Syntax-Enhanced Self-Attention-Based Semantic Role Labeling

Yue Zhang, Rui Wang, Luo Si
Alibaba Group, China
{shiyu.zy, masi.wr, luo.si}@alibaba-inc.com

Abstract

As a fundamental NLP task, semantic role labeling (SRL) aims to discover the semantic roles for each predicate within one sentence. This paper investigates how to incorporate syntactic knowledge into the SRL task effectively. We present different approaches of encoding the syntactic information derived from dependency trees of different quality and representations; we propose a syntax-enhanced self-attention model and compare it with other two strong baseline methods; and we conduct experiments with newly published deep contextualized word representations as well. The experiment results demonstrate that with proper incorporation of the high quality syntactic information, our model achieves a new state-of-the-art performance for the Chinese SRL task on the CoNLL-2009 dataset.

1 Introduction

The task of semantic role labeling (SRL) is to recognize arguments for a given predicate in one sentence and assign labels to them, including "who" did "what" to "whom", "when", "where", etc. Figure 1 is an example sentence with both semantic roles and syntactic dependencies. Since the nature of semantic roles is more abstract than the syntactic dependencies, SRL has a wide range of applications in different areas, e.g., text classification (Sinoara et al., 2016), text summarization (Genest and Lapalme, 2011; Khan et al., 2015), recognizing textual entailment (Burchardt et al., 2007; Stern and Dagan, 2014), information extraction (Surdeanu et al., 2003), question answering (Shen and Lapata, 2007; Yih et al., 2016), and so on.

Traditionally, syntax is the bridge to reach semantics. However, along with the popularity of the end-to-end models in the NLP community, various recent studies have been discussing the necessity of syntax in the context of SRL. For instance, He et al. (2017) have observed that only good syntax helps with the SRL performance. Xia et al. (2019) have explored what kind of syntactic information or structure is better suited for the SRL model. Cai et al. (2018) have compared syntax-agnostic and syntax-aware approaches and claim that the syntax-agnostic model surpasses the syntax-aware ones.

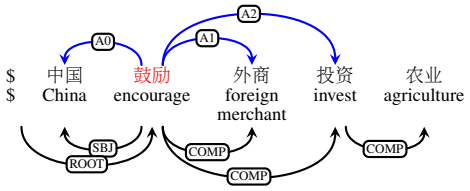

Figure 1: An example of one sentence with its syntactic dependency tree and semantic roles. Arcs above the sentence are semantic role annotations for the predicate "鼓励 (encourage)" and below the sentence are syntactic dependency annotations of the whole sentence. The meaning of this sentence is "China encourages foreign merchants to invest in agriculture".

In this paper, we focus on analyzing the relationship between the syntactic dependency information and the SRL performance. In particular, we investigate the following four aspects: 1) *Quality* of the syntactic information: whether the performance of the syntactic parser output affects the SRL performance; 2) *Representation* of the syntactic information: how to represent the syntactic dependencies to better preserve the original structural information; 3) *Incorporation* of the syntactic information: at which layer of the SRL model and how to incorporate the syntactic information; and 4) the *Relationship* with other external resources: when we append other external resources into the SRL model, whether their contributions are orthogonal to the syntactic dependencies.

For the main architecture of the SRL model, many neural-network-based models use BiLSTM

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 616–626, Hong Kong, China, November 3–7, 2019.

as the encoder (e.g., Cai et al. (2018); Li et al. (2018); He et al. (2018)), while recently self-attention-based encoder becomes popular due to both the effectiveness and the efficiency (Vaswani et al., 2017; Tan et al., 2017; Strubell et al., 2018). By its nature, the self-attention-based model directly captures the relation between words in the sentence, which is convenient to incorporate syntactic dependency information. Strubell et al. (2018) replace one attention head with pre-trained syntactic dependency information, which can be viewed as a *hard* way to inject syntax into the neural model. Enlightened by the machine translation model proposed by Shaw et al. (2018), we introduce the *Relation-Aware* method to incorporate syntactic dependencies, which is a *softer* way to encode richer structural information.

Various experiments for the Chinese SRL on the CoNLL-2009 dataset are conducted to evaluate our hypotheses. From the empirical results, we observe that: 1) The quality of the syntactic information is essential when we incorporate structural information into the SRL model; 2) Deeper integration of the syntactic information achieves better results than the simple concatenation to the inputs; 3) External pre-trained contextualized word representations help to boost the SRL performance further, which is not entirely overlapping with the syntactic information.

In summary, the contributions of our work are:

- We present detailed experiments on different aspects of incorporating syntactic information into the SRL model, in what quality, in which representation and how to integrate.
- We introduce the relation-aware approach to employ syntactic dependencies into the self-attention-based SRL model.
- We compare our approach with previous studies, and achieve state-of-the-art results with and without external resources, i.e., in the so-called *closed* and *open* settings.

2 Related work

Traditional semantic role labeling task (Gildea and Jurafsky, 2002) presumes that the syntactic structure of the sentence is given, either being a constituent tree or a dependency tree, like in the CoNLL shared tasks (Carreras and Màrquez, 2005; Surdeanu et al., 2008; Hajič et al., 2009). Recent neural-network-based approaches can be roughly categorized into two classes: 1) making use of the syntactic information (FitzGerald et al., 2015; Roth and Lapata, 2016; Qian et al., 2017; Marcheggiani and Titov, 2017), and 2) pure end-to-end learning from tokens to semantic labels, e.g., Zhou and Xu (2015); Marcheggiani et al. (2017).

Roth and Lapata (2016) utilize an LSTM model to obtain embeddings from the syntactic dependency paths; while Marcheggiani and Titov (2017) construct Graph Convolutional Networks to encode the dependency structure. Although He et al. (2017)'s approach is a pure end-to-end learning, they have included an analysis of adding syntactic dependency information into English SRL in the discussion section. Cai et al. (2018) have compared syntax-agnostic and syntax-aware approaches and Xia et al. (2019) have compared different ways to represent and encode the syntactic knowledge.

In another line of research, Tan et al. (2017) utilize the Transformer network for the encoder instead of the BiLSTM. Strubell et al. (2018) present a novel and effective multi-head self-attention model to incorporate syntax, which is called LISA (Linguistically-Informed Self-Attention). We follow their approach of replacing one attention head with the dependency head information, but use a softer way to capture the pairwise relationship between input elements (Shaw et al., 2018).

For the datasets and annotations of the SRL task, most of the previous research focuses on 1) PropBank (Palmer et al., 2005) and NomBank (Meyers et al., 2004) annotations, i.e., the CoNLL 2005 (Carreras and Màrquez, 2005) and CoNLL 2009 (Hajič et al., 2009) shared tasks; 2) OntoNotes annotations (Weischedel et al., 2011), i.e., the CoNLL 2005 and CoNLL 2012 datasets and more; 3) and FrameNet (Baker et al., 1998) annotations. For the non-English languages, not all of them are widely available. Apart from these, in the broad range of semantic processing, other formalisms non-exhaustively include abstract meaning representation (Banarescu et al., 2013), universal decompositional semantics (White et al., 2016), and semantic dependency parsing (Oepen et al., 2015). Abend and Rappoport (2017) give a better overview of various semantic representations. In this paper, we primarily work on the Chinese and English datasets from the

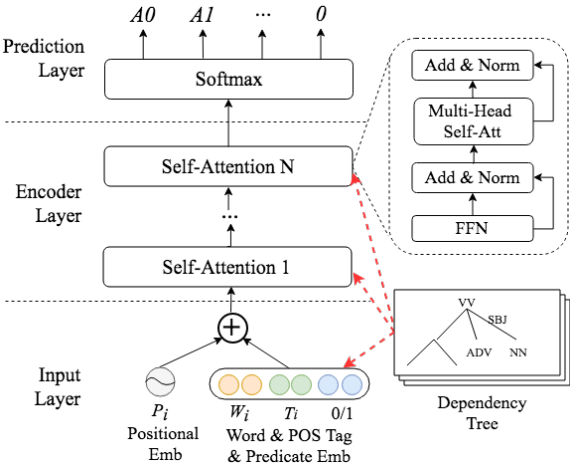

Figure 2: Architecture of our syntax-enhanced self-attention-based SRL model. Red dotted arrows indicate different locations where we incorporate linguistic knowledge in different forms. The dotted box on the upper right is the detailed composition of the self-attention block.

CoNLL-2009 shared task and focus on the effectiveness of incorporating syntax into the Chinese SRL task.

3 Approaches

In this section, we first introduce the basic architecture of our self-attention-based SRL model, and then present two different ways to encode the syntactic dependency information. Afterwards, we compare three approaches to incorporate the syntax into the base model, concatenation to the input embedding, LISA, and our proposed relation-aware method.

3.1 The Basic Architecture

Our basic model is a multi-head self-attention-based model, which is effective in SRL task as previous work proves (Tan et al., 2018). The model consists of three layers: the input layer, the encoder layer and the prediction layer as shown in Figure 2.

3.1.1 Input Layer

The input layer contains three types of embeddings: token embedding, predicate embedding, and positional embedding.

Token Embedding includes word embedding, part-of-speech (POS) tag embedding.

Predicate Embedding has been proposed by He et al. (2017), and its binary embedding is used to indicate the predicates indices in each sentence.

Positional Embedding encodes the order of the input word sequence. We follow Vaswani et al. (2017) to use time positional embedding, which is formulated as follows:

$$\begin{aligned} PE(t, 2i) &= sin(t/10000^{2i/d}) \\ PE(t, 2i+1) &= cos(t/10000^{2i/d}) \end{aligned} \quad (1)$$

where t is the position, i means the dimension, and d is the dimension of the model input embedding.

3.1.2 Encoder Layer

The self-attention block is almost the same as Transformer encoder proposed by Vaswani et al. (2017). Specifically the Transformer encoder contains a feed-forward network (**FFN**) and a multi-head attention network. The former is followed by the latter. In this work, we exchange their order, so that the multi-head attention module is moved behind the FFN module[1] as Figure 2 shows.

FFN The FFN module consists of two affine layers with a ReLU activation in the middle. Formally, we have the following equation:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \quad (2)$$

Multi-Head Attention The basic attention mechanism used in the multi-head attention function is called "Scaled Dot-Product Attention", which is formulated as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (3)$$

where Q is queries, K is keys, and V is values.

In the multi-head attention setting, it first maps the input matrix X into queries, keys and values matrices by using h different learned linear projections. Taking queries Q as an example:

$$Linear_i^Q(X) = XW_i^Q + b_i^Q \quad (4)$$

where $0 \leq i < h$. Keys and values use similar projections.

On each of these projections, we perform the scaled dot-product attention in parallel. These parallel output values are concatenated and once again projected into the final values. Equation 5 depicts the above operations.

$$MultiHead(X) = Concat(head_1, \ldots, head_h)W^O \quad (5)$$

[1]Changing the order delivers better empirical results (Tan et al., 2018) and our experiments show the same trend. The details of the experiments are not listed in this paper.

where

$$head_i = Attention(Linear_i^Q(X), Linear_i^K(X), Linear_i^V(X)) \quad (6)$$

More details about multi-head attention can be found in Vaswani et al. (2017).

Add & Norm We employ a residual connection to each module, followed by a layer normalization (Ba et al., 2016) operation. The output of each module is formulated as

$$x = LayerNorm(x + f(x)) \quad (7)$$

where $f(x)$ is implemented by each above module.

3.2 Representation of the Syntactic Dependencies

3.2.1 Dependency Head & Relation

The most intuitive way to represent syntactic information is to use individual dependency relations directly, like dependency head and dependency relation label, denoted as DEP and REL for short.

Except for LISA, where DEP is a one-hot matrix of dependency head word index described in 3.3.2, in other cases, we use the corresponding head word. REL is the dependency relation between the word and its syntactic head. We take both DEP and REL as common strings and map them into dense vectors in the similar way of word embedding.

3.2.2 Dependency Path & Relation Path

In order to preserve the structural information of dependency trees as much as possible, we take the syntactic path between candidate arguments and predicates in dependency trees as linguistic knowledge. Referring to Xia et al. (2019), we use the Tree-based Position Feature (TPF) as Dependency Path (DEPPATH) and use the Shortest Dependency Path (SDP) as Relation Path (RELPATH).

To generate DEPPATH & RELPATH between candidate argument and predicate, we firstly find their lowest common ancestor. Then we get two sub-paths, one is from the ancestor to the predicate and the other is from the ancestor to the argument. For DEPPATH, we compute distance from ancestor to predicate and argument respectively and then concatenate two distances with the separator ‘,’.

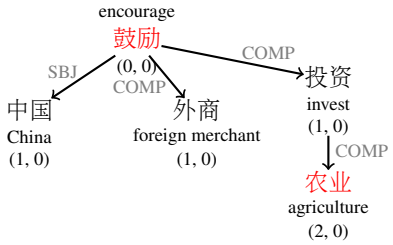

Figure 3: The syntactic dependency tree of the sentence “中国鼓励外商投资农业” (China encourages foreign merchants to invest in agriculture). Numbers in brackets are the DEPPATH for each candidate argument with the predicate “鼓励 (encourage)”. Light grey labels on the arcs are the syntactic dependency labels.

For RELPATH, we concatenate the labels appearing in each sub-path with the separator “_” respectively to get two label paths, and then concatenate the two label paths with the separator ‘,’.

As shown in Figure 3, the lowest common ancestor of the predicate “鼓励 (encourage)” and the candidate argument “农业 (agriculture)” is “鼓励 (encourage)”, so their DEPPATH is “2,0” and its RELPATH is “COMP_COMP,”[2].

We take both DEPPATH and RELPATH as common strings and map them into dense vectors in the similar way of DEP and REL.

3.3 Incorporation Methods

3.3.1 Input Embedding Concatenation

To incorporate syntactic knowledge, one simple method is to take it as part of the neural network input, denoted as INPUT. We represent the syntactic information with dense vectors, and concatenate it with other information like word embedding:

$$input = E_W \oplus E_S. \quad (8)$$

where $\oplus$ means concatenation; E_W means the original inputs of the neural model and E_S means the embedding of syntax information, such as DEP/REL or DEPPATH/RELPATH.

3.3.2 LISA

Strubell et al. (2018) propose the linguistically-informed self-attention model (LISA for short) to combine SRL and dependency parsing as multi-task learning in a subtle way. Based on the multi-head self-attention model, LISA uses one attention head to predict the dependency results and it

[2] When predicate is the ancestor of argument, sub-path from ancestor to predicate is none. We use ‘0’ to represent distance and use empty string to represent the label path, and vice versa.

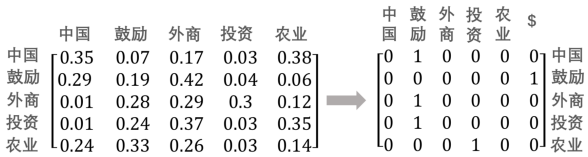

Figure 4: Attention matrix of the replaced attention head in the LISA model. The left matrix is the original $softmax$ attention, and the right is a one-hot matrix copied from the syntactic dependency head results.

can also directly use pre-trained dependency head results to replace the attention matrix during testing.

Being different from their multi-task learning, we make the replacement of one attention head during both training and testing. Instead of the original $softmax$ attention matrix, we use a one-hot matrix, generated by mapping the dependency head index of each word into a 0-1 vector of the sentence length as Figure 4 shows.

We add the dependency relation information with V in the replaced head so that we can make full use of the syntactic knowledge. The replaced attention head is formulated as follows:

$$Attention(Q, K, V) = M_D\ (V \oplus E_R) \quad (9)$$

where M_D is the one-hot dependency head matrix and E_R means the embedding of dependency relation information, such as REL or RELPATH.

3.3.3 Relation-Aware Self-Attention

Relation-aware self-attention model (RELAWE for brevity) incorporates external information into the attention. By this way, the model considers the pairwise relationships between input elements, which highly agrees with the task of SRL, i.e., aiming to find the semantic relations between the candidate argument and predicate in one sentence.

Compared to the standard attention, in this paper, we add the dependency information into Q and V in each attention head, like equation (10) shows:

$$Attention(Q, K, V) = softmax(\frac{(Q + E_D + E_R)K^T}{\sqrt{d_k}})(V + E_D + E_R) \quad (10)$$

where E_D and E_R mean the syntactic dependency head and relation information respectively. For our multi-layer multi-head self-attention model, we make this change to each head of the first N self-attention layers.

	dev		test	
	UAS	**LAS**	**UAS**	**LAS**
AUTO	80.50	78.34	80.70	78.46
BIAFFINE	89.00	85.86	89.05	85.60
BIAFFINEBERT	91.76	89.08	92.14	89.23

Table 1: Syntactic dependency performance for different parsers. AUTO indicates the automatic dependency trees provided by the CoNLL-09 Chinese dataset. BIAFFINE means the trees are generated by BiaffineParser with pre-trained word embedding on the Gigaword corpus while BIAFFINEBERT is the same parser with BERT. We use the labeled accuracy score (LAS) and unlabeled accuracy score (UAS) to measure the quality of syntactic dependency trees.

4 Experiment

4.1 Settings

Datasets & Evaluation Metrics Our experiments are conducted on the CoNLL-2009 shared task dataset (Hajič et al., 2009). We use the official evaluation script to compare the output of different system configurations, and report the labeled precision (P), labeled recall (R) and labeled f-score (F1) for the semantic dependencies.

Word Representations Most of our experiments are conducted in the *closed* setting without any external word embeddings or data resources than those provided by the CoNLL-2009 datasets. In the *closed* setting, word embedding is initialized by a Gaussian distribution with mean 0 and variance $\frac{1}{\sqrt{d}}$, where d is the dimension of embedding size of each layer.

For the experiments with external resources in the *open* setting, we utilize 1) word embeddings pre-trained with GloVe (Pennington et al., 2014) on the Gigaword corpus for Chinese and the published embeddings with 100 dimensions pre-trained on Wikipedia and Gigaword for English; and 2) ELMo[3] (Peters et al., 2018) and BERT[4] (Devlin et al., 2018), two recently proposed effective deep contextualized word representations[5].

[3]We use the released model on their website: https://github.com/allenai/allennlp/blob/master/tutorials/how_to/elmo.md

[4]We generate our pre-trained BERT embedding with the released model in https://github.com/google-research/bert. The model uses character-based tokenization for Chinese, which require us to maintain alignment between our input text and output text of Bert. So we take take embedding of the first word piece as the whole word representation.

[5]In *open* setting, we use pre-trained word embedding instead of random initialized embedding. If using BERT and Elmo additionally, we project them into representation vectors of same dimension as word embedding and concatenate

Other embeddings, i.e., POS embedding, linguistic knowledge embedding, and so on are initialized in same way as random word embedding no matter in *closed* or *open* setting.

Syntactic Parsers In Table 1, both AUTO and GOLD syntactic dependencies are provided by the dataset. Since the performance of the AUTO is far behind the state-of-the-art BiaffineParser (Dozat and Manning, 2016), we generate more dependency results by training BiaffineParser[6] with different external knowledge, including pre-trained word embedding and BERT. Performance for different parsers is listed in Table 1.

Parameters In this work, we set word embedding size $d_w = 100$, POS embedding size $d_t = 50$. The predicate embedding size is set as $d_p = 100$. The syntax-related embedding size varies along with different configurations, so as the feature embedding size d_f.

To facilitate residual connections, all sublayers in the model produce outputs of dimension $d_{model} = d_f + d_p$. The hidden dimension $d_{ff} = 800$ is applied for all the experiments. We set the number of shared self-attention blocks $N = 10$. The number of heads varies with d_{model}, but dimension of each head is 25. Besides, LISA incorporates syntax knowledge in the 5-th self-attention layer while RELAWE incorporates in the first 5 layers.

We apply the similar dropout strategy as Vaswani et al. (2017), i.e., the attention and residual dropout values are 0.2 and 0.3 respectively. The dropout is also applied in the middle layer of FFN with value 0.2. We also employ label smoothing (Szegedy et al., 2015) of value 0.1 during training.

We use softmax-cross-entropy as our loss function, and use the Adadelta optimizer (Zeiler, 2012) with $\epsilon = 10^{-6}$ and $\rho = 0.95$. For all experiments, we train the model $200,000$ steps with learning rate $lr = 1.0$, and each batch has 4096 words.

All the hyper-parameters are tuned on the development set.

Configurations We use different abbreviations to represent the parsing results, syntactic dependency representations, and incorporation methods. All the system configurations in our experiments are listed in Table 2.

them with other input.

[6]We split train data into 5-fold, and train model with 4-fold to generate automatic trees of the left 1-fold train data.

Abbreviation	Description
NONE	No syntactic knowledge
Syn. Parser	
AUTO	Parsing result from CoNLL09
AUTODEL	Remove syntax errors of AUTO
BIAFFINE	BiaffineParser result
BIAFFINEBERT	BiaffineParser with BERT
GOLD	Gold syntax from CoNLL09
Syn. Representation	
DEP	Dependency head
REL	Dependency relation label
DEPPATH	Tree-based position feature
RELPATH	Shortest relation label path
Incorporation	
INPUT	Add to the input embedding
LISA	From Strubell et al. (2018)
RELAWE	Relation-aware self-attention

Table 2: A glossary of abbreviations for different system configurations in our experiments.

	P	R	F1
NONE	83.97	82.94	83.45
AUTO	85.92	84.62	85.26
BIAFFINE	86.04	85.43	85.73
BIAFFINEBERT	87.32	87.10	**87.21**
AUTODEL	88.33	87.67	88.00
GOLD	91.00	91.43	91.22

Table 3: SRL results with dependency trees of different quality on the Chinese dev set. These experiments are conducted on the RELAWE model with DEP&REL representations.

4.2 Quality of the Syntactic Dependencies

We use the above-mentioned dependency trees of different quality for comparison, with DEP&REL representation on our RELAWE model. In addition, we generate one more data AUTODEL by deleting all the erroneous dependency heads and relations from the provided AUTO data according to the gold heads and relations, and we do not replace them with any alternative heads and relations[7]. We take this setting as another reference (along with GOLD) to indicate that erroneous syntax information may hurt the performance of the SRL model. We take the GOLD as the upperbound reference of our task setting. Experiment results in Table 3 demonstrate that, incorporating syntactic knowledge into the SRL model can achieve better performance and overall, the better the quality is, the better the SRL model performs. This is consistent with the previous study by He et al. (2017) on the English dataset.

[7]For the AUTODEL data, we cannot guarantee that there exists a syntactic path between the two words. Therefore, we do not conduct experiments under the DepPath&RelPath setting.

	P	R	F1
AUTO			
DEPPATH&RELPATH	84.76	81.85	83.28
BIAFFINE			
DEP	84.33	84.47	84.40
REL	85.84	85.23	85.54
DEP&REL	86.04	85.43	85.73
DEPPATH	85.48	84.17	84.82
RELPATH	86.85	83.92	85.36
DEPPATH&RELPATH	86.40	85.52	**85.96**
GOLD			
DEPPATH&RELPATH	92.20	92.53	92.37

Table 4: SRL results with different syntactic representations on the Chinese dev set. Experiments are conducted on the RELAWE method.

Closer observation reveals two additional interesting phenomena. Firstly, SRL performance improvement is not proportionate to the improvement of dependency quality. When switching syntactic dependency trees from AUTO to BIAFFINE, SRL performance improves 0.5%, although syntactic dependency improves about 8%. In contrast, the difference between BIAFFINE and BIAFFINEBERT shows more significant improvement of 1.5%. The possible reason is that BIAFFINEBERT provides key dependency information which is missing in other configurations. Secondly, the SRL performance gap between AUTODEL and AUTO is large though they provide the same correct syntactic information. This may indicate that incorporating erroneous syntactic knowledge hurts the SRL model, and even providing more correct dependencies cannot make up for the harm (cf. BIAFFINEBERT).

4.3 Representation of the Syntactic Dependencies

Apart from DEP and REL, we also use DEPPATH and RELPATH to encode the syntactic knowledge. In this subsection, we conduct experiments to compare different syntactic encoding in our SRL model. We base the experiments on our RELAWE model, since it is easier to incorporate different representations for comparison. When generating the RELPATH, we filter the paths 1) when the dependency distance between the predicate and the candidate argument is more than 4, and 2) when the RELPATH's frequency is less than 10[8].

No matter in which representation, dependency label information is more important than the head and the combination of the two achieves better performance as our experiment results in Table 4 show. Furthermore, using BIAFFINE dependency trees, DEPPATH and RELPATH perform better than DEP and REL. This is because of the capability of DEPPATH and RELPATH to capture more structural information of the dependency trees.

Comparing Table 3 and 4, when using gold dependencies, DEPPATH&RELPATH can achieve much better result than DEP&REL. But with the AUTO trees, DEPPATH&RELPATH is much worse. Therefore, structural information is much more sensitive to the quality of dependency trees due to error propagation.

[8]As we know, dependency parsers cannot deal with long distance dependency well and it is unlikely to deliver a reliable result. And our experiments show that filtration achieves empirically better results.

	P	R	F1
INPUT			
DEP	83.89	83.61	83.75
DEP&REL	86.21	85.00	85.60
DEP&RELPATH	86.01	85.38	85.69
DEPPATH&RELPATH	85.84	85.54	**85.69**
LISA			
DEP	84.68	85.38	85.03
DEP&REL	85.56	85.89	85.73
DEP&RELPATH	85.84	85.64	**85.74**
DEPPATH&RELPATH[9]	na	na	na
RELAWE			
DEP	84.33	84.47	84.40
DEP&REL	86.04	85.43	85.73
DEP&RELPATH	86.21	85.01	85.60
DEPPATH&RELPATH	86.40	85.52	**85.96**

Table 5: SRL results with different incorporation methods of the syntactic information on the Chinese dev set. Experiments are conducted on the BIAFFINE parsing results.

	P	R	F1
BIAFFINE			
RANDOM	86.40	85.52	85.96
GIGA	86.73	85.58	86.15
ELMO	86.84	86.74	86.79
BERT	88.16	89.57	**88.86**
BIAFFINEBERT			
BERT	88.05	89.65	88.84

Table 6: SRL results with different external knowledge on the Chinese dev set. We use the RELAWE model and DEPPATH&RELPATH syntax representation.

4.4 Incorporation Methods

This subsection discusses the effectiveness of different incorporation methods of the syntactic knowledge. We take BIAFFINE's output as our de-

[9]From the mechanism of LISA, we can find that the replaced attention head can't copy the syntactic dependency heads from DEPPATH.

Chinese	P	R	F1
NONE	81.99	80.65	81.31
Closed			
CoNLL09 SRL Only	na	na	78.6
INPUT(DEPPATH&RELPATH)	84.19	83.65	83.92
LISA(DEP&RELPATH)	83.84	83.54	83.69
RELAWE(DEPPATH&RELPATH)	84.77	83.68	**84.22**
Open			
Marcheggiani and Titov (2017)	na	na	82.5
Cai et al. (2018)	84.7	84.0	84.3
INPUT(DEPPATH&RELPATH) + BERT	86.89	87.75	87.32
LISA(DEP&RELPATH) + BERT	86.45	87.90	87.17
RELAWE(DEPPATH&RELPATH) + BERT	86.73	87.98	**87.35**
GOLD	91.93	92.36	92.14

Table 7: SRL results on the Chinese test set. We choose the best settings for each configuration of our model.

pendency information for the comparison.

Firstly, results in Table 5 show that with little dependency information (DEP), LISA performs better, while incorporating richer syntactic knowledge (DEP&REL or DEP&RELPATH), three methods achieve similar performance. Overall, RELAWE achieves best results given enough syntactic knowledge.

Secondly, INPUT and LISA achieve much better performance when we combine the dependency head information and the relation, while Strubell et al. (2018) have not introduced relation information to the LISA model and Xia et al. (2019) have not combined the head and relation information either. Our proposed RELAWE method with DEPPATH&RELPATH representation performs the best, which encodes the richest syntactic knowledge.

Lastly, under the same settings, LISA and RELAWE perform better than INPUT, which indicates the importance of the location where the model incorporates the syntax, the input layer vs. the encoder layer.

4.5 External Resources

Apart from the experiments with syntactic knowledge itself, we also compare different external resources to discover their relationship with the syntax, including pre-trained word embeddings, ELMo, and BERT. We conduct experiments with our best setting, the RELAWE model with DEPPATH & RELPATH and the results are listed in Table 6.

The plain word embedding improves a little in such settings with syntactic information, while for the newly proposed ELMO and BERT, both of them can boost the models further.

4.6 Final Results on the Chinese Test Data

Based on the above experiments and analyses, we present the overall results of our model in this subsection. We train the three models (INPUT, LISA, and RELAWE) with their best settings without any external knowledge as CLOSED, and we take the same models with BERT as OPEN. The DEPPATH&RELPATH from GOLD without external knowledge serves as the GOLD for reference. Since we have been focusing on the task of argument identification and labeling, for both CLOSED and OPEN, we follow Roth and Lapata (2016) to use existing systems' predicate senses (Johansson and Nugues, 2008) to exclude them from comparison.

Table 7 shows that our OPEN model achieves more than 3 points of f1-score than the state-of-the-art result, and RELAWE with DEPPATH&RELPATH achieves the best in both CLOSED and OPEN settings. Notice that our best CLOSED model can almost perform as well as the state-of-the-art model while the latter utilizes pre-trained word embeddings. Besides, performance gap between three models under OPEN setting is very small. It indicates that the representation ability of BERT is so powerful and may contains rich syntactic information. At last, the GOLD result is much higher than the other models, indicating that there is still large space for improvement for this task.

English	P	R	F1
LISA(DEP)[10]	89.26	85.46	87.32
LISA(DEP&RELPATH)	89.74	85.38	87.51
INPUT(DEPPATH)	89.33	85.60	87.42
INPUT(DEPPATH&RELPATH)	89.70	85.48	87.54
RELAWE(DEPPATH&RELPATH)	89.55	85.92	**87.70**

Table 8: SRL results on the English test set. We use syntactic dependency results generated by BiaffineParser (On test set, syntactic performance is: UAS = 94.35%, and LAS = 92.54%, which improves about 6% compared to automatic trees in CoNLL-2009.).

4.7 Results on the English Data

We also conduct several experiments on the English dataset to validate the effectiveness of our approaches on other languages than Chinese and the results are in Table 8. Although both configurations are not exactly the same as their original papers, we tried our best to reproduce their methods on the CoNLL2009 dataset for our comparison. Overall, the results are consistent with the Chinese experiments, while the improvement is not as large as the Chinese counterparts. The RELAWE model with DEPPATH&RELPATH still achieves the best performance. Applying our syntax-enhanced model to more languages will be an interesting research direction to work on in the future.

5 Conclusion and Future Work

This paper investigates how to incorporate syntactic dependency information into semantic role labeling in depth. Firstly, we confirm that dependency trees of better quality are more helpful for the SRL task. Secondly, we present different ways to encode the trees and the experiments show that keeping more (correct) structural information during encoding improves the SRL performance. Thirdly, we compare three incorporation methods and discover that our proposed relation-aware self-attention-based model is the most effective one.

Although our experiments are primarily on the Chinese dataset, the approach is largely language independent. Apart from our tentative experiments on the English dataset, applying the approach to other languages will be an interesting research direction to work on in the future.

[10]We reimplement LISA in Strubell et al. (2018) as LISA(DEP), and Xia et al. (2019)'s best DEPPATH approach as INPUT(DEPPATH). Therefore, we can compare with their work as fairly as possible. Other settings are the best configurations for their corresponding methods.

References

Omri Abend and Ari Rappoport. 2017. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 77–89, Vancouver, Canada. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. abs/1607.06450.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 86–90, Stroudsburg, PA, USA. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Aljoscha Burchardt, Nils Reiter, Stefan Thater, and Anette Frank. 2007. A semantic approach to textual entailment: System evaluation and task analysis. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, RTE '07, pages 10–15, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntax-agnostic over syntax-aware? *CoRR*, abs/1808.03815.

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*, pages 152–164, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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

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, Portugal. Association for Computational Linguistics.

Pierre-Etienne Genest and Guy Lapalme. 2011. Framework for abstractive summarization using text-to-text generation. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 64–73, Portland, Oregon. Association for Computational Linguistics.

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

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. Association for Computational Linguistics.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 364–369.

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*, pages 473–483. Association for Computational Linguistics.

Richard Johansson and Pierre Nugues. 2008. The effect of syntactic representation on semantic role labeling. In *COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK*, pages 393–400.

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

Zuchao Li, Shexia He, Jiaxun Cai, Zhuosheng Zhang, Hai Zhao, Gongshen Liu, Linlin Li, and Luo Si. 2018. A unified syntax-aware framework for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2401–2411.

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. Association for Computational Linguistics.

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(EMNLP2017)*, pages 1506–1515, Copenhagen, Denmark. Association for Computational Linguistics.

A. Meyers, R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. 2004. The nombank project: An interim report. In *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31, Boston, Massachusetts, USA. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinkova, Dan Flickinger, Jan Hajic, and Zdenka Uresova. 2015. Semeval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926, Denver, Colorado. Association for Computational Linguistics.

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

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, 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 2018*, pages 2227–2237.

Feng Qian, Lei Sha, Baobao Chang, LuChen Liu, and Ming Zhang. 2017. Syntax aware lstm model for semantic role labeling. In *Proceedings of the 2nd Workshop on Structured Prediction for Natural Language Processing*, pages 27–32, Copenhagen, Denmark. Association for Computational Linguistics.

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*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 464–468.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical*

Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 12–21, Prague, Czech Republic. Association for Computational Linguistics.

R. A. Sinoara, R. G. Rossi, and S. O. Rezende. 2016. Semantic role-based representations in text classification. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2313–2318.

Asher Stern and Ido Dagan. 2014. Recognizing implied predicate-argument relationships in textual inference. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 739–744, Baltimore, Maryland. Association for Computational Linguistics.

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, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 8–15, Sapporo, Japan. Association for Computational Linguistics.

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 Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England. Coling 2008 Organizing Committee.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2017. Deep semantic role labeling with self-attention.

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,*, pages 4929–4936.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6000–6010. Curran Associates, Inc.

Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2011. Ontonotes: A large training corpus for enhanced processing. *Handbook of Natural Language Processing and Machine Translation. Springer*.

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on universal dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas. Association for Computational Linguistics.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019. Syntax-aware neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

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*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701.

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*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.