Dependency-Gated Cascade Biaffine Network for Chinese Semantic Dependency Graph Parsing

Abstract. The Chinese Semantic Dependency Graph (CSDG) parsing breaks the limitation of the syntactic or semantic tree structure dependency system with a richer representation ability to express more complex language phenomena and semantic relationships. Most of the existing CSDG parsing systems used transition-based approach. It needs to define a complex transition system and its performance depends heavily on whether the model can properly represent the transition state. In this paper, we adopt neural graph-based approach which using Biaffine network to solve the CSDG parsing task. Furthermore, considering that dependency edge and label have the strong relationship, we design an effective dependency-gated cascade mechanism to improve the accuracy of dependency label prediction. Our work is the first to cascade the dependency edge and the dependency label predictions in an end-to-end model. We test our system on the SemEval-2016 Task 9 dataset. Experiment result shows that our model achieves state-of-the-art performance with 6.05% labeled F1-score improvement compared to the previous best model.

Keywords: Chinese semantic dependency graph paring, Dependency-gated cascade mechanism, Biaffine network.

1 Introduction

Chinese is a flexible language, especially when it comes to word order. In order to obtain the semantic information of sentences, semantic parsing has been widely studied in the past decade. However, previous semantic parsing work [1,2] focused on Semantic Role Labeling, which is a shallow semantic parsing task and does not have a deep understanding of sentence semantics.

Compared to the Semantic Role Labeling, Semantic Dependency Parsing [3,4] is a deeper semantic analysis. Semantic Dependence Parsing aims to directly capture deep semantic information across the constraints of the syntactic structure of sentences.

A lot of work in previous years focused on the tree structure dependency parsing [5]. However, in natural language, a word can be the argument of multiple predicates (*non-local*), and the dependency arcs may cross each other (*non-projection*), which results in a directed acyclic graph (DAG) structure. Traditional dependency tree structures cannot express these linguistic phenomena.

In order to express these linguistic phenomena under the dependency system, [6] proposed Chinese Semantic Dependence Graph (CSDG) Parsing, developed Chinese semantic parsing into Semantic Dependency Graph Parsing. The dependency graph system breaks the limitation of the tree structure with a richer representation ability.

Due to the complexity of semantic dependency graph structures, it is still a challenge to design effective algorithms for graph structures. [7] used a two-stage approach based

on transition-based approach by first producing a semantic dependency tree and then predicting the dependency arcs of the multi-head word. [8] adopted Stack-LSTM architecture to represent the transition state, Tree-LSTM to represent sub-graph structures and Bi-LSTM Subtraction to represent sentence fragments. Nevertheless, a transition-based approach requires a complex transition system for the graph structures and its parsing accuracy is lower than graph-based parsers [10,11,15] because the greedy decoding strategy cannot obtain the global optimal solution.

In this paper, we port the Biaffine network [10,11,12] to the Chinese Semantic Dependence Graph (CSDG) Parsing task [6] to handle complex graph structure dependency parsing. Furthermore, in order to capture the dependency edge information of words and further improve the classification accuracy of dependent labels, we propose the dependency-gated cascade mechanism, which contains an additional dependency-gate that allows dependent edge vector of words predicted by the first edge Biaffine classifier to be passed to the following label Biaffine classifier. To the best of our knowledge, our work is the first to cascade the dependency edge and the dependency label predictions in an end-to-end model, and is the first to use word-dependent path information to help predict the dependency label. The ablation experiments show that the dependency-gated cascade mechanism can further improve the performance of the model.

We use SemEval-2016 Task 9: Chinese Semantic Dependency Graph Parsing [6] as our testbed. Experiment result shows that our model achieves state-of-the-art performance, beating the previous more complex state-of-the-art system by 6.05% labeled F1-score.

2 Background

2.1 Chinese Semantic Dependency Graph Parser

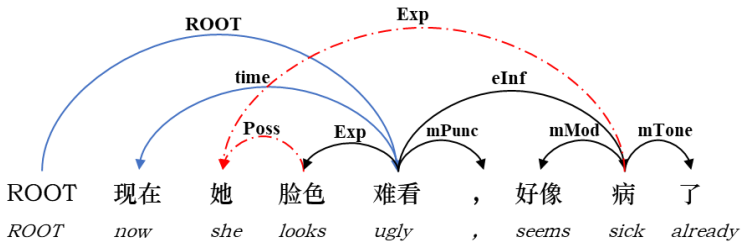


Fig. 1. An example of CSDG in SemEval-2016 Task 9

Semantic Dependency Parsing extracts the semantic relationship between all modifiers and head words in a sentence [6]. The semantic relationships between words in a sentence can be formalized in a directed graph. Its form is simple, easy to understand and use.

Unlike the restricted representation of tree structure, graph structure dependency parser can fully and naturally represent the linguistic phenomenon of natural language. In

the dependency graph parser, as long as there is a semantic relationship between the two words in the sentence, there is a dependency arc between them [6]. This means that in the parser results allow a word to have multiple parent nodes (*non-local*), and cross-over (*non-projection*) may occur between the dependent arcs.

Figure 1 shows an example of a Chinese semantic dependency graph presented in SemEval-2016 Task 9. The dashed red arcs indicate non-local dependencies and the solid blue arcs indicate non-projection phenomenon. Here, "她(she)" is the argument of "脸色(looks)" and it is also an argument of "病(sick)". Dependent arc "病(sick)"→"她(she)" and dependent arc "难看(ugly)"→ "脸色(looks)" cross each other.

Formally, given a set of $L = \{l_1, \ldots, l_{|L|}\}$ of semantic dependency labels, for a sentence $x = (w_0, \ldots, W_n)$, its sematic dependency graph is a labeled *directed acyclic graph* (DAG): $G = (V, A)$, where

- $V = \{0,1, \ldots, n\}$ is a set of nodes that represent each word in the sentence (including *ROOT*);
- $A \subseteq V \times V \times L$ is a set of labeled arcs.

2.2 Related Work

Two typical approaches to dependency parsing are the transition-based approach [8,9] and the graph-based approach [10,11,12,9]. The transition-based approach constructs dependency structures by predicting predefined transition actions such as shift, reduce and so on. The graph-based approach predicts the dependency structures with the highest score through the learned scoring function and decoding algorithm of the corresponding structures.

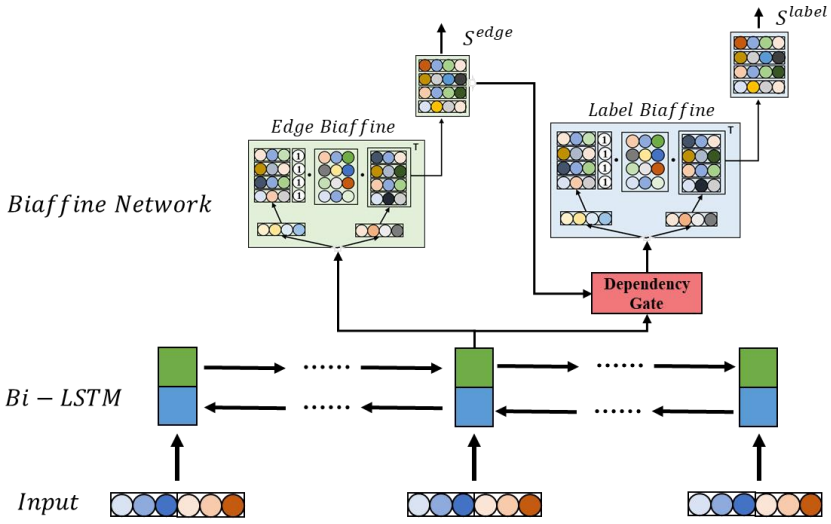


Fig. 2. Biaffine Model with Dependency-Gated Cascade Mechanism

In recent years, the neural network has been successfully applied to the task of dependency parsing. [13] have made a breakthrough in the syntactic dependency parsing

with the simple feedforward neural network in a transition-based framework. The researchers then extended the neural network to improve the performance of dependency parsing. [14] proposed a structured perceptron training method for neural transition-based dependency parsing and adopted the beam search technology in the decoding stage. [15] used bidirectional LSTM feature extractor and multi-layer perceptron scorer to experiment on a transition-based approach and a graph-based approach, respectively.

The graph-based approach, which uses the biaffine mechanism, achieves the start of art performance on the tree-structured syntactic dependency parsing, and it can be extended to the graph-structured dependency parsing [11]. [21] applied the objective function of max-margin and the AD^3 decoding algorithm to the neural graph-based semantic dependency graph parsing.

3 Architecture

We propose a novel dependency-gated cascade Biaffine network which contains an additional dependency-gate that allows dependent edge information to be passed to the label Biaffine classifier. The model architecture is illustrated in Figure 2, which consists two parts, a three-layer highway Bi-LSTM as encoding layer and a cascade Biaffine network.

3.1 Encoding Layer

In this paper, we use deep Bi-directional LSTM network to capture multi-granularity semantic information of the input sequence. To avoid overfitting, we use a Dropout LSTM cell. At the same time, in order to ensure that the model can be trained normally, we use the Highway connection to modify the information flow between the LSTM layers.

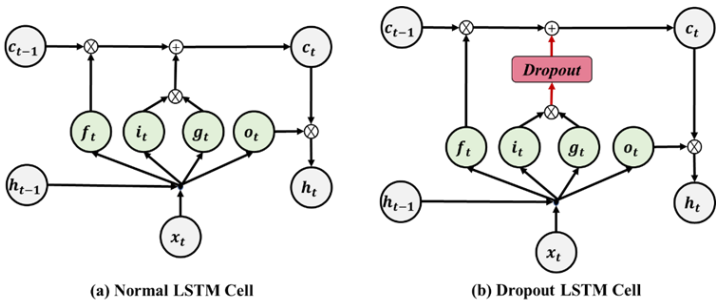


Fig. 3. Difference between (a) normal LSTM cell and (b) Dropout LSTM cell

Inputs. Each Chinese word x_i is represented as the concatenation of word embedding $e_i^{(word)}$, POS tag embedding $e_i^{(pos)}$ and chars' representation $h_i^{(char)}$.

$$x_i = e_i^{(word)} \oplus e_i^{(pos)} \oplus h_i^{(char)} \tag{1}$$

Here $h_i^{(char)}$ is the sum of the output states of a character-level LSTM network.

Dropout LSTM Cell. Overcoming over-fitting during training is an important issue for LSTM network, especially as the depth and size of the network increases.

Differently from the widely adopted dropout method [18], as shown in Figure 3, in this paper we apply dropout to the cell update vector $\tilde{C}_t$. Previous work [16] has proved that this method can avoid over-fitting well without causing losing long-term memory.

Highway LSTM Network. Highway network [17] include two non-linear gates: a carry gate, $C_{gate}(x)$, and a transform gate, $T_{gate}(x)$.

Unlike the highway LSTM used in previous work [19], we only do skip connect to the state h_t of the LSTM, which makes the calculation simpler and more efficient.

$$h_t^l = LSTM(h_t^{l-1}; W_H^l) \cdot T_{gate}(h_t^{l-1}) + h_t^{l-1} \cdot C_{gate}(h_t^{l-1}) \quad (2)$$

$$T_{gate}(h_t^{l-1}) = sigmoid(W_T \cdot h_t^{l-1} + b_T) \quad (3)$$

$$C_{gate}(h_t^{l-1}) = sigmoid(W_C \cdot h_t^{l-1} + b_C) \quad (4)$$

$W_H^l,\ W_T, b_T, W_C, b_C$ are matrices and bias terms.

3.2 Biaffine Scorer

We adopt biaffine attention mechanism proposed by [12] which can well model the relationship of dependent word and head word. For predicting dependency edge, we first feed the LSTM encoded representation H_{lstm} into single-layer feedforward networks (FNN) in order to distinguish dependent information from head information:

$$h_i^{(edge-head)} = FNN^{(edge-head)}\left(h_{lstm}^i\right) \quad (5)$$

$$h_i^{(edge-\mathrm{dep})} = FNN^{(edge-dep)}\left(h_{lstm}^i\right) \quad (6)$$

Then, we use the biaffine transformation (Eq 7) to obtain the scoring matrix of all possible edges in the sentence. Different from the tree-structured parsing, every word can have multiple heads in graph-structured parsing. Following recent work [10], we perform an additional sigmoid transformation on the scoring matrix to determine the existence of each edge.

$$Biaffine(x_1, x_2) = x_1^T U x_2 + W(x_1 \oplus x_2) + b \quad (7)$$

$$s_{i,j}^{(edge)} = Biaffine^{(edge)}\left(h_i^{edge-dep}, h_j^{edge-head}\right) \quad (8)$$

$$p_{i,j}^{*(edge)} = Sigmoid\left(s_{i,j}^{(edge)}\right) \quad (9)$$

The method of label prediction is similar to that of edge prediction. It's worth explaining that we normalize the vector space of all label categories with softmax normalization when calculating the label probability.

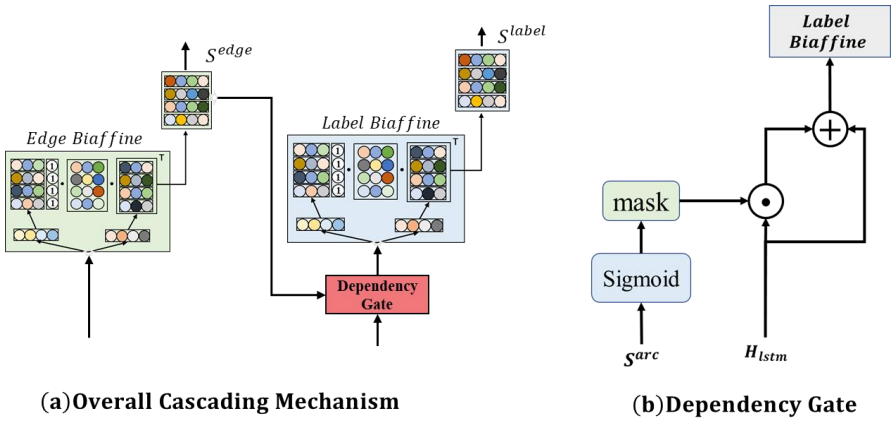
$$h_i^{(\text{label}-head)} = FNN^{(\text{label}-head)}(r_i) \tag{10}$$

$$h_i^{(\text{label}-\text{dep})} = FNN^{(\text{label}-dep)}(r_i) \tag{11}$$

$$s_{i,j}^{(\text{label})} = Biaffine^{(label)}\left(h_i^{label-dep}, h_j^{label-head}\right) \tag{12}$$

$$p_{i,j}^{*(label)} = Softmax\left(s_{i,j}^{(label)}\right) \tag{13}$$

3.3 Dependency-Gated Cascade Mechanism



(a)**Overall Cascading Mechanism** (b)**Dependency Gate**

Fig. 4. Illustration of the Dependency-Gated Cascade Mechanism

Compared with the normal Biaffine network structure, this paper introduces the dependency-gated cascade mechanism. As shown in Figure 2, the proposed mechanism introduces an additional gate that leverages dependent edge vector of words in order to improve dependency label prediction performance.

First, the dependency edge score is calculated by a softmax function to obtain the dependency edge probability.

$$P^{edge} = Sigmoid(S^{edge}) \tag{14}$$

In order to extract the true dependency edge information as much as possible, and to mask the influence of the impossible dependency edge, we reset the probability that the probability value is less than 0.5 to 0 (Eq 15).

$$P_{mask}^{edge} = \begin{cases} p_{i,j}^{edge}, & p_{i,j}^{edge} \geq 0.5 \\ 0, & p_{i,j}^{edge} < 0.5 \end{cases} \tag{15}$$

We consider this probability as a special dependent attention score, which represents the semantic dependency correlation between two words in a sentence. We treat P_{mask}^{edge} as a weight and perform a weighted arithmetic mean on the LSTM output to get H^{gated}.

Then connect H and H^{gated} to get the input H^{label} of the dependent label Biaffine classifier.

$$H^{gated} = H \cdot P^{edge} \tag{16}$$

$$H^{label} = H \oplus H^{gated} \tag{17}$$

At this point, the representation of each word in H^{label} contains the original LSTM output vector and its dependent path weighted arithmetic mean representation vector. In this way, the model can better understand the semantic relationship of the two words from the perspective of the entire dependency graph, thereby improving the accuracy of the dependent label classification. At the same time, the cascading mechanism causes the loss of the edge classifier to include a portion of the label prediction loss. This forces the edge classifier to also consider partial label information in the prediction.

3.4 Joint Optimization

We calculate the loss of the edge Biaffine classifier and the label Biaffine classifier separately.

$$Loss_{i,j}^{(edge)} = -p_{i,j}^{(edge)} \, log \, p_{i,j}^{*(edge)} - (1 - p_{i,j}^{(edge)}) \, log(\, 1 - p_{ij}^{*(edge)}) \tag{18}$$

$$Loss_{i,j}^{(\text{labe})} = -\sum_{label} log \, p_{i,j}^{*(label)} \tag{19}$$

To train the entire semantic dependency graph parser, we use joint losses with interpolation as shown in the Eq 20. Interpolation can control the importance of each part loss, which is very critical for the training of parser.

$$Loss^{(parser)} = \lambda Loss^{(label)} + (1 - \lambda) Loss^{(edge)} \tag{20}$$

where λ is the weight that controls the interaction of the loss terms

4 Experiments

4.1 Dataset and evaluation metrics

We use the SemEval-2016 Task 9: *Chinese Semantic Dependency Graph Parsing* dataset as our benchmark dataset. It contains two distinguished corpora in the domain of NEWS and TEXTBOOKS (from primary school textbooks). We follow the official evaluation setup. For convenience of comparison, we use the same evaluation metrics with [8]: **LF** (labeled F-score) and **UF** (unlabeled F-score). LF is the primary evaluation metric when comparing the performance. At the same time, to measure the performance of the model on non-local dependency arcs, we use two special metrics: **NLF** (non-local labeled F-score) and **NUF** (non-local unlabeled F-score). Statistics about the dataset are shown in Table 1.

Table 1. Statistics of the dataset. #n-rate is the rate of non-local dependency arc

Domain	Dataset	#sent	#word	#arc	#g-rate	#n-rate
NEWS	Train	8301	250249	257252	43.55%	4.82%
	Dev	534	15325	15695	41.76%	4.27%
	Test	1233	34305	34872	29.52%	2.95%
TEXT	Train	10754	128095	131975	23.30%	5.06%
	Dev	1535	18257	18815	23.65%	5.10%
	Test	3073	36097	37221	23.01%	5.04%

4.2 Setup

We almost use the same hyperparameter settings as [11], with a few exceptions. The word embedding we use is trained on the Chinese Gigawords[1] using word2vec model [20]. The word embedding size and POS embedding size are both 100. We use a 3-layer Highway Bi-LSTM with 600-dimensional hidden size as encoding layer. The output of the LSTM network is passed to the Biaffine classifier with 600-dimensional hidden size. For the character model, we use 100-dimensional character embeddings with 400-dimensional recurrent states.

The batch size we use is 3000. Every step, we dropout word embedding with 20% probability and dropout the inputs of Char-LSTM, Highway-LSTM, and Biaffine classifiers with 33% probability. The dropout probability of Highway-LSTM's recurrent connections was 20%. We set the joint optimization parameter λ to 0.5. We use Adam optimizer with L_2 regularization, setting learning rate, β_1, β_2 to e^{-3}, 0, 0.95 respectively.

Table 2. Results on NEWS domain and TEXTBOOKS domain test dataset.

System	NEWS				TEXTBOOKS			
	LF	**UF**	**NLF**	**NUF**	**LF**	**UF**	**NLF**	**NUF**
Ding et al.2014	62.29	80.56	39.93	64.29	71.94	85.24	50.67	69.97
Wang et al.2018	63.30	81.14	51.16	**66.92**	72.92	85.71	61.91	72.74
Our Model	**69.66**	**84.25**	**53.34**	66.01	**80.40**	**90.05**	**65.97**	**74.35**

4.3 Experiment Results and Analysis

Table 2 shows the results of our model on the NEWS and TEXTBOOKS test sets respectively. To prove the effectiveness of our proposed model, we compared it with two strong baseline models proposed by [7] and [8]. On the TEXTBOOKS test, our model was significantly superior to the two baselines on all four metrics. The LF score, as the most critical metric of system performance, improved by 7.28% compared to Wang's

[1] https://catalog.ldc.upenn.edu/LDC2003T09

work and 8.46% compared to Ding's work. On the NEWS test set, our model achieved the optimal results in three main metrics: LF, UF and NLF. The LF score also got a 6.05% boost compared to Wang's work and 7.37% improvement compared to Ding's work.

Since the transition-based approach makes the prediction of graph structure by predicting the transition action, once the prediction of the transition action is wrong, it will seriously affect the subsequent prediction, resulting in the low performance of dependency parsing. In our model, we use the graph-based approach to directly model the arc relationship and label information between words by using the Biaffine scorer, thus greatly improving the performance of the parser. Even so, because CSDG has a lot of non-local dependencies, and a complex dependent labeling system, CSDG paring task is still extremely challenging. We believe that the information on the dependent edge will help the model to distinguish between dependent labels. Therefore, we propose the dependency-gated cascade mechanism to add the prediction result of dependency edge into the classifier of the dependency label, which further improves the performance of our model.

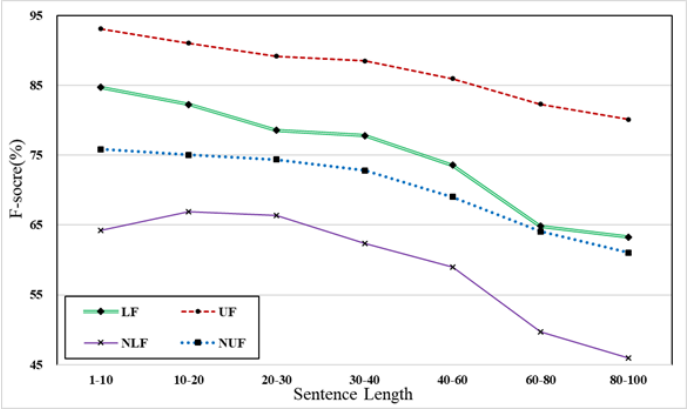


Fig. 5. The metrics of CSDG parsing under different sentence lengths

Experiments show that the performance of the CSGD parser on the TEXTBOOKS test set is lower than that on the NEWS test set. One possible reason is that the average sentence length of the NEWS dataset is longer than that of the TEXTBOOKS dataset. To test this hypothesis, as shown in figure 1, we grouped all test sets according to the length of the sentence and calculated metrics respectively. We can find that as the sentence length increases, all metrics tend to decrease. This explains to some extent why the performance of the model on the NEWS dataset dropped so much.

4.4 Analysis of the Dependency-Gated Cascade Mechanism

In order to further verify the effectiveness of the dependency-gated cascade mechanism, we compared the network performance using the cascading mechanism and the network performance without using the cascading mechanism. Both models use the same encoder layer, Biaffine network, and parameter settings.

As shown in Table 3, whether on the TEXT dataset or on the NEWS dataset, the model using the cascading mechanism is superior to the model without the cascading mechanism on four evaluation metrics.

Especially on the NLF and NUF metrics, the model using the cascading mechanism is significantly better than the model without the cascading mechanism. This shows that the cascading mechanism is of great help to the prediction of non-local dependent arcs. This is because in a sentence containing non-local dependency, the word has no more than one head nodes. In this case, the cascading mechanism can encode rich dependency path information, which can help the model better understand the semantic relationship between words.

Need to point out that on the NEWS dataset, the help of the cascading mechanism is not as obvious on the TEXT dataset. As mentioned earlier, the sentences in the NEWS dataset are longer and more complex, so it is more difficult to encode the dependent path information. The difficulty of the data affects the capabilities of the cascading mechanism, although it still improves the performance of the model.

Table 3. The ablation experiments result

Domain	Model	LF	UF	NLF	NUF
TEXTBOOKS	Our Model	**80.40**	**90.05**	**65.97**	**74.35**
	w/o Dep-gate	80.05	89.83	64.05	73.53
NEWS	Our Model	**69.66**	**84.25**	**53.34**	**66.01**
	w/o Dep-gate	69.59	84.15	51.53	65.09

In general, cascading mechanism can further improve the performance of the Biaffine network in Chinese semantic dependency graph parsing task. By encoding the dependency path information of the words, the label Biaffine classifier can predict dependent labels more accurately, thereby further improving the overall performance of the parser.

5 Conclusion

In this paper, we propose a novel dependency-gated cascade Biaffine network which contains an additional dependency-gate that allows dependent edge information predicted by the edge Biaffine classifier to be passed to the following label Biaffine classifier. And we apply this model for the Chinese Semantic Dependence Graph (CSDG) Parsing task. Our work is the first attempt to cascade the dependency edge and the dependency label predictions in an end-to-end model. We use SemEval-2016 Task 9 dataset as our benchmark dataset. Experiment result shows that our model achieves the state-of-the-art performance. The ablation experiments show that the dependency-gated cascade mechanism is effective and necessary.

References

1. Xue, Nianwen, and Martha Palmer. "Automatic semantic role labeling for Chinese verbs." IJCAI. Vol. 5. (2005).
2. Carreras, Xavier, and Lluís Màrquez. "Introduction to the CoNLL-2004 shared task: Semantic role labeling." Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004. (2004).
3. Toutanova, Kristina, Aria Haghighi, and Christopher D. Manning. "Joint learning improves semantic role labeling." Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, (2005).
4. Hajič, Jan, et al. "The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages." Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task. Association for Computational Linguistics, (2009).
5. McDonald, Ryan, et al. "Non-projective dependency parsing using spanning tree algorithms." Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing. Association for Computational Linguistics, (2005).
6. Che, Wanxiang, et al. "Semeval-2012 task 5: Chinese semantic dependency parsing." Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation. Association for Computational Linguistics, (2012).
7. Ding, Yu, et al. "Dependency graph based chinese semantic parsing." Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data. Springer, Cham,58-69, (2014).
8. Wang, Yuxuan, et al. "A neural transition-based approach for semantic dependency graph parsing." Thirty-Second AAAI Conference on Artificial Intelligence. (2018).
9. Dyer, Chris, et al. "Transition-based dependency parsing with stack long short-term memory." arXiv preprint arXiv:1505.08075 (2015).
10. Dozat, Timothy, and Christopher D. Manning. "Simpler but more accurate semantic dependency parsing." arXiv preprint arXiv:1807.01396 (2018).
11. Dozat, Timothy, Peng Qi, and Christopher D. Manning. "Stanford's graph-based neural dependency parser at the conll 2017 shared task." Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. (2017).
12. Dozat, Timothy, and Christopher D. Manning. "Deep biaffine attention for neural dependency parsing." arXiv preprint arXiv:1611.01734 (2016).
13. Chen, Danqi, and Christopher Manning. "A fast and accurate dependency parser using neural networks." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). (2014).
14. Weiss, David, et al. "Structured training for neural network transition-based parsing." arXiv preprint arXiv:1506.06158 (2015).

15. Kiperwasser, Eliyahu, and Yoav Goldberg. "Simple and accurate dependency parsing using bidirectional LSTM feature representations." Transactions of the Association for Computational Linguistics 4: 313-327, (2016).
16. Semeniuta, Stanislau, Aliaksei Severyn, and Erhardt Barth. "Recurrent dropout without memory loss." arXiv preprint arXiv:1603.05118 (2016).
17. Srivastava, Rupesh Kumar, Klaus Greff, and Jürgen Schmidhuber. "Highway networks." arXiv preprint arXiv:1505.00387 (2015).
18. Moon, Taesup, et al. "Rnndrop: A novel dropout for rnns in asr." 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU). IEEE, (2015).
19. Zilly, Julian Georg, et al. "Recurrent highway networks." Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, (2017).
20. Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. (2013).
21. Peng, Hao, Sam Thomson, and Noah A. Smith. "Deep multitask learning for semantic dependency parsing." arXiv preprint arXiv:1704.06855 (2017).