End-to-end Semantic Role Labeling with Neural Transition-based Model

Hao Fei,[1] Meishan Zhang,[2] Bobo Li,[1] Donghong Ji[1]*

[1] Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China
[2] Department of School of New Media and Communication, Tianjin University, Tianjin, China
{hao.fei, boboli, dhji}@whu.edu.cn, mason.zms@gmail.com

Abstract

End-to-end semantic role labeling (SRL) has been received increasing interest. It performs the two subtasks of SRL: predicate identification and argument role labeling, jointly. Recent work is mostly focused on graph-based neural models, while the transition-based framework with neural networks which has been widely used in a number of closely-related tasks, has not been studied for the joint task yet. In this paper, we present the first work of transition-based neural models for end-to-end SRL. Our transition model incrementally discovers all sentential predicates as well as their arguments by a set of transition actions. The actions of the two subtasks are executed mutually for full interactions. Besides, we suggest high-order compositions to extract non-local features, which can enhance the proposed transition model further. Experimental results on CoNLL09 and Universal Proposition Bank show that our final model can produce state-of-the-art performance, and meanwhile keeps highly efficient in decoding. We also conduct detailed experimental analysis for a deep understanding of our proposed model.

Introduction

Semantic role labeling (SRL), as one of the core tasks to identify the semantic predicates in text as well as their semantic roles, has sparked much interest in natural language processing (NLP) community (Pradhan et al. 2005; Lei et al. 2015; Xia et al. 2019). SRL is a shallow semantic parsing, aiming to uncover the predicate-argument structures, such as '*who did what to whom, when and where*', The task can be beneficial for a range number of downstream tasks, such as information extraction (Christensen et al. 2011; Bastianelli et al. 2013), question answering (Shen and Lapata 2007; Berant et al. 2013) and machine translation (Xiong, Zhang, and Li 2012; Shi et al. 2016).

Traditionally, SRL is accomplished via two pipeline steps: predicate identification (Scheible 2010) and argument role labeling (Pradhan et al. 2005). More recently, there is growing interest in end-to-end SRL, which aims to achieve both two subtasks by a single model (He et al. 2018). Given a sentence, the goal is to recognize all possible predicates together with their arguments jointly. Figure 1 shows an example of end-to-end SRL. The end-to-end joint architecture can

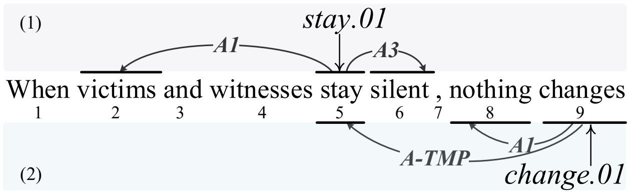


Figure 1: An example of end-to-end SRL, where two propositions are shown in one sentence.

greatly alleviate the error propagation problem, and meanwhile simplify the overall decoding process, thus receives increasing attention. Graph-based models have been the mainstream methods to end-to-end SRL, which are achieved by enumerating all the possible predicates and their arguments exhaustively (He et al. 2018; Cai et al. 2018; Li et al. 2019). Their results show that end-to-end modeling can obtain better SRL performance.

Alternatively, the transition-based framework offers another solution for end-to-end modeling, which is totally orthogonal to the graph-based models. Transition-based models have been widely exploited for end-to-end sequence labeling (Zhang and Clark 2010; Lyu, Zhang, and Ji 2016; Zhang, Zhang, and Fu 2018), structural parsing (Zhou et al. 2015; Dyer et al. 2015; Yuan, Jiang, and Tu 2019) and relation extraction (Wang et al. 2018; Zhang et al. 2019), which are closely related to SRL. These models can also achieve very competitive performances for a range of tasks, and meanwhile maintain high efficiencies with linear-time decoding complexity.

In this work, we present the first work of exploiting the neural transition-based architecture to end-to-end SRL. The model handles SRL incrementally by predicting a sequence of transition actions step by step, which are used to detect all predicates as well as their semantic roles in a given sentence. The two subtasks of end-to-end SRL, predicate identification and argument role labeling, are performed mutually in a single model to make full interactions of them. For argument role labeling, the recognition is conducted in a close-first way, where the near-predicate roles are processed first. The partial outputs of the incremental processing are denoted as transition states. In addition, we suggest explicit high-order compositions to enhance our transition-based model, lever-

*Corresponding author

aging the precedent partially-recognized argument-predicate structures for the current action classification.

Our neural transition system is built upon standard embedding-based word representations, and then is improved with dependency-aware representations by using recursive TreeLSTM (Tai, Socher, and Manning 2015). Concretely, we embed the surface words, characters, POS tags and dependency structures as input representations. During decoding, we represent the transition states by using standard BiLSTM (Hochreiter and Schmidhuber 1997) and Stack-LSTM (Dyer et al. 2015) to encode the elements in buffers and stacks, respectively. Finally, we predict transition actions incrementally based on the state representations.

We conduct experiments on dependency-based SRL benchmarks, including CoNLL09 (Hajič et al. 2009) for the English language, and Universal Proposition Bank (Akbik et al. 2015; Akbik and Li 2016) for seven other languages. Our end-to-end neural transition model wins the best results against the baselines, giving the state-of-the-art performances on both the predicate identification and argument role labeling, meanwhile keeping efficient on decoding. We also show that with recent contextualized word representations, e.g., ELMo (Devlin et al. 2019), BERT (Peters et al. 2018) or XLNet (Yang et al. 2019), the overall SRL performances can be further improved. In-depth analysis is conducted to uncover the important components of our final model, which can help comprehensive understanding of our model. Following we summarize our contributions:

• We fill the gap in the literature of employing neural transition-based model for end-to-end SRL. We also enhance the parsing procedure with a close-first scheme.

• We compose the high-order features (i.e., with one more predicate-role attachments from multiple predicates) in our transition framework for end-to-end SRL to model long-term substructure information explicitly.

• Our transition framework wins new state-of-the-art performances against all current graph-based methods on benchmark datasets, meanwhile being faster on decoding.

Related Work

Gildea and Jurafsky (2000) pioneer the task of semantic role labeling, as a shallow semantic parsing. Approaches for SRL can be largely divided into two folds. Earlier efforts are paid for designing hand-crafted discrete features with machine learning classifiers (Pradhan et al. 2005; Punyakanok, Roth, and Yih 2008; Zhao et al. 2009). Later, A great deal of work takes advantages of neural networks with automatic distributed features (FitzGerald et al. 2015; Roth and Lapata 2016; Marcheggiani and Titov 2017; Strubell et al. 2018; Fei, Zhang, and Ji 2020). Also it is worth noticing that many previous work shows that integrating syntactic tree structure can greatly facilitate the SRL (Marcheggiani, Frolov, and Titov 2017; Zhang, Wang, and Si 2019; Fei et al. 2020).

Prior works traditionally separate the SRL into two individual subtasks, i.e., predicate disambiguation and argument role labeling. They mainly conduct argument role labeling based on the pre-identified predicate oracle. More recently, several researches consider the end-to-end solution that handles both two subtasks by one single model. All of them employs graph-based neural model, exhaustively enumerating all the possible predicate and argument mentions, as well as their relations (He et al. 2018; Cai et al. 2018; Li et al. 2019; Fei, Ren, and Ji 2020). The distinctions among them are quite slight, mostly lying in the encoder, or relation scorer. For example, He et al. (2018) use the BiLSTM (Hochreiter and Schmidhuber 1997) as base encoder, and use a feed-forward layer as predicate-argument relation scorer, while Cai et al. (2018) employ a biaffine scorer, and Li et al. (2018) enhance the BiLSTM with highway connections.

Graph-based and transition-based models are always two mainstream approaches for structural learning. In this work, we employ a neural transition model, an algorithm that has been extensively exploited for a wide range of NLP parsing tasks, e.g., part-of-speech (POS) tagging (Zhang and Clark 2010; Lyu, Zhang, and Ji 2016), word segmentation (Zhang, Zhang, and Fu 2016, 2018), dependency parsing (Zhou et al. 2015; Dyer et al. 2015; Yuan, Jiang, and Tu 2019) and information extraction (Wang et al. 2018; Zhang et al. 2019)etc. Note that we have no bias on the graph-based and transition-based models, and both the two kinds of models have their own advantages and disadvantages. Compared with graph-based models, one big advantage of transition-based approaches is that high-order features (i.e., subtrees, partial parsed results) can be explicitly modeled conveniently by using transition states (i.e., state representation). Currently there is no work for transition-based neural SRL. Besides, transition-based model can achieve highly competitive task performances meanwhile with good efficiency, i.e., linear-time encoding complexity.

It worth noticing that Choi and Palmer (2011) employ a transition model for handling SRL. Unfortunately their work is based on discrete features, while ours is in the literature the first neural transition-based work. We also present two enhancements for the task, i.e., 1) close-first parsing scheme, and 2) high-order feature composition. Besides, our work focuses on end-to-end SRL, but Choi and Palmer (2011) assume that predicates are already given.

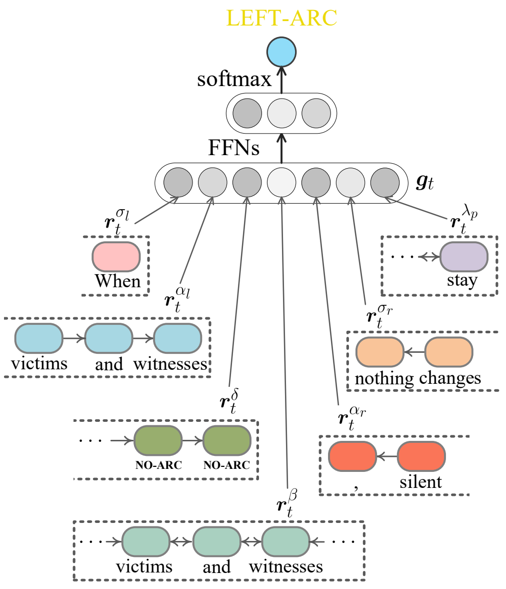
Method

Following prior end-to-end SRL work (He et al. 2018; Li et al. 2019), we treat the task as predicate-argument-role triplets prediction. Given an input sentence $S = \{w_1, \cdots, w_n\}$, the transition system is expected to output a set of triplets $Y \in P \times A \times R$, where $P = \{p_1, \cdots, p_k\}$ are all possible predicate tokens, $A = \{a_1, \cdots, a_l\}$ are all associated argument tokens, and R are the corresponding role labels for each a_i. Technically, we model Y as a graph structure, where the nodes are predicates or arguments, and the directed edges represent the roles from predicate to argument. For example the sentence in Figure 1, the final outputs are four triplets: <*stay*, *silent*, *A3*>, <*stay*, *victims*, *A1*>, <*changes*, *stay*, *A-TMP*> and <*changes*, *nothing*, *A1*>.

In this section, we first elaborate the transition system including the states and actions. We then describe the vanilla neural transition model, upon which we further introduce the high order model.

Step	Action	σ_l	α_l	λ_p	α_r	σ_r	β	Y
0	-	[]	[]	Null	[]	[9,$\cdots$,2]	[1,$\cdots$,9]	Φ
1	NO-PRD	[1]	[]	Null	[]	[9,$\cdots$,3]	[2,$\cdots$,9]	Φ
$\cdots$		*(NO-PRD)*						
4	PRD-GEN	[1,2,3,4]	[]	5^p	[]	[9,8,7,6]	[5,$\cdots$,9]	Φ
5	NO-ARC	[1,2,3]	[4]	5^p	[]	[9,8,7,6]	[5,$\cdots$,9]	Φ
6	Right-ARC	[1,2,3]	[4]	5^p	[6]	[9,8,7]	[5,$\cdots$,9]	$Y \cup \{5 \overset{\text{A3}}{\curvearrowright} 6\}$
$\cdots$		*(NO-ARC)*						
9	LEFT-ARC	[1]	[4,3,2]	5^p	[6,7]	[9,8]	[5,$\cdots$,9]	$Y \cup \{2 \overset{\text{A1}}{\curvearrowleft} 5\}$
$\cdots$		*(NO-ARC)*						
12	SHIFT	[1,$\cdots$,5]	[]	Null	[]	[9,8,7]	[6,7,8,9]	
13	NO-PRD	[1,$\cdots$,6]	[]	Null	[]	[9,8]	[7,8,9]	
$\cdots$		*(NO-PRD)*						
15	PRD-GEN	[1,$\cdots$,8]	[]	9^p	[]	[]	[9]	
16	LEFT-ARC	[1,$\cdots$,7]	[8]	9^p	[]	[]	[9]	$Y \cup \{8 \overset{\text{A1}}{\curvearrowleft} 9\}$
$\cdots$		*(NO-ARC)*						
19	LEFT-ARC	[1,2,3,4]	[8,7,6,5]	9^p	[]	[]	[9]	$Y \cup \{5 \overset{\text{A-TMP}}{\curvearrowleft} 9\}$
$\cdots$		*(NO-ARC)*						
23	SHIFT	[1,$\cdots$,9]	[]	Null	[]	[]	[]	

(a) The transition system.



(b) Model architecture.

Figure 2: Illustration of the transition framework, where the input sequence is the same as that in Figure 1. For brevity, here we omit the role labeling operation, as it is performed with *LEFT/RIGHT-ARC* actions synchronously.

Transition System

Our transition framework transforms the structural learning into a sequence of action predictions. The system consists of *actions* and *states*. The actions are used to control state transitions, while the states store partially parsed results.

States. We define the transition states as $s = (\sigma_l, \alpha_l, \lambda_p, \alpha_r, \sigma_r, \beta, Y)$. σ_l and σ_r are stacks holding processed arguments in left-side (l) and right-side (r), respectively, α_l and α_r are the corresponding stacks temporarily storing the arguments popped out of σ_* ($* \in \{l,r\}$) which will be pushed back later. λ_p is a variable referring to a candidate predicate. β is a buffer loading the unprocessed words in a sentence. Y stores the output triplets.

Actions. We design total six actions as follows.

- NO-PRD, current candidate w_i is not a predicate, so moves w_i from β to σ_l while popping the top element out of σ_r.
- PRD-GEN, current candidate w_i is a predicate (i.e., p_j), so generate representation for p_j onto λ_p.
- LEFT-ARC, the top element in σ_l is a argument (i.e, a_t), so assign an arc between p_j and a_t, and pop the top element out of σ_l into α_l.
- RIGHT-ARC, the top element in σ_r is a argument (i.e, a_t), so assign an arc between p_j and a_t, and pop the top element out of σ_r into α_r.
- NO-ARC, no arc between p_j and the top element in σ_l or σ_r, so pop the top elements out of σ_l or σ_r into α_l or α_r.
- SHIFT, end of the detection of arguments for p_j. Pop all the elements out of α_* into σ_*, clear λ_p, and moves w_i from β to σ_l while popping the top element out of σ_r.

Parsing scheme. The vanilla transition framework generally process the words in a sentence incrementally from left to right, while such reading scheme may be rigid for SRL parsing. Intuitively, it is always easier for humans to first grasp the core ideas then dig into more details. To this end, we consider a *close-first* parsing procedure (Goldberg and Elhadad 2010; Cai and Lam 2019; Kurita and Søgaard 2019). We first perform recognition for any possible predicate, e.g., p_j, and then the process of detecting the arguments of p_j starts down the path from near-to-p_j to far-to-p_j, until all the arguments are determined.

Given a sentence, before conducting the transitions, the σ_r will pre-load all the words from β except the first word in reverse order. When a predicate is determined, the system starts searching for its arguments in leftward and rightward alternatively, from near to far. Once an arc is confirmed, a labeler assigns a role label for the argument. It is worth noticing that only a subset of actions are legal to a certain transition state. And invalid actions can lower encoding efficiency, and lead to bad directed graph structure. For example, a *NO-PRD* action must be followed by either *NO-PRD* or *PRD-GEN* action, and **-ARC* must starts by a *PRD-GEN* action. We thus also design some constraints for each action. Figure 2(a) shows the transition process.

The Vanilla Neural Model

Word representation. We first use the word form representation under two types of settings: $\boldsymbol{v}_i^w, \tilde{\boldsymbol{v}}_i^w$, where $\boldsymbol{v}_i^w$ is the representation initialized with a pre-trained embedding vector for word w_i, which further can be trained, and $\tilde{\boldsymbol{v}}_i^w$ is the fixed version. We also use convolutional neural networks (CNNs) to encode characters inside a word w_i into character-level representation $\boldsymbol{v}_i^c$. In addition, given the input sentence S, we exploit the POS tag of the tokens, and use the embedding $\boldsymbol{v}_i^{pos}$ via a lookup table.

Dependency features enhancement. Besides, we employ a TreeLSTM (Tai, Socher, and Manning 2015; Miwa and Bansal 2016) to encode the dependency structural feature representation $\boldsymbol{v}_i^{syn}$ for each corresponding word w_i. We concatenate these representations into unified input representation: $\boldsymbol{x}_i = [\boldsymbol{v}_i^w; \tilde{\boldsymbol{v}}_i^w; \boldsymbol{v}_i^c; \boldsymbol{v}_i^{pos}; \boldsymbol{v}_i^{syn}]$ Then, a BiLSTM is used as encoder for learning the context representations. We then concatenate two hidden states at each time step i as the sequence representation: $\boldsymbol{h}_i = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n\}$, which will be used for state representation learning.

State representation. The actions are predicted based on the neural transition state representations, as depicted in Figure 2(b). We first use a BiLSTM to generate representations $\boldsymbol{r}^\beta$ for each word w_i in β. Another BiLSTM is used for generating predicate representation $\boldsymbol{r}^{\lambda_p}$ at λ. Taking the 9-th transition step as example, the current representations of predicate 'stay' in λ and the word 'witnesses' in β are from BiLSTMs, respectively.

We use Stack-LSTM (Zhang et al. 2019; Yuan, Jiang, and Tu 2019) to encode the elements in stacks (e.g., σ_*, α_*). For instance, the representations of the top elements in stacks (i.e., 'When' in σ_l, 'witnesses' in α_l, 'nothing' in σ_r, ',' in α_r) is obtained from the Stack-LSTM, respectively. Besides of the regular states introduced in s, we also maintain the trace of action histories δ via a stack. We summarize all these state representations via concatenation as the initiation for the next action prediction:

$$\boldsymbol{g}_t = [\boldsymbol{r}_t^{\sigma_l}; \boldsymbol{r}_t^{\sigma_r}; \boldsymbol{r}_t^{\alpha_l}; \boldsymbol{r}_t^{\alpha_r}; \boldsymbol{r}_t^{\lambda_p}; \boldsymbol{r}_t^{\beta}; \boldsymbol{r}_t^{\delta}] \,. \tag{1}$$

Action prediction. Based on state $\boldsymbol{g}_t$, the model predicts the action probability, as in Figure 2(b):

$$\boldsymbol{P}^a = \text{softmax}(\text{FFNs}(\boldsymbol{g}_t))\,, \tag{2}$$

where FFNs($\cdot$) is a feed-forword network. At the meantime, once an arc is determined, a labeler[1] will assign a role label for the argument:

$$\boldsymbol{P}^r = \text{softmax}(\text{FFNs}(\boldsymbol{g}_t))\,. \tag{3}$$

Training. During training, we set the goal to maximize the likelihood of actions from the current states under the gold-standard actions. We first convert gold output structures of training data into action sequences. We minimize the following loss:

$$\mathcal{L} = -\textstyle\sum_t \hat{\varphi}_t \log p(\varphi_t|\Theta) + \frac{\zeta}{2}||\Theta||^2\,, \tag{4}$$

where $\hat{\varphi}$ denotes gold-standard actions, Θ is the set of all model parameters, and ζ is a co-efficient. Instead of greedily choosing the output with maximum probability, we apply beam-search algorithm (Zhang and Clark 2008; Lyu, Zhang, and Ji 2016) for decoding, where top-B partial outputs are cached for each transition decision, avoiding achieving sub-optimal results.

[1]In traditional transition methods, the arcs are determined jointly with their role labels by joint actions, e.g., *LEFT-ARC-A1*, whereas we find in our preliminary experiments that separating these two predictions can bring better performances.

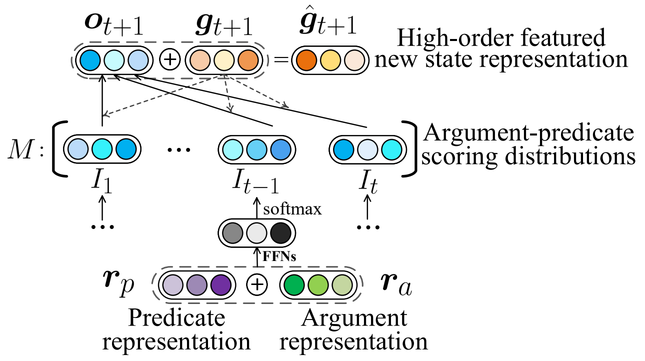


Figure 3: New state representations with high-order feature for argument recognition and role labeling.

The High-Order Model

In the vanilla transition model, the actions are incrementally predicted with the first-order local features, considering the dependencies to the predicates only. Here we present a high-order model further, leveraging the previously recognized argument-predicate information into the current decision. Figure 3 shows the overall network structure of our high-order features. We exploit the argument-predicate scoring distributions as the major sources for high-order compositions, which is mostly motivated by Lyu, Cohen, and Titov (2019). Two types of argument-predicate scoring distributions (i.e., $\boldsymbol{I}^a$ and $\boldsymbol{I}^r$) are exploited for action (a) and role (r) label predictions, respectively, and their network compositions are exactly the same. In the following, we describe the high-order networks in detail.

Concretely, the high-order features are calculated as follows. First, we derive a sequence of distribution-based features (i.e., $\boldsymbol{M}^\tau = \{\boldsymbol{I}_1^\tau, \cdots, \boldsymbol{I}_{t-1}^\tau, \boldsymbol{I}_t^\tau\}$) from the historical argument-predicates. For each $i \in [1, t]$, we have:

$$\boldsymbol{I}_t^\tau = \text{softmax}(\text{FFN}([\boldsymbol{r}_a; \boldsymbol{r}_p]))\,, \tag{5}$$

where $\boldsymbol{r}_a, \boldsymbol{r}_p$ are the current argument and predicate representation from σ_* and λ_p, respectively, $\tau \in \{a, r\}$ indicating that the goal of the feature is for either action predication or role labeling. Note that the output dimension is the same as their final classification goal for $\boldsymbol{P}^\tau$ (see Eq 2 and 3).

Second, we use an attention mechanism guided by the first-order representation $\boldsymbol{g}_{t+1}$ to obtain the high-order feature vector:

$$\begin{aligned} u_k^\tau &= \tanh(\boldsymbol{W}_1^\tau \boldsymbol{g}_{t+1} + \boldsymbol{W}_2^\tau \boldsymbol{I}_k^\tau) \\ \alpha_k^\tau &= \text{softmax}(u_k^\tau) \\ \boldsymbol{o}_{t+1}^\tau &= \textstyle\sum_{k=1}^{j} \alpha_k^\tau \boldsymbol{I}_k^\tau\,, \end{aligned} \tag{6}$$

where $\boldsymbol{o}_{t+1}^\tau$ is the desired representation, and $\boldsymbol{W}_1^\tau$ and $\boldsymbol{W}_2^\tau$ are parameters. Finally, we concatenate the first-order

	Without dependency feature				With dependency feature				
	Arg.			Prd.	Arg.			Prd.	Decode
	P	R	F1	F1	P	R	F1	F1	
He et al. (2018)	89.0	87.6	88.9	91.3	89.5	88.0	89.2	92.6	604
Cai et al. (2018)	89.5	88.4	89.2	94.6	89.7	89.0	89.5	94.9	450
Li et al. (2019)	**89.9**	89.2	89.6	95.0	90.2	89.5	89.9	95.2	432
Ours (Vanilla model)	88.0	88.9	88.4	92.3	88.5	89.6	88.9	94.3	**1,058**
Ours (High-Order model)	89.4	**91.1**	**89.8**	**95.2**	**90.4**	**90.0**	**90.2**	**95.5**	897

Table 1: Results on CoNLL09 English in-domain test set. Decode means decoding speed (token per second).

	Arg.	Prd.
Ours (High-Order model)	**91.0**	**96.3**
• Input features		
w/o Char	90.7	96.0
w/o POS	90.4	95.6
w/o Dep-tree	89.8	95.1
• Incremental high-order feature		
w/o $\boldsymbol{o}^a$	89.4	95.4
w/o $\boldsymbol{o}^r$	88.8	95.7
Ours (Vanilla model)	88.6	94.1
• Transition parsing direction		
left-to-right	88.5	93.0
right -to-left	88.0	92.8

Table 2: Ablation results (F1 score).

and high-order feature representation together (i.e., $\hat{\boldsymbol{g}}_{t+1} = [\boldsymbol{g}_{t+1}; \boldsymbol{o}^{\tau}_{t+1}]$) for transition action and role label predictions.

In the high-order model, the action prediction, decoding and learning processing are kept consistent with the vanilla neural model in § . But with the above high-order feature, the recognition for the following and farther arguments will be informed by the earlier decisions, avoiding the error accumulation in the vanilla transition system. It also worth noticing that with beam-search strategy, we are able to facilitate the use of high-order feature, making the information communication between arguments most efficient.

Experiments

Settings

We employ two dependency-based SRL benchmarks: CoNLL09 English, and Universal Proposition Bank (UPB) for other total seven languages. We use the pre-trained fasttext[2] word embeddings for each language as default word representation. The hidden sizes in BiLSTM, TreeLSTM and Stack-LSTM are 200. We use the two layer version of BiLSTM, Stack-LSTM. The dimension of transition states is 150 universally. We adopt the Adam optimizer with initial learning rate of 1e-5. ζ is 0.2, and beam size **B** is 32, according to our developing experiments. We train the model by mini-batch size in [16,32] with early-stop strategy. We evaluate the contextualized word representations, i.e., ELMo[3], BERT (base-cased-version)[4] and XLNet (base-version)[5]. Each dataset comes with its own train, develop and test sets. We use the precision (P), recall (R) and F1 score as the metric. We conduct significance tests via Dan Bikel's randomized evaluation comparator. Our codes are open at https://github.com/scofield7419/TransitionSRL.

Ablation Study

We study the contributions from each part of our methods based on CoNLL09 in-domain developing set, in Table 2.

Input features. When removing the character encoder, POS tags and syntactic dependencies, respectively, the performances drop consistently. We notice that the dependency structural knowledge is of the greatest importance for both the argument and predicate recognition, among other features, which coincides with the prior findings of the structural syntax information (Marcheggiani, Frolov, and Titov 2017; Zhang, Wang, and Si 2019; Fei et al. 2020).

High-order feature. We ablate the argument-predicate scoring distribution $\boldsymbol{o}^a$ and $\boldsymbol{o}^r$ to see the contribution of the proposed interaction mechanism. We can find that both these two distributions plays key role for arguments role labeling, especially the $\boldsymbol{o}^r$. When removing such incremental high-order feature away (i.e., vanilla model w/o $\boldsymbol{o}^a$&$\boldsymbol{o}^r$), the results drop significantly. This verifies the usefulness of the proposed high-order feature.

Transition parsing order. Our model performs argument parsing around the predicate down the 'from-near-to-far' direction. Compared with the traditional 'left-to-right' reading order, or the reversed 'right -to-left' order, we can find that the *close-first* parsing scheme is much more effective.

Main Results

CoNLL09 in-domain results. We mainly make comparisons with the recent previous end-to-end SRL models. In

[2]https://fasttext.cc/

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

[4]https://github.com/google-research/bert

[5]https://github.com/zihangdai/xlnet

	DE	FR	IT	ES	PT	FI	ZH	Avg.
He et al. (2018)	68.5	84.3	67.6	73.0	79.2	70.4	64.9	72.6
Cai et al. (2018)	69.9	85.5	67.8	**74.5**	78.9	71.0	65.8	73.3
Li et al. (2019)	69.7	85.0	69.0	73.8	79.6	70.5	**66.0**	73.4
Ours (Vanilla model)	68.9	86.2	68.6	73.0	80.1	69.9	63.6	72.9
Ours (High-Order model)	**70.5**	**87.8**	**69.2**	73.6	**80.7**	**71.6**	65.6	**74.2**

Table 3: End-to-end SRL on UPB data. Values are F1 scores for argument recognition and role labeling.

	Arg.			Prd.
	P	R	F1	F1
He et al. (2018)	79.5	76.2	78.6	78.6
Cai et al. (2018)	**80.4**	76.9	79.5	80.5
Li et al. (2019)	79.9	78.5	79.2	82.0
Ours (High-Order model)	80.2	**79.8**	**80.0**	**82.7**

Table 4: Results on out-of-domain data of CoNLL09.

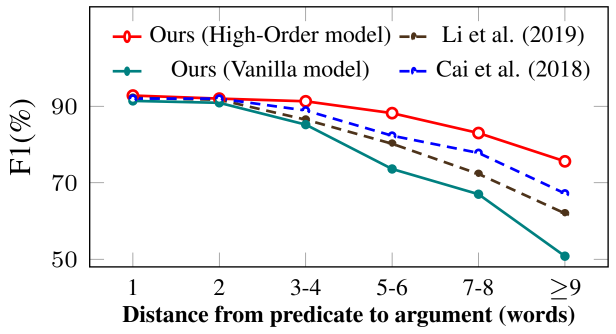


Figure 4: Argument recognition under varying surface distance between predicates and arguments.

	P	R	F1
Zhao et al. (2009)	-	-	85.4
Björkelund et al. (2010)	87.1	84.5	85.8
FitzGerald et al. (2015)	-	-	86.7
Roth and Lapata (2016)	88.1	85.3	86.7
Marcheggiani and Titov (2017)	89.1	86.8	88.0
He et al. (2018)	89.8	89.6	89.7
Cai et al. (2018)	89.9	90.2	90.0
Li et al. (2019)	**90.9**	90.2	90.5
Ours (High-Order model)	90.3	**91.0**	**90.7**

Table 5: Pipeline argument role labeling on CoNLL09.

	ELMo	BERT	XLNet
He et al. (2018)	89.7	90.8	90.3
Cai et al. (2018)	90.8	91.7	91.4
Li et al. (2019)	**91.5**	92.0	91.8
Ours (High-Order model)	91.3	**92.2**	**92.0**

Table 6: Results with contextualized word representation.

Table 1, first of all, we find that the syntactic dependency feature configurations universally contribute to both the argument recognition and predicate disambiguation, compared with the syntax-agnostic models. Most importantly, our high-order model outperforms these baselines on both two subtasks. With dependency features, the improvements are more significant, i.e., 90.2% (Arg.) and 95.5% (Prd.). Also our model is especially better at predicate disambiguation. Besides, the transition systems beat competitors with higher decoding speed, nearly two times faster.

Results for other languages. Table 3 shows the performances on UPB for other languages (with dependency features). Overall, our high-order model wins the best averaged F1 score (74.2%). Also the high-order feature universally helps to improve the vanilla transition model with average 1.3%(=74.2-72.9) F1 score. The improvements so far by our model demonstrate its effectiveness.

CoNLL09 out-of-domain results. Table 4 compares the performances on out-of-the-domain test set of CoNLL09 data with dependency features. Overall, the similar trends are kept as that on in-domain dataset. Our model still performs the best, yielding 80.0% (Arg.) and 82.7% (Prd.) F1 scores, verifying its generalization capability on capturing the semantic preferences of the task.

Argument role labeling. We next compare the results in Table 5 for standalone argument role labeling, where the models are given by the gold pre-identified predicates as input, and output the argument boundaries and roles. Compared with the baseline pipeline systems, the end-to-end systems can perform better, while our transition model gives a new state-of-the-art 90.7% F1 score.

Contextualized word representation. Finally, we take advances of the recent contextualized word representations, ELMo, BERT and XLNet. As shown in Table 6, all the end-to-end SRL models can benefit a lot. With enhanced word representations, we see that the graph-based model by Li et al. (2019) can achieve better results (91.5%) with ELMo. while our model can obtain the best performances by BERT (92.2%), and XLNet (92.0%).

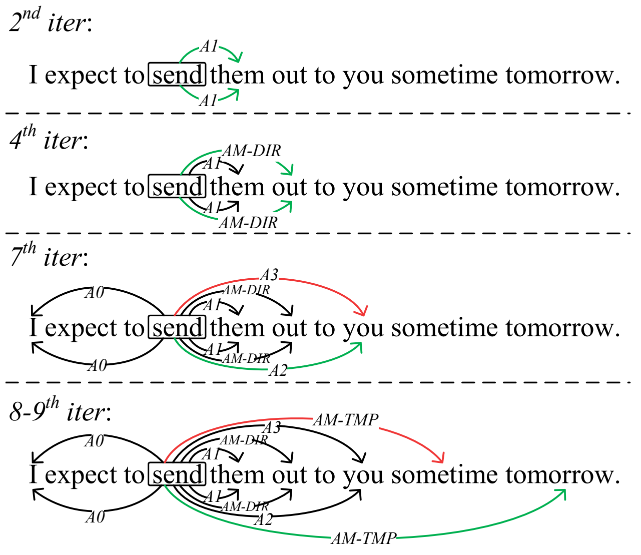


Figure 5: Visualizations of the parses under varying transition iterations. The predicate is in the box. The arrows above the sentence are from the vanilla model, while the ones beneath are by our high-order system. Red and green arrows represent incorrect and correct argument or role, respectively.

Discussion

Necessity of back refining. We introduce the *incremental high order interaction* for improving the argument recognition, by informing the detection of following and farther arguments with previous recognition history information, that is, the later decisions are influenced by former decisions. Such uni-directional propagation naturally spurs a potential question: *Is it necessary to use the later refined decisions to inform and refine back the former decisions?* Theorically speaking, the negative influences only exist at the initial transition decisions, because our proposed interaction mechanism combined with beam search strategy can largely refined the future decisions. One the other hand, the *close-first* parsing scheme can ensure that the nearer-to-predicate arguments recognized earlier are more likely to be correct. In addition, in our experiment we proves the unnecessity of the back refining. Figure 4 shows the performance under varying surface distance from predicate to arguments. It is clear that within the distances of 3-4, the performances by our full model changes very little. Even when removing the interaction mechanism (into vanilla model), the decreases under short surface distances is not much significant. Lastly, our high-order model handles better the long-distance dependency issues, compared with baselines.

High-order features. We now look into the incremental high order interaction mechanism. We consider conducting empirical visualization of the transition parsing steps. We compare the transition results by high-order model, and the vanilla model without high-order feature, as illustrated in Figure 5. At the earlier steps with short distance between predicate and argument, e.g., second and fourth iteration steps, both two transition models can make correct prediction. At 7th step, the vanilla model wrongly assigns a *A3* for argument ‘you’, and further falsely determines the ‘sometime’ as *AM-TMP* argument at 8th step. On the contrary, our high-order model with such interaction can perform correct role labeling, and yield accurate argument detection even in remote distance.

	U	C	R
Gold	55	0	88
Ours (High-Order model)	82	8	98
Ours (Vanilla model)	106	10	122
left-to-right Trans.	212	13	139
He et al. (2018)	242	14	208
Cai et al. (2018)	210	11	162

Table 7: Results for argument role violation.

Role violation. To further explore the strengths of our transition model, we study the argument role violation (Punyakanok et al. 2004; FitzGerald et al. 2015; He et al. 2018). Consider categorizing the role labels into three types: (U) unique core roles (A0-A5, AA), (C) continuation roles and (R) reference roles. 1) If a core role appears more than once, U is violated. 2) If C-X role not precedes by the X role (for some X), C is violated; 3) R is violated if R-X role does not appear. Table 7 shows the role violation results.

First of all, our high-order model gives the most consistent results with gold ones, with minimum role violations. This partially indicates that the system can learn the latent constraints. But without the *incremental high order interaction*, performances by vanilla model for continuation and reference roles get hurt. This is reasonable, since generally the non-core roles are more remote from their predicate. We also find that our transition model with *close-first* parsing order performs better, especially for the unique core role, compared with the ‘left-to-right’ parsing order. Finally, the graph-based baselines tend to predict more duplicate core arguments, and also recognize fewer reference argument roles.

Conclusion

We investigated a neural transition model for end-to-end semantic role labeling. We designed the transition system where the predicates were first discovered one by one, and then the associated arguments were determined in a *from-near-to-far* order. We proposed to use the incremental high-order feature, leveraging the previously recognized argument-predicate scoring distribution into the current decision. Experimental results on the dependency-based SRL benchmarks, including CoNLL09 and Universal Proposition Bank datasets, showed that the transition model brings state-of-the-art performances, meanwhile keeping higher decoding efficiency. Further analysis demonstrated the usefulness of the high-order feature and the close-first parsing order.

Acknowledgments

We thank the anonymous reviewers for providing constructive comments and suggestions. This work is supported by the National Natural Science Foundation of China (No. 61772378 and 61602160), the National Key Research and Development Program of China (No. 2017YFC1200500), the Research Foundation of Ministry of Education of China (No. 18JZD015).

References

Akbik, A.; Chiticariu, L.; Danilevsky, M.; Li, Y.; Vaithyanathan, S.; and Zhu, H. 2015. Generating high quality proposition banks for multilingual semantic role labeling. In *Proceedings of the ACL*, 397–407.

Akbik, A.; and Li, Y. 2016. Polyglot: Multilingual semantic role labeling with unified labels. In *Proceedings of ACL*, 1–6.

Bastianelli, E.; Castellucci, G.; Croce, D.; and Basili, R. 2013. Textual Inference and Meaning Representation in Human Robot Interaction. In *Proceedings of the Joint Symposium on Semantic Processing.*, 65–69.

Berant, J.; Chou, A.; Frostig, R.; and Liang, P. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the EMNLP*, 1533–1544.

Björkelund, A.; Bohnet, B.; Hafdell, L.; and Nugues, P. 2010. A High-Performance Syntactic and Semantic Dependency Parser. In *Proceedings of the Coling*, 33–36.

Cai, D.; and Lam, W. 2019. Core Semantic First: A Top-down Approach for AMR Parsing. In *Proceedings of the EMNLP*, 3799–3809.

Cai, J.; He, S.; Li, Z.; and Zhao, H. 2018. A Full End-to-End Semantic Role Labeler, Syntactic-agnostic Over Syntactic-aware? In *Proceedings of the CoLing*, 2753–2765.

Choi, J. D.; and Palmer, M. 2011. Transition-based Semantic Role Labeling Using Predicate Argument Clustering. In *Proceedings of the ACL*, 37–45.

Christensen, J.; Mausam; Soderland, S.; and Etzioni, O. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the International Conference on Knowledge Capture*, 113–120.

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 NAACL*, 4171–4186.

Dyer, C.; Ballesteros, M.; Ling, W.; Matthews, A.; and Smith, N. A. 2015. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. In *Proceedings of the ACL*, 334–343.

Fei, H.; Ren, Y.; and Ji, D. 2020. High-order Refining for End-to-end Chinese Semantic Role Labeling. In *Proceedings of the AACL*, 100–105.

Fei, H.; Zhang, M.; and Ji, D. 2020. Cross-Lingual Semantic Role Labeling with High-Quality Translated Training Corpus. In *Proceedings of the ACL*, 7014–7026.

Fei, H.; Zhang, M.; Li, F.; and Ji, D. 2020. Cross-lingual Semantic Role Labeling with Model Transfer. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28: 2427–2437.

FitzGerald, N.; Täckström, O.; Ganchev, K.; and Das, D. 2015. Semantic Role Labeling with Neural Network Factors. In *Proceedings of the EMNLP*, 960–970.

Gildea, D.; and Jurafsky, D. 2000. Automatic Labeling of Semantic Roles. In *Proceedings of the ACL*, 512–520.

Goldberg, Y.; and Elhadad, M. 2010. An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Proceedings of the NAACL*, 742–750.

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 CoNLL*, 1–18.

He, L.; Lee, K.; Levy, O.; and Zettlemoyer, L. 2018. Jointly Predicting Predicates and Arguments in Neural Semantic Role Labeling. In *Proceedings of the ACL*, 364–369.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.

Kurita, S.; and Søgaard, A. 2019. Multi-Task Semantic Dependency Parsing with Policy Gradient for Learning Easy-First Strategies. In *Proceedings of the ACL*, 2420–2430.

Lei, T.; Zhang, Y.; Màrquez, L.; Moschitti, A.; and Barzilay, R. 2015. High-Order Low-Rank Tensors for Semantic Role Labeling. In *Proceedings of the NAACL*, 1150–1160.

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*, 6730–6737.

Lyu, C.; Cohen, S. B.; and Titov, I. 2019. Semantic Role Labeling with Iterative Structure Refinement. In *Proceedings of EMNLP*, 1071–1082.

Lyu, C.; Zhang, Y.; and Ji, D. 2016. Joint Word Segmentation, POS-Tagging and Syntactic Chunking. In *Proceedings of the AAAI*, 3007–3014.

Marcheggiani, D.; Frolov, A.; and Titov, I. 2017. A Simple and Accurate Syntax-Agnostic Neural Model for Dependency-based Semantic Role Labeling. In *Proceedings of the CoNLL*, 411–420.

Marcheggiani, D.; and Titov, I. 2017. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In *Proceedings of the EMNLP*, 1506–1515.

Miwa, M.; and Bansal, M. 2016. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In *Proceedings of the ACL*, 1105–1116.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *Proceedings of the NAACL*, 2227–2237.

Pradhan, S.; Ward, W.; Hacioglu, K.; Martin, J.; and Jurafsky, D. 2005. Semantic Role Labeling Using Different Syntactic Views. In *Proceedings of the ACL*, 581–588.

Punyakanok, V.; Roth, D.; and Yih, W. 2008. The Importance of Syntactic Parsing and Inference in Semantic Role Labeling. *Computational Linguistics* 34(2): 257–287.

Punyakanok, V.; Roth, D.; Yih, W.-t.; Zimak, D.; and Tu, Y. 2004. Semantic Role Labeling Via Generalized Inference Over Classifiers. In *Proceedings of the CoNLL*, 130–133.

Roth, M.; and Lapata, M. 2016. Neural Semantic Role Labeling with Dependency Path Embeddings. In *Proceedings of the ACL*, 1192–1202.

Scheible, C. 2010. An Evaluation of Predicate Argument Clustering using Pseudo-Disambiguation. In *Proceedings of the LREC*.

Shen, D.; and Lapata, M. 2007. Using Semantic Roles to Improve Question Answering. In *Proceedings of the EMNLP*, 12–21.

Shi, C.; Liu, S.; Ren, S.; Feng, S.; Li, M.; Zhou, M.; Sun, X.; and Wang, H. 2016. Knowledge-Based Semantic Embedding for Machine Translation. In *Proceedings of the ACL*, 2245–2254.

Strubell, E.; Verga, P.; Andor, D.; Weiss, D.; and McCallum, A. 2018. Linguistically-Informed Self-Attention for Semantic Role Labeling. In *Proceedings of the EMNLP*, 5027–5038.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the ACL*, 1556–1566.

Wang, B.; Lu, W.; Wang, Y.; and Jin, H. 2018. A Neural Transition-based Model for Nested Mention Recognition. In *Proceedings of the EMNLP*, 1011–1017.

Xia, Q.; Li, Z.; Zhang, M.; Zhang, M.; Fu, G.; Wang, R.; and Si, L. 2019. Syntax-Aware Neural Semantic Role Labeling. In *Proceedings of the AAAI*, 7305–7313.

Xiong, D.; Zhang, M.; and Li, H. 2012. Modeling the Translation of Predicate-Argument Structure for SMT. In *Proceedings of the ACL*, 902–911.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J. G.; Salakhutdinov, R.; and Le, Q. V. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Proceedings of the NeurIPS*, 5754–5764.

Yuan, Y.; Jiang, Y.; and Tu, K. 2019. Bidirectional Transition-Based Dependency Parsing. In *Proceedings of the AAAI*, 7434–7441.

Zhang, J.; Qin, Y.; Zhang, Y.; Liu, M.; and Ji, D. 2019. Extracting Entities and Events as a Single Task Using a Transition-Based Neural Model. In *Proceedings of the IJCAI*, 5422–5428.

Zhang, M.; Zhang, Y.; and Fu, G. 2016. Transition-Based Neural Word Segmentation. In *Proceedings of the ACL*, 421–431.

Zhang, M.; Zhang, Y.; and Fu, G. 2018. Transition-Based Neural Word Segmentation Using Word-Level Features. *Journal of Artificial Intelligence Research* 63: 923–953.

Zhang, Y.; and Clark, S. 2008. A Tale of Two Parsers: Investigating and Combining Graph-based and Transition-based Dependency Parsing. In *Proceedings of the EMNLP*, 562–571.

Zhang, Y.; and Clark, S. 2010. A Fast Decoder for Joint Word Segmentation and POS-Tagging Using a Single Discriminative Model. In *Proceedings of the EMNLP*, 843–852.

Zhang, Y.; Wang, R.; and Si, L. 2019. Syntax-Enhanced Self-Attention-Based Semantic Role Labeling. In *Proceedings of EMNLP*, 616–626.

Zhao, H.; Chen, W.; Kazama, J.; Uchimoto, K.; and Torisawa, K. 2009. Multilingual Dependency Learning: Exploiting Rich Features for Tagging Syntactic and Semantic Dependencies. In *Proceedings of the CoNLL*, 61–66.

Zhou, H.; Zhang, Y.; Huang, S.; and Chen, J. 2015. A Neural Probabilistic Structured-Prediction Model for Transition-Based Dependency Parsing. In *Proceedings of the ACL*, 1213–1222.