

Figure 6: The Sem-F_1 scores of our models with different quality of syntactic inputs vs. GCNs (Marcheggiani and Titov, 2017) on test set.

proposed by Surdeanu et al. (2008) as evaluation metric.[6] The benefits of this measure are twofold: quantitatively evaluating syntactic contribution to SRL and impartially estimating the true performance of SRL, independent of the performance of the input syntactic parser.

Table 9 reports the performance of existing models[7] in term of Sem-F_1/LAS ratio on CoNLL-2009 English test set. Interestingly, even though our system has significantly lower scores than others by 3.8% LAS in syntactic components, we obtain the highest results both on Sem-F_1 and the Sem-F_1/LAS ratio, respectively. These results show that our SRL component is relatively much stronger. Moreover, the ratio comparison in Table 9 also shows that since the CoNLL-2009 shared task, most SRL works actually benefit from the enhanced syntactic component rather than the improved SRL component itself. All post-CoNLL SRL systems, either traditional or neural types, did not exceed the top systems of CoNLL-2009 shared task, (Zhao et al., 2009c) (SRL-only track using the provided predicated syntax) and (Zhao et al., 2009a) (Joint track using self-developed parser). We believe that this work for the first time reports both higher Sem-F_1 and higher Sem-F_1/LAS ratio since CoNLL-2009 shared task.

We also perform our first and tenth order pruning models with different erroneous syntactic inputs generated from STG and evaluate their per-

[6]The idea of ratio score in Surdeanu et al. (2008) actually was from author of this paper, Hai Zhao, which has been indicated in the acknowledgement part of Surdeanu et al. (2008).

[7]Note that several SRL systems without providing syntactic information are not listed in the table.

formance using the Sem-F_1/LAS ratio. Figure 6 shows Sem-F_1 scores at different quality of syntactic parse inputs on the English test set whose LAS varies from 85% to 100%. Compared to previous state-of-the-arts (Marcheggiani and Titov, 2017). Our tenth-order pruning model gives quite stable SRL performance no matter the syntactic input quality varies in a broad range, while our first-order pruning model yields overall lower results (1-5% F_1 drop), owing to missing too many true arguments. These results show that high-quality syntactic parses may indeed enhance dependency SRL. Furthermore, it indicates that our model with an accurate enough syntactic input as Marcheggiani and Titov (2017), namely, 90% LAS, will give a Sem-F_1 exceeding 90% for the first time in the research timeline of semantic role labeling.

5 Related Work

Semantic role labeling was pioneered by Gildea and Jurafsky (2002). Most traditional SRL models rely heavily on feature templates (Pradhan et al., 2005; Zhao et al., 2009b; Björkelund et al., 2009). Among them, Pradhan et al. (2005) combined features derived from different syntactic parses based on SVM classifier, while Zhao et al. (2009b) presented an integrative approach for dependency SRL by greedy feature selection algorithm. Later, Collobert et al. (2011) proposed a convolutional neural network model of inducing word embeddings substituting for hand-crafted features, which was a breakthrough for SRL task.

With the impressive success of deep neural networks in various NLP tasks (Zhang et al., 2016; Qin et al., 2017; Cai et al., 2017), a series of neural SRL systems have been proposed. Foland and Martin (2015) presented a dependency semantic role labeler using convolutional and time-domain neural networks, while 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, researchers consider multiple ways to effectively integrate syntax into SRL learning. Roth and Lapata (2016) introduced 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 neural models. Differently, Marcheggiani et al. (2017) proposed a syntax-agnostic model using effective word representation for dependency SRL, which for the first time achieves comparable performance as state-of-the-art syntax-aware SRL models.

However, most neural SRL works seldom pay much attention to the impact of input syntactic parse over the resulting SRL performance. This work is thus more than proposing a high performance SRL model through reviewing the highlights of previous models, and presenting an effective syntactic tree based argument pruning. Our work is also closely related to (Punyakanok et al., 2008; He et al., 2017). Under the traditional methods, Punyakanok et al. (2008) investigated the significance of syntax to SRL system and shown syntactic information most crucial in the pruning stage. He et al. (2017) presented extensive error analysis with deep learning model for span SRL, including discussion of how constituent syntactic parser could be used to improve SRL performance.

6 Conclusion and Future Work

This paper presents a simple and effective neural model for dependency-based SRL, incorporating syntactic information with the proposed extended k-order pruning algorithm. With a large enough setting of k, our pruning algorithm will result in a syntax-agnostic setting for the argument labeling model, which smoothly unifies syntax-aware and syntax-agnostic SRL in a consistent way. Experimental results show that with the help of deep enhanced representation, our model outperforms the previous state-of-the-art models in both syntax-aware and syntax-agnostic situations.

In addition, we consider the Sem-F1/LAS ratio as a mean of evaluating syntactic contribution to SRL, and true performance of SRL independent of the quality of syntactic parser. Though we again confirm the importance of syntax to SRL with empirical experiments, we are aware that since (Pradhan et al., 2005), the gap between syntax-aware and syntax-agnostic SRL has been greatly reduced, from as high as 10% to only 1-2% performance loss in this work. However, maybe we will never reach a satisfying conclusion, as whenever one proposes a syntax-agnostic SRL system which can outperform all syntax-aware ones at then, always there comes argument that you have never fully explored creative new method to effectively exploit the syntax input.

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)*. Seattle, Washington, USA, pages 1533–1544.

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)*. Beijing, China, pages 33–36.

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*. Boulder, Colorado, pages 43–48.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for Chinese. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada, pages 608–615.

Ronan Collobert, Jason Weston, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12(1):2493–2537.

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)*. Philadelphia, Pennsylvania, USA, pages 239–246.

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*. Boulder, Colorado, pages 1–18.

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)*. Vancouver, Canada, pages 473–483.

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

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic-semantic analysis with propbank and nombank. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL)*. pages 183–187.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd 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.

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)*. Vancouver, Canada, pages 411–420.

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

Todor Mihaylov and Anette Frank. 2016. Discourse relation sense classification using cross-argument semantic similarity based on word embeddings. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task (CoNLL)*. Berlin, Germany, pages 100–107.

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)*. Doha, Qatar, pages 1532–1543.

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)*. Ann Arbor, Michigan, pages 581–588.

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.

Lianhui Qin, Zhisong Zhang, Hai Zhao, Zhiting Hu, and Eric Xing. 2017. Adversarial connective-exploiting networks for implicit discourse relation classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada, pages 1006–1017.

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)*. Berlin, Germany, pages 1192–1202.

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)*. Berlin, Germany, pages 2245–2254.

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)*. Manchester, England, pages 159–177.

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)*. Berlin, Germany, pages 201–206.

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 (ACL)*. Berlin, Germany, pages 1382–1392.

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)*. Boulder, Colorado, pages 61–66.

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 (EMNLP)*. Singapore, 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)*. Boulder, Colorado, pages 55–60.

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.