Span-Based Semantic Role Labeling with Argument Pruning and Second-Order Inference

Zixia jia[*1,2,3,4], **Zhaohui Yan**[*1,2,3,4], **Haoyi Wu**[1], **Kewei Tu**[†1,2,3,4]

[1]School of Information Science and Technology, ShanghaiTech University
[2]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[3]Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences
[4]University of Chinese Academy of Sciences
{jiazx, yanzh, wuhy1, tukw}@shanghaitech.edu.cn

Abstract

We study graph-based approaches to span-based semantic role labeling. This task is difficult due to the need to enumerate all possible predicate-argument pairs and the high degree of imbalance between positive and negative samples. Based on these difficulties, high-order inference that considers interactions between multiple arguments and predicates is often deemed beneficial but has rarely been used in span-based semantic role labeling. Because even for second-order inference, there are already $O(n^5)$ parts for a sentence of length n, and exact high-order inference is intractable. In this paper, we propose a framework consisting of two networks: a predicate-agnostic argument pruning network that reduces the number of candidate arguments to $O(n)$, and a semantic role labeling network with an optional second-order decoder that is unfolded from an approximate inference algorithm. Our experiments show that our framework achieves significant and consistent improvement over previous approaches.

Introduction

The task of semantic role labeling (SRL) aims to recognize the predicate-argument structure of each predicate in a sentence. Lots of previous work shows that the shallow semantic structures found by SRL are beneficial to many downstream natural language processing tasks, especially those involving text understanding, such as machine reading comprehension (Wang et al. 2015; Zhang et al. 2020), question answering (Eckert and Neves 2018; Khashabi et al. 2018) and machine translation (Marcheggiani, Bastings, and Titov 2018).

There are two annotation types of SRL, namely dependency-based SRL where each argument is a single token and span-based SRL where each argument is a span that consists of one or more words. Span-based SRL is more difficult than dependency-based SRL because it needs to not only label semantic roles but also detect argument boundaries. In this paper, we focus on span-based SRL.

Two main classes of approaches to span-based SRL are sequence labeling approaches (Yang and Mitchell 2017; He et al. 2017; Tan et al. 2018; Strubell et al. 2018) and graph-based approaches (He et al. 2018; Li et al. 2019; Ouchi, Shindo, and Matsumoto 2018; Marcheggiani and Titov 2020). Sequence labeling approaches often employ the BIO tagging scheme and predict a BIO tag for each word in the input sentence. In comparison, graph-based approaches directly predict predicate-argument pairs, which can better leverage span-level features. Besides, graph-based approaches can handle multiple predicates in the same sentence, while sequence labeling approaches cannot.

There are two major challenges faced by graph-based approaches. The first is that we need to enumerate all possible predicate-argument pairs, resulting in $O(n^3)$ complexity, which is too computationally expensive. The second is that a sentence typically contains only a few predicate-argument pairs among $O(n^3)$ word-span pairs, leading to a severe data imbalance problem in training and low recall in prediction. These two challenges can be addressed with argument pruning, which tries to remove unlikely arguments and produce a much smaller and more balanced candidate argument set for subsequent SRL prediction. Previous work trains the argument pruner jointly with the SRL predictor (He et al. 2018; Li et al. 2019). However, joint training is problematic in that the SRL training loss is defined over candidate arguments produced by the pruner and hence when the pruner changes during training, so does the SRL training loss, leading to a moving target for training. In this paper, we propose to train argument pruning independently from SRL prediction, leading to more stable training and better SRL performance. We also propose a new argument pruning model that empirically outperforms previous methods.

Utilizing high-order information has been proved beneficial on many tasks such as dependency parsing (Wang, Huang, and Tu 2019; Wang and Tu 2020; Zhang, Li, and Zhang 2020) and dependency-based SRL (Lyu, Cohen, and Titov 2019; Chen, Lyu, and Titov 2019; Li et al. 2020b). It is often deemed beneficial to SRL prediction. For example, in the sentence "*We watch the musician who is respected by many people playing violin on TV*", "*musician*" is far from "*playing*" while "*people*" is adjacent to "*playing*", which may confuse an SRL model, but if we know "*violin*" is the argument (of role Arg1) of predicate "*play*", then "*musician*" becomes a more likely argument (of role Arg0) of "*play*" than "*people*". However, high-order inference on span-based SRL is extremely difficult because of two challenges. First,

[*]These authors contributed equally.
[†]Corresponding author.

for a sentence of length n, there are $O(n^2)$ possible spans, so even for second-order inference which models interactions between two related predicate-argument pairs, there are already $O(n^5)$ second-order parts. Second, even second-order inference is NP-hard. We address the first challenge by using our pruner to reduce the number of arguments to $O(n)$ and the second challenge by designing a second-order neural decoder that is unfolded from mean-field variational inference (Wang, Huang, and Tu 2019).

Our contributions can be summarized as follows: First, we propose a new argument pruning model and train it independently from SRL prediction, avoiding the drawback of previous work and achieving better pruning accuracy. Second, we are the first to propose second-order neural model of span-based SRL, which is enabled by our argument pruning model and an efficient second-order decoder. Third, we show that the combination of independent argument pruning and second-order inference achieves new state-of-the-art accuracies on span-based SRL. We provide our source code in https://github.com/JZXXX/Span-srl.

Model

Our SRL framework contains two parts: a predicate-agnostic argument pruning network (PAPN) and a semantic role labeling network (SRLN). PAPN filters out unlikely argument spans and SRLN performs SRL prediction based on the reduced argument candidate set. While many previous approaches assume that predicates are identified in the input sentence, our model works with and without pre-identified predicates. The overall architecture is shown in Figure 1.

The PAPN does not share any component with SRLN and is trained independently from SRLN. After the training of PAPN is done, we then train SRLN based on the argument set filtered by PAPN. This is different from previous work by He et al. (2018) that trains a single model for both argument pruning and SRL prediction. We empirically find that separating the two makes learning much more stable and efficient.

Predicate-Agnostic Argument Pruning Network

Given a sentence with n words $\mathbf{w} = w_1, w_2, ..., w_n$, we aim to predict candidate spans that are potential arguments of some predicate in this sentence while prune the rest of the spans.

PAPN ignores predicates and focuses only on argument identification, which makes the training and prediction much more efficient and effective. Specifically, if we consider predicates in PAPN, we would face the same challenges of SRL prediction, i.e., $O(n^3)$ computational complexity and serious data imbalance. Without considering predicates, we are able to reduce the computational complexity to $O(n^2)$ and reduce the degree of data imbalance to achieve high recall (so that the predicted candidate set contains more true arguments).

Inspired by previous work on graph-based Name Entity Recognition (Yu, Bohnet, and Poesio 2020), we cast span prediction as predicting a dependency arc from the start word to the end word of the span, and hence design our PAPN in the form of a biaffine dependency parser (Dozat and Manning 2017, 2018).

Network Architecture We concatenate pretrained word embeddings $\mathbf{e}_i^{(\text{word})}$ and character-based word embeddings $\mathbf{e}_i^{(\text{char})}$ and feed them into a multi-layer bi-directional LSTM to produce contextualized word representations. Then two single-layer feedforward neural networks (FNN) are used to produce two vectors ($\mathbf{h}_i^{(\text{start})}$ and $\mathbf{h}_i^{(\text{end})}$) for each word, one representing the word being the start word of an argument and the other representing it being the end word. Then a biaffine function is applied to score each word pair.

$$\begin{aligned}
\mathbf{x}_i &= \mathbf{e}_i^{(\text{word})} \oplus \mathbf{e}_i^{(\text{char})} \\
R &= \mathbf{BiLSTM}(X) \\
\mathbf{h}_i^{(\text{start})} &= \mathbf{FNN}^{(\text{start})}(\mathbf{r}_i) \\
\mathbf{h}_i^{(\text{end})} &= \mathbf{FNN}^{(\text{end})}(\mathbf{r}_i) \\
s_{ij} &= \mathbf{h}_i^{(\text{start})\top} W_1 \mathbf{h}_j^{(\text{end})} + b \quad (i \le j)
\end{aligned}$$

where W_1 is a square matrix, and b is a scalar.

Learning We train the PAPN by optimizing the cross-entropy loss:

$$\begin{aligned}
\mathcal{L}_{\text{PAPN}} = -\sum_{i \le j} \big(& y_{ij} \log \mathbf{Sigmoid}(s_{ij}) \\
& + (1 - y_{ij}) \log(1 - \mathbf{Sigmoid}(s_{ij})) \big)
\end{aligned} \tag{1}$$

where y_{ij} indicates whether span $(w_i, ..., w_j)$ is indeed an argument.

Pruning After training a PAPN, we rank spans by their scores s_{ij} and choose the top-K spans as candidate arguments. We set $K = \lambda n$, where λ is a hyper-parameter controlling the strength of pruning. In this way, we reduce the number of candidate arguments from $O(n^2)$ to $O(n)$.

Semantic Role Labeling Network

Given a sentence $\mathbf{w}$ and its candidate argument set $\bar{\mathcal{A}}(\mathbf{w}) = \{\bar{a}_{ij}\}$ provided by PAPN, we formulate SRL as predicting labeled edges from words (as predicates) to candidate arguments, with the edge labels indicating semantic roles. We additionally formulate predicate identification as predicting an edge with label *V* from a word to a candidate argument containing only the word[1]. In the setting of given gold predicates, we skip this additional formulation and only predict edges from gold predicates. Our SRLN contains two parallel modules for edge existence prediction and edge label prediction respectively, with a shared encoding layer.

Word Representation We concatenate pretrained word embeddings and character-based word embeddings to represent words. In the setting of given gold predicates, we additionally concatenate indicator embeddings and predicate lemma embeddings. The indicator embedding of a binary feature indicates if the word is a gold predicate or not. The predicate lemma embedding represents the lemma of the

[1]This additional formulation is not necessary because the previous formulation already identifies predicates, but we empirically find that it improves prediction accuracy.

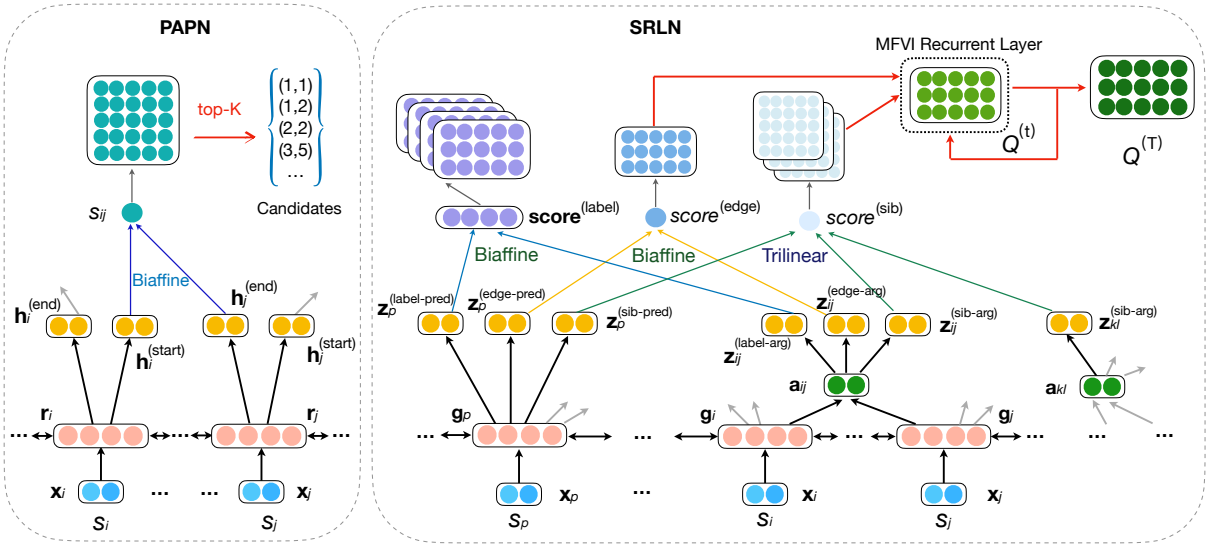


Figure 1: Overall architecture of our model. For clarity, we only show one possible predicate and one candidate argument as examples.

word if it is a gold predicate, and is a fixed arbitrary embedding if the word is not a gold predicate. We feed the word representations into a multi-layer bi-directional LSTM and denote the resulting contextualized word representations as $G = [\mathbf{g}_1; \mathbf{g}_2; \ldots; \mathbf{g}_n]$.

Span Representation We try different span embedding methods discussed by Toshniwal et al. (2020) and find that Coherent (Seo et al. 2019) has the best performance in our model. Given an argument span $\bar{a}_{ij} = (w_i, ..., w_j)$, we take the contextualized representations of its two endpoints $\mathbf{g}_i, \mathbf{g}_j \in \mathcal{R}^d$ and split each representation into four parts:

$$\mathbf{g}_i = [\mathbf{g}_i^1; \mathbf{g}_i^2; \mathbf{g}_i^3; \mathbf{g}_i^4]$$
$$\mathbf{g}_j = [\mathbf{g}_j^1; \mathbf{g}_j^2; \mathbf{g}_j^3; \mathbf{g}_j^4]$$

Where $\mathbf{g}_i^1, \mathbf{g}_j^2 \in \mathcal{R}^a$, $\mathbf{g}_i^3, \mathbf{g}_j^4 \in \mathcal{R}^b$ and $2(a+b) = d$. Then we represent this argument $\bar{a}_{ij}$ as:

$$\mathbf{a}_{ij} = [\mathbf{g}_i^1; \mathbf{g}_j^2; \mathbf{g}_i^3 \cdot \mathbf{g}_j^4] \quad (2)$$

The dot product $\mathbf{g}_i^3 \cdot \mathbf{g}_j^4$ produces a real number and is referred to as the coherence term.

First-Order Scoring We use four FNNs to encode each word w_p as a predicate and each candidate argument $\bar{a}_{ij}$ for our edge-module and label-module respectively.

$$\mathbf{z}_p^{(\text{edge-pred})} = \mathbf{FNN}^{(\text{edge-pred})}(\mathbf{g}_p)$$
$$\mathbf{z}_{ij}^{(\text{edge-arg})} = \mathbf{FNN}^{(\text{edge-arg})}(\mathbf{a}_{ij})$$
$$\mathbf{z}_p^{(\text{label-pred})} = \mathbf{FNN}^{(\text{label-pred})}(\mathbf{g}_p)$$
$$\mathbf{z}_{ij}^{(\text{label-arg})} = \mathbf{FNN}^{(\text{label-arg})}(\mathbf{a}_{ij})$$

where all the vectors are of dimension d. Then we score each potential predicate-argument pair $(w_p, \bar{a}_{ij})$ and its possible label with two biaffine functions:

$$score_{p,ij}^{(\text{edge})} = \mathbf{z}_p^{(\text{edge-pred})\top} W_2 \mathbf{z}_{ij}^{(\text{edge-arg})} + b$$
$$\mathbf{score}_{p,ij}^{(\text{label})} = \mathbf{z}_p^{(\text{label-pred})\top} \mathbf{U} \mathbf{z}_{ij}^{(\text{label-arg})} + \mathbf{b}$$

Here, $W_2 \in \mathcal{R}^{d \times d}$ and $\mathbf{U} \in \mathcal{R}^{d \times R \times d}$, where R is the size of the label (semantic role) set. To reduce the number of parameters, we follow the work of Dozat and Manning (2018) and require the tensor $\mathbf{U}$ to be diagonal (i.e., $u_{i,k,j} = 0$ wherever $i \neq j$), which has been shown to not hurt the performance.

Second-Order Scoring In order to model interactions between arguments, we score each pair of potential predicate-argument pairs that share the same predicate. Such a pair is called a *sibling* part in the dependency parsing literature. There is another type of second-order parts, *co-parent*, which consists of two predicates sharing an argument. Co-parent parts are relatively rare in span-based SRL and we do not empirically find them helpful. Without argument pruning, there would be $O(n^5)$ sibling parts, making scoring and subsequent inference infeasible in practice. However, with argument pruning by PAPN, there are only $O(n)$ arguments left and hence the number of sibling parts is reduced to $O(n^3)$. We only apply second-order scoring and inference in the edge-module, because there are typically dozens of labels (semantic roles) and we find that scoring all the labels for all the sibling parts in the label-module is too time and space consuming.

We use two FNNs to encode each word w_p as a predicate and each candidate argument $\bar{a}_{ij}$:

$$\mathbf{z}_p^{(\text{sib-pred})} = \mathbf{FNN}^{(\text{sib-pred})}(\mathbf{g}_p)$$
$$\mathbf{z}_{ij}^{(\text{sib-arg})} = \mathbf{FNN}^{(\text{sib-arg})}(\mathbf{a}_{ij})$$

where all the vectors are of dimension d. We then use a decomposed trilinear function to score a sibling part consisting of one potential predicate and two argument candidates $(w_p, \bar{a}_{ij}, \bar{a}_{kl})$.

$$\mathbf{z}'_p = \mathbf{U}_1 \mathbf{z}_p^{(\text{sib-pred})}$$
$$\mathbf{z}'_{ij} = \mathbf{U}_2 \mathbf{z}_{ij}^{(\text{sib-arg})} \quad \mathbf{z}'_{kl} = \mathbf{U}_3 \mathbf{z}_{kl}^{(\text{sib-arg})}$$
$$score_{p,ij,kl}^{(\text{sib})} \equiv score_{p,kl,ij}^{(\text{sib})} = \sum_{m=1}^{d} (\mathbf{z}'_p \circ \mathbf{z}'_{ij} \circ \mathbf{z}'_{kl})_m$$

where $\mathbf{U}_1$, $\mathbf{U}_2$ and $\mathbf{U}_3$ are three $(d \times d)$-dimensional matrices, $\circ$ means element-wise product.

Inference In first-order inference, we first try to identify all the predicate-argument pairs such that the sum of their first-order edge scores is maximized, under the constraint that the arguments of the same predicate do not overlap. We can find the exact solution using dynamic programming. We then label each predicate-argument pair by picking the highest-scoring label.

In second-order inference, we want to identify a set of predicate-argument pairs to maximize the sum of their first-order edge scores *and* the scores of all the sibling parts formed by these pairs. This is in general NP-hard. Inspired by Wang, Huang, and Tu (2019), we formulate this problem as MAP inference on a Conditional Random Field (CRF). We define boolean variable $X_{p,ij}$ to indicate whether span $\bar{a}_{ij}$ is an argument of predicate p. We define a unary potential $\phi_u(X_{p,ij})$ for each variable $X_{p,ij}$:

$$\phi_u(X_{p,ij}) = \begin{cases} \exp(score_{p,ij}^{(\text{edge})}) & X_{p,ij} = 1 \\ 1 & X_{p,ij} = 0 \end{cases}$$

For each pair of arguments $\bar{a}_{ij}$ and $\bar{a}_{kl}$, we define a binary potential $\phi_b(X_{p,ij}, X_{p,kl})$.

$$\phi_b(X_{p,ij}, X_{p,kl}) = \begin{cases} \exp(score_{p,ij,kl}^{(sib)}) & X_{p,ij} = X_{p,kl} = 1 \\ 1 & \text{Otherwise} \end{cases}$$

We can use Mean Field Variational Inference for approximate inference on this CRF. The algorithm updates the factorized posterior distribution $Q_{p,ij}(X_{p,ij})$ of each variable iteratively.

$$\mathcal{F}_{p,ij}^{(t-1)} = \sum_{k \neq i \text{ or } l \neq j} Q_{p,kl}^{(t-1)}(1) score_{p,ij,kl}^{(\text{sib})}$$
$$Q_{p,ij}^{(t)}(0) \propto 1$$
$$Q_{p,ij}^{(t)}(1) \propto \exp\{score_{p,ij}^{(\text{edge})} + \mathcal{F}_{p,ij}^{(t-1)}\}$$

At $t = 0$, $Q_{p,ij}^{(0)}(X_{p,ij})$ is initialized by normalizing the unary potential. The iterative update steps can be unfolded as recurrent neural network layers parameterized by first-order and second-order scores. After a fixed T number of iterations, we output $Q_{p,ij}^{(T)}(X_{p,ij} = 1)$, the estimated probability of a predicate-argument pair. We then use dynamic programming to identify all the predicate-argument pairs in a similar way to first-order inference.

Learning Our SRLN is trained with the following loss function using gradient descent. We use a tunable interpolation constant $\alpha \in (0, 1)$ to balance the edge-module loss and the label-module loss.

$$\mathcal{L}_{\text{SRLN}} = \alpha \mathcal{L}^{(\text{label})} + (1 - \alpha) \mathcal{L}^{(\text{edge})} \tag{3}$$

$$\mathcal{L}^{(\text{edge})} = -\sum_{p,ij} \log P(y_{p,ij}^{(\text{edge})} | \mathbf{w})$$
$$\mathcal{L}^{(\text{label})} = -\sum_{p,ij} \mathbf{1}(y_{p,ij}^{(\text{edge})}) \log P(y_{p,ij}^{(\text{label})} | \mathbf{w})$$

where $y_{p,ij}^{(\text{edge})}$ and $y_{p,ij}^{(\text{label})}$ denote gold annotation of the edge existence and label. The conditional distributions over edge existence $\hat{y}_{p,ij}^{\text{edge}}$ with our first-order and second-order inference are calculated as follows:

$$P(\hat{y}_{p,ij}^{(\text{edge})} | \mathbf{w}) = \mathbf{SoftMax}([score_{p,ij}^{(\text{edge})}; 0])$$
$$P(\hat{y}_{p,ij}^{(\text{edge})} | \mathbf{w}) = Q_{p,ij}^{(T)}(X_{p,ij})$$

The conditional distribution over the label $\hat{y}_{p,ij}^{(\text{label})}$ is:

$$P(\hat{y}_{p,ij}^{(\text{label})} | \mathbf{w}) = \mathbf{SoftMax}(\mathbf{score}_{p,ij}^{(\text{label})})$$

Experiments

Experiment settings

We experiment on the CoNLL 2005 (Carreras and Màrquez 2005) and CoNLL 2012 (Pradhan et al. 2012) English datasets following the official training-development-test split. We evaluate the performance of our model using official script on the micro-average F1 score for correctly predicting (predicate, argument span, label) tuples.

We report results of two SRL settings. In the *predicted predicates* setting, our SRLN treats each word in a sentence as a candidate predicate and predicts the existence and label of each predicate-argument pair. In order to compare with previous work, in the *gold predicates* setting, our SRLN takes gold predicates as input and only needs to find semantic roles of gold predicates. We repeat each experiment three times and report the average results.

Network configuration We use randomly initialized character-level embeddings and pretrained 300-dimensional GloVe embeddings (Pennington, Socher, and Manning 2014) for all experiments. We also follow the previous work to evaluate our model with ELMo embeddings (Peters et al. 2018). In addition, following more recent work, we replace ELMo embeddings with transformer-based pre-trained embeddings such as BERT (Devlin et al. 2019) and RoBERTa (Liu et al. 2019). We fixed these contextualized embeddings during training. We tune the hyper-parameter λ of PAPN between $\{0.8, 1.0, 1.5\}$ and find λ does not have much effect on the results because the recall of our PAPN is high. The hyper-parameter a and b of span representation in SRLN are set to 560 and 40 respectively, following the proportion of a and b set by Seo et al. (2019). We do not tune these two hyper-parameters.

Predicted predicate	WSJ	Brown	CoNLL12	Avg.
GloVe embedding	F1	F1	F1	
He et al. (2017)	81.2	68.5	76.8	75.5
He et al. (2018)	82.5	70.8	79.8	77.7
Li et al. (2019)	83.0	-	-	-
Ours (1st order)	84.29	71.78	81.61	79.23
Ours (2nd order)	**84.50**	**72.70**	**81.63**	79.57
Strubell et al. (2018)* (LISA)	83.61	71.91	80.70	78.73
Zhou, Li, and Zhao (2020)*	84.56	72.55	-	-
Contextualized embedding				
He et al. (2018)$^{\triangleright}$	86.0	76.1	82.9	81.7
Li et al. (2019)$^{\triangleright}$	86.3	76.4	83.1	81.9
Ours (1st order) + ELMo	86.56	77.29	83.98	82.61
Ours (2nd order) + ELMo	86.61	77.45	83.87	82.64
Ours (1st order) + BERT	86.63	78.75	84.16	83.18
Ours (2nd order) + BERT	86.70	78.58	84.22	83.17
Ours (1st order) + RoBERTa	86.57	79.44	84.17	83.45
Ours (2nd order) + RoBERTa	**87.03**	**79.71**	**84.33**	83.69
Strubell et al. (2018)*$^{\triangleright}$ (LISA)	86.55	78.05	83.12	82.57
Zhou, Li, and Zhao (2020)*$^{\diamond}$	87.62	80.34	-	-
Mohammadshahi et al. (2021)*$^{\diamond}$	87.57	80.53	-	-

Table 1: Results with *predicted predicates*. We do significance test ($p < 0.05$) based on Almost Stochastic Dominance (ASD) (Dror, Shlomov, and Reichart 2019) on F1 scores. Scores being boldfaced means that they are significantly better. * indicates syntax-aware models which are shown only for reference. $\triangleright$ indicates ELMo embedding and $\diamond$ indicates BERT embedding.

Learning We use the Adam and AMSGrad optimizer (Reddi, Kale, and Kumar 2018) to optimize our loss functions (Eq.1 and Eq.3). We tune the constant α in Eq.3 between $\{0.03, 0.05, 0.1, 0.2\}$ with different experiment settings and keep the values of most hyper-parameters the same as in previous work (Dozat and Manning 2018).

Main Results

Our model does not use any syntactic information, so we compare our results with previous syntax-agnostic neural models on the CoNLL 2005 (in-domain WSJ and out-of-domain Brown) and CoNLL 2012 test sets. The latest results of syntax-aware models are also listed for reference.

Results with *predicted predicates* In Table 1, we report results on the setting of *predicted predicates* with different word embeddings. With GloVe embedding, our first-order model has substantial advantages over all of the previous syntax-agnostic models on all the datasets (over 1.29% F1 on the WSJ and 0.98% on the Brown test sets, and over 1.81% F1 on the CoNLL 2012 test set), which shows the effectiveness of our framework. Our second-order model further improves the performance and is even comparable with syntax-aware models on the CoNLL05 test sets. These results demonstrate the strength of our approaches when rich features, e.g., gold predicates, contextualized embeddings, and syntactic information, are not available.

Gold predicate	WSJ	Brown	CoNLL12	Avg.
GloVe embedding	F1	F1	F1	
He et al. (2017)	83.10	72.10	81.70	78.97
Tan et al. (2018)	84.80	74.10	82.70	80.53
He et al. (2018)	83.90	73.70	82.10	79.90
Ouchi et al. (2018)	83.50	73.10	83.00	79.87
Swayamdipta et al. (2018)	-	-	83.80	-
Shi et al. (2020)	-	-	83.20	-
Ours (1st order)	86.17	75.58	84.75	82.17
Ours (2nd order)	**86.22**	**75.64**	**84.85**	82.24
Strubell et al. (2018)* (LISA)	84.64	74.55	-	-
Marcheggiani and Titov (2020)*	85.40	75.50	84.40	81.77
Zhou, Li, and Zhao (2020)*	85.84	75.72	-	-
Contextualized embedding				
Peters et al. (2018)$^{\triangleright}$	-	-	84.60	-
He et al. (2018)$^{\triangleright}$	87.40	80.40	85.50	84.43
Ouchi et al. (2018)$^{\triangleright}$	87.60	78.70	86.20	84.17
Li et al. (2019)$^{\triangleright}$	87.70	80.50	86.00	84.73
Li et al. (2020a)$^{\circ}$	88.03	79.80	86.61	84.81
Conia and Navigli (2020)$^{\diamond}$	-	-	87.30	-
Shi et al. (2020)$^{\diamond}$	-	-	85.80	-
Fei et al. (2021)$^{\circ}$	87.60	79.82	86.40	84.61
Ours (1st order) + ELMo	88.00	80.23	86.72	84.98
Ours (2nd order) + ELMo	87.91	80.29	86.72	84.97
Ours (1st order) + BERT	88.15	81.77	87.08	85.67
Ours (2nd order) + BERT	88.25	81.90	87.18	85.78
Ours (1st order) + RoBERTa	88.17	81.92	87.12	85.74
Ours (2nd order) + RoBERTa	**88.51**	**82.12**	**87.36**	86.00
Wang et al. (2019)*$^{\triangleright}$	88.20	79.30	86.40	84.63
Marcheggiani and Titov (2020)*$^{\circ}$	87.90	80.60	86.80	85.10
Zhou, Li, and Zhao (2020)*$^{\diamond}$	88.91	81.43	-	-
Fei et al. (2021)*$^{\circ}$	89.04	83.67	88.59	87.10
Mohammadshahi et al. (2021)*$^{\diamond}$	88.93	83.21	-	-

Table 2: Results with *gold predicates*. We do significance test ($p < 0.05$) based on Almost Stochastic Dominance (ASD) (Dror, Shlomov, and Reichart 2019) on F1 scores. Scores being boldfaced means that they are significantly better. * indicates syntax-aware models which are shown only for reference. $\triangleright$ indicates ELMo embedding, $\diamond$ indicates BERT embedding, and $\circ$ indicates RoBERTa embedding.

With contextualized embeddings, we can see that our first-order model still performs better on all the test sets than previous work and our second-order model still outperforms our first-order model in most cases.

Results with *gold predicates* As shown in Table 2, with GloVe embedding, both our first-order and second-order models perform much better than the syntax-agnostic state-of-the-art models on all the test sets and even outperform some of the syntax-aware models. These results again demonstrate the effectiveness of our framework when using weak word features.

Similar to the case of *predicted predicates*, when using

		CoNLL 2005		CoNLL 2012	
	λ	P	R	P	R
MLP, GloVe	1.0	14.35	99.04	15.07	98.70
	1.5	8.68	99.40	10.19	99.37
Biaffine, GloVe	1.0	20.52	99.77	22.33	99.52
	1.5	13.82	99.93	15.04	99.82
MLP, ELMo	1.0	14.43	99.61	15.14	99.21
	1.5	9.71	99.73	10.21	99.59
Biaffine, ELMo	1.0	20.53	99.86	22.37	99.72
	1.5	13.81	99.92	15.05	99.90

Table 3: Performance of PAPN with different scorers on development sets. MLP represents that we first get span representation as in Eq. 2, then feed them into a MLP to get the predicated argument scores for pruning. Biaffine represents our model.

Gold predicates	WSJ			Brown		
	P	R	F1	P	R	F1
PAPN, GloVe	86.89	85.47	86.17	77.90	73.40	75.58
w/o PAPN, GloVe	87.55	82.02	83.43	79.51	66.16	72.22
PAPN, ELMo	88.20	87.80	88.00	81.09	79.39	80.23
w/o PAPN, ELMo	87.84	85.63	86.72	81.72	76.24	78.88

Table 4: Comparison of the results of our first-order model with and without PAPN on CoNLL 2005 test sets.

contextualized embeddings, our second-order model with RoBERTa embedding performs the best.

Analysis

Effectiveness of PAPN In Table 3, we show the effectiveness of our PAPN in argument detection. With $\lambda = 1.0$, our PAPN can keep most of the gold argument spans. The recall would further increase if we increase λ, but the precision would decrease, putting more negative examples into the candidate set.

In Table 4, we compare the performance of our first-order model with and without PAPN (note that second-order inference is infeasible without PAPN). Without PAPN, the SRLN faces much more negative samples, so its recall is significantly reduced while its precision is moderately increased in some cases, leading to a lower F1 score, as shown in the results. With PAPN, our second-order method becomes feasible, and its inference is only slightly slower than our first-order method. In addition, the running time of the first-order SRLN without PAPN is around twice of that with PAPN.

Independent and joint learning In Table 5, we compare independent PAPN and SRLN training (our framework) with joint training of the two with shared embedding and LSTM layers. The table also shows results of previous state-of-the-art methods (He et al. 2018; Li et al. 2019) which perform joint training. We only show the results of our first-order model because we empirically find that the second-order model cannot converge with joint training. We can see that independent training has better performance. As to the running time, our framework can finish training in no more than one day in all the cases, but the models of He et al. (2018) and Li et al. (2019) need more than 36 hours for training on the TITAN V GPU with their provided settings.

ELMo	Predicted predicates		Gold Predicates	
	WSJ	Brown	WSJ	Brown
He et al. (2018)†	86.00	76.10	87.40	80.40
He et al. (2018) ‡	85.94	76.51	87.34	79.01
Li et al. (2019)†	86.30	76.40	87.70	80.50
Li et al. (2019)‡	85.69	75.48	87.21	78.05
JointL	86.15	76.84	87.78	79.83
Ours	86.56	77.29	88.00	80.23

Table 5: Comparison of separate and joint learning of PAPN. †: their reported results; ‡: the average results of our 5 runs using their published code and configuration (we also tune some of the hyper-parameters based on their configuration but some of the results are still below the reported results).

		WSJ		Brown	
		first	second	first	second
ELMo	w/o self-loop	86.48	86.46	77.18	77.26
	ours	86.56	86.61	77.29	77.45
GloVe	w/o self-loop	84.34	84.46	71.79	72.28
	ours	84.29	84.5	71.78	72.7

Table 6: F1 scores of our first-order and second-order models with and without predicting self-loop edge for each possible predicate.

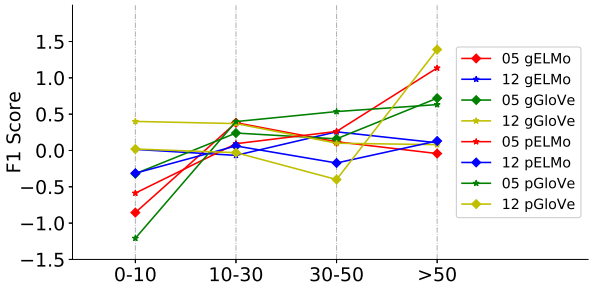
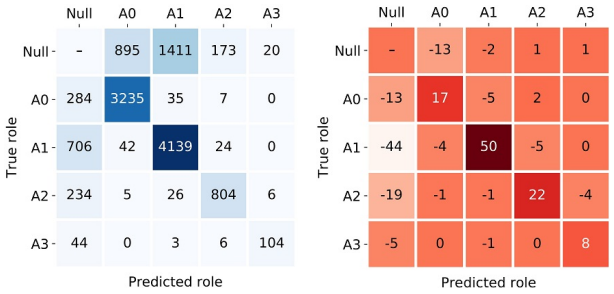


Figure 2: Relative improvements of our second-order model over our first-order model on sentences of different lengths. *g* and *p* in the legend represent the settings of *gold predicates* and *predicted predicates* respectively.

Performance on different sentence lengths We study the performance gap between our first-order and second-order models on sentences of different lengths. Figure 2 shows that advantage of our second-order model over the first-order model generally increases as test sentences become longer. One possible reason is that SRL on longer sentences is harder and hence requires more sophisticated decoding.



(a) first-order (b) error correction matrix

Figure 3: Confusion matrix of the main roles. (a) our first-order model with GloVe on WSJ. (b) our second-order model relative to the first-order model. We do not have statistics on Null-Null.

Self-loop edges We show the utility of additionally formulating predicate identification as predicting a self-loop edge with label V of each word. From Table 6, we can see that adding self-loop edges is beneficial in general, except for the first-order model with GloVe embedding.

Error correction analysis From Figure 3(a), we can see that the errors are fairly balanced in our first-order model. Figure 3(b) shows that our second-order model not only corrects the errors of the first-order model in role type predictions, but also correctly predicts arguments that are not predicted by our first-order model (as shown by the first column of the error correction matrix).

Related Work

There are two main styles of SRL annotation: PropBank-style (Palmer, Gildea, and Kingsbury 2005) and FrameNet-style (Baker, Fillmore, and Lowe 1998). We focus on PropBank-style SRL in this paper. PropBank-style SRL contains two argument annotation settings. One is span-based SRL (Carreras and Màrquez 2005; Pradhan et al. 2012) which we focus on in this work, and the other is dependency-based SRL (Hajič et al. 2009).

Recently, graph-based models achieved strong performance for their ability to leverage argument representations. Ouchi, Shindo, and Matsumoto (2018) directly took into account all possible predicate-argument pairs and scored their labels. He et al. (2018) proposed an end-to-end approach that can jointly predict predicates, arguments, and their relations. Their model was extended by Li et al. (2019) with a biaffine scorer and applied to both span-based SRL and dependency-based SRL. They both used pruning to reduce the high complexity caused by enumerating spans. Our first-order model is similar to these two approaches, but differs from them in the pruning method. Specifically, our argument pruning network is a stand-alone model with different scoring functions and independent training from the downstream SRL prediction network. We find that it makes learning more efficient and produces better performance. Pruning methods have also been explored in dependency-based SRL. He, Li, and Zhao (2019) prune arguments guided by syntactic rules. Our argument pruning method may also benefit from constituency syntactic parsing, which we leave for future work.

Enhanced models with high-order information have been proved effective by lots of previous work on syntactic dependency parsing (Ma and Zhao 2012; Gormley, Dredze, and Eisner 2015; Wang and Tu 2020; Zhang, Li, and Zhang 2020). In the semantic analysis field, Wang, Huang, and Tu (2019) considered three types of second-order parts of semantic dependencies and approximate decoding with mean field variational inference or loopy belief propagation. Recently, there are two approaches that modeled interactions between argument labeling decisions on dependency-based SRL using structure refinement. Lyu, Cohen, and Titov (2019) proposed a refinement network that iteratively refines an output produced by a factorized SRL model and designed a restricted network architecture capturing non-local interactions. Chen, Lyu, and Titov (2019) used a capsule network which models each proposition as a tuple of capsules, one capsule per argument type. The capsules interact with each other and with representations of words at each layer, and based on these capsules, predictions about the SRL structure are updated iteratively. Besides these approaches, Li et al. (2020b) applied second-order decoding to dependency-based SRL. Our work differs from these previous approaches in that we consider span-based SRL and therefore model interactions between predicate-span pairs rather than interactions between predicate-word pairs. Consequently, the computational complexity of second-order decoding is much higher, which necessitates effective pruning.

Our approximate second-order inference algorithm is inspired by Wang, Huang, and Tu (2019), whose techniques were also employed by Li et al. (2020b) for dependency-based SRL. One difference between our method and theirs is that we use self-loop edges to indicate predicates and thus integrate predicate prediction and predicate-argument prediction within the same high-order inference process. In contrast, Li et al. (2020b) predicted predicates with a sequence labeling process that is independent of their predicate-argument labeling process.

Conclusion

In this paper, we propose a novel graph-based approach to span-based SRL. Our approach consists of a predicate-agnostic argument pruning network and an SRL network. The argument pruning network reduces the number of candidate arguments to $O(n)$. The SRL network can perform both first-order decoding and second-order decoding using recurrent neural networks unfolded from mean-field variational inference. Our experiments show that our approach achieves significant and consistent improvement over previous approaches.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (61976139).

References

Baker, C. F.; Fillmore, C. J.; and Lowe, J. B. 1998. The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, 86–90.

Carreras, X.; and Màrquez, L. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the ninth conference on computational natural language learning (CoNLL-2005)*, 152–164.

Chen, X.; Lyu, C.; and Titov, I. 2019. Capturing Argument Interaction in Semantic Role Labeling with Capsule Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5415–5425. Hong Kong, China: Association for Computational Linguistics.

Conia, S.; and Navigli, R. 2020. Bridging the Gap in Multilingual Semantic Role Labeling: a Language-Agnostic Approach. In *Proceedings of the 28th International Conference on Computational Linguistics*, 1396–1410. Barcelona, Spain (Online): International Committee on Computational Linguistics.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Dozat, T.; and Manning, C. D. 2017. Deep biaffine attention for neural dependency parsing. *ICLR*.

Dozat, T.; and Manning, C. D. 2018. Simpler but More Accurate Semantic Dependency Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 484–490. Melbourne, Australia: Association for Computational Linguistics.

Dror, R.; Shlomov, S.; and Reichart, R. 2019. Deep dominance-how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2773–2785.

Eckert, F.; and Neves, M. 2018. Semantic role labeling tools for biomedical question answering: a study of selected tools on the BioASQ datasets. In *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*, 11–21. Brussels, Belgium: Association for Computational Linguistics.

Fei, H.; Wu, S.; Ren, Y.; Li, F.; and Ji, D. 2021. Better Combine Them Together! Integrating Syntactic Constituency and Dependency Representations for Semantic Role Labeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 549–559. Online: Association for Computational Linguistics.

Gormley, M. R.; Dredze, M.; and Eisner, J. 2015. Approximation-aware dependency parsing by belief propagation. *Transactions of the Association for Computational Linguistics*, 3: 489–501.

Hajič, J.; Ciaramita, M.; Johansson, R.; Kawahara, D.; Martí, M. A.; Màrquez, L.; Meyers, A.; Nivre, J.; Padó, S.; Štěpánek, J.; Straňák, P.; Surdeanu, M.; Xue, N.; and Zhang, Y. 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*, 1–18. Boulder, Colorado: Association for Computational Linguistics.

He, L.; Lee, K.; Levy, O.; and Zettlemoyer, L. 2018. Jointly Predicting Predicates and Arguments in Neural Semantic Role Labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 364–369. Melbourne, Australia: Association for Computational Linguistics.

He, L.; Lee, K.; Lewis, M.; and Zettlemoyer, L. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 473–483.

He, S.; Li, Z.; and Zhao, H. 2019. Syntax-aware Multilingual Semantic Role Labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5350–5359. Hong Kong, China: Association for Computational Linguistics.

Khashabi, D.; Khot, T.; Sabharwal, A.; and Roth, D. 2018. Question answering as global reasoning over semantic abstractions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Li, T.; Jawale, P. A.; Palmer, M.; and Srikumar, V. 2020a. Structured Tuning for Semantic Role Labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 8402–8412. Online: Association for Computational Linguistics.

Li, Z.; He, S.; Zhao, H.; Zhang, Y.; Zhang, Z.; Zhou, X.; and Zhou, X. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6730–6737.

Li, Z.; Zhao, H.; Wang, R.; and Parnow, K. 2020b. High-order Semantic Role Labeling. *arXiv preprint arXiv:2010.04641*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Lyu, C.; Cohen, S. B.; and Titov, I. 2019. Semantic Role Labeling with Iterative Structure Refinement. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 1071–1082. Hong Kong, China: Association for Computational Linguistics.

Ma, X.; and Zhao, H. 2012. Fourth-order dependency parsing. In *Proceedings of COLING 2012: posters*, 785–796.

Marcheggiani, D.; Bastings, J.; and Titov, I. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. *Proceedings of the 2018 Conference of*

the North American Chapter of the Association for Computational Linguistics.

Marcheggiani, D.; and Titov, I. 2020. Graph Convolutions over Constituent Trees for Syntax-Aware Semantic Role Labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 3915–3928. Online: Association for Computational Linguistics.

Mohammadshahi, A.; and Henderson, J. 2021. Syntax-Aware Graph-to-Graph Transformer for Semantic Role Labelling. arXiv:2104.07704.

Ouchi, H.; Shindo, H.; and Matsumoto, Y. 2018. A Span Selection Model for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1630–1642. Brussels, Belgium: Association for Computational Linguistics.

Palmer, M.; Gildea, D.; and Kingsbury, P. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1): 71–106.

Pennington, J.; Socher, R.; and Manning, C. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. Doha, Qatar: Association for Computational Linguistics.

Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237. New Orleans, Louisiana: Association for Computational Linguistics.

Pradhan, S.; Moschitti, A.; Xue, N.; Uryupina, O.; and Zhang, Y. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, 1–40.

Reddi, S. J.; Kale, S.; and Kumar, S. 2018. On the Convergence of Adam and Beyond. In *International Conference on Learning Representations*.

Seo, M.; Lee, J.; Kwiatkowski, T.; Parikh, A.; Farhadi, A.; and Hajishirzi, H. 2019. Real-Time Open-Domain Question Answering with Dense-Sparse Phrase Index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4430–4441. Florence, Italy: Association for Computational Linguistics.

Strubell, E.; Verga, P.; Andor, D.; Weiss, D.; and McCallum, A. 2018. Linguistically-Informed Self-Attention for Semantic Role Labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 5027–5038. Brussels, Belgium: Association for Computational Linguistics.

Swayamdipta, S.; Thomson, S.; Lee, K.; Zettlemoyer, L.; Dyer, C.; and Smith, N. A. 2018. Syntactic Scaffolds for Semantic Structures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3772–3782. Brussels, Belgium: Association for Computational Linguistics.

Tan, Z.; Wang, M.; Xie, J.; Chen, Y.; and Shi, X. 2018. Deep semantic role labeling with self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Toshniwal, S.; Shi, H.; Shi, B.; Gao, L.; Livescu, K.; and Gimpel, K. 2020. A Cross-Task Analysis of Text Span Representations. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, 166–176. Online: Association for Computational Linguistics.

Wang, H.; Bansal, M.; Gimpel, K.; and McAllester, D. 2015. Machine comprehension with syntax, frames, and semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 700–706.

Wang, X.; Huang, J.; and Tu, K. 2019. Second-Order Semantic Dependency Parsing with End-to-End Neural Networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4609–4618. Florence, Italy: Association for Computational Linguistics.

Wang, X.; and Tu, K. 2020. Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, 93–99. Suzhou, China: Association for Computational Linguistics.

Wang, Y.; Johnson, M.; Wan, S.; Sun, Y.; and Wang, W. 2019. How to Best Use Syntax in Semantic Role Labelling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 5338–5343. Florence, Italy: Association for Computational Linguistics.

Yang, B.; and Mitchell, T. 2017. A Joint Sequential and Relational Model for Frame-Semantic Parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1247–1256. Copenhagen, Denmark: Association for Computational Linguistics.

Yu, J.; Bohnet, B.; and Poesio, M. 2020. Named Entity Recognition as Dependency Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6470–6476. Online: Association for Computational Linguistics.

Zhang, Y.; Li, Z.; and Zhang, M. 2020. Efficient Second-Order TreeCRF for Neural Dependency Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3295–3305. Online: Association for Computational Linguistics.

Zhang, Z.; Wu, Y.; Zhao, H.; Li, Z.; Zhang, S.; Zhou, X.; and Zhou, X. 2020. Semantics-aware bert for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 9628–9635.

Zhou, J.; Li, Z.; and Zhao, H. 2020. Parsing All: Syntax and Semantics, Dependencies and Spans. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4438–4449. Online: Association for Computational Linguistics.