Stack-based Multi-layer Attention for Transition-based Dependency Parsing

Zhirui Zhang[1], **Shujie Liu**[2], **Mu Li**[2], **Ming Zhou**[2], **Enhong Chen**[1]
University of Science and Technology of China, Hefei, China[1]
Microsoft Research Asia, Beijing, China[2]
zr011036@mail.ustc.edu.cn cheneh@ustc.edu.cn[1]
{shujie,muli,mingzhou}@microsoft.com[2]

Abstract

Although sequence-to-sequence (seq2seq) network has achieved significant success in many NLP tasks such as machine translation and text summarization, simply applying this approach to transition-based dependency parsing cannot yield a comparable performance gain as in other state-of-the-art methods, such as stack-LSTM and head selection. In this paper, we propose a stack-based multi-layer attention model for seq2seq learning to better leverage structural linguistics information. In our method, two binary vectors are used to track the decoding stack in transition-based parsing, and multi-layer attention is introduced to capture multiple word dependencies in partial trees. We conduct experiments on PTB and CTB datasets, and the results show that our proposed model achieves state-of-the-art accuracy and significant improvement in labeled precision with respect to the baseline seq2seq model.

1 Introduction

Deep learning models have been proven very effective in solving various NLP problems such as language modeling, machine translation and syntactic parsing. For dependency parsing, one line of research aims to incrementally integrate distributed word representations into classic dependency parsing (Chen and Manning, 2014; Weiss et al., 2015; Andor et al., 2016; Cross and Huang, 2016; Kiperwasser and Goldberg, 2016; Dozat and Manning, 2016). Another line of research attempts to leverage end-to-end neural network to perform dependency parsing, such as stack-LSTM and sequence-to-sequence (seq2seq) model (Dyer et al., 2015; Zhang et al., 2017; Wiseman and Rush, 2016). Recently seq2seq model has made significant success in many NLP tasks, such as machine translation and text summarization (Cho et al., 2014; Sutskever et al., 2014; Rush et al., 2015). Unfortunately, to our best knowledge, simply applying seq2seq model to transition-based dependency parsing cannot achieve comparable results as in other state-of-the-art methods like stack-LSTM and head selection (Dyer et al., 2015; Zhang et al., 2017).

One issue with the simple seq2seq neural network for dependency parsing is that structural linguistic information, which plays a key role in classic transition-based or graph-based dependency parsing models, cannot be explicitly employed. For example, classic transition-based parsing algorithm utilizes a stack to manage the heads of partial sub-trees and leverages these evidents for action selection, while such information is missing from current seq2seq models. Another problem is related to the limit of the conventional attention network being used in seq2seq network, which is unable to capture dependencies between words in the input. As a matter of fact, various types of features (word unigram, bigram, trigram, ...) traditionally adopted by transition-based parsing algorithm are usually ignored by the current attention mechanism, but they are very important to capture word dependencies in generated partial trees.

In this paper, we propose a stack-based multi-layer attention mechanism to solve the above problems. To simulate the stack used in the transition-based dependency parsing, we introduce two binary vectors, one indicates whether a word is pushed into the stack, and another indicates whether a word is popped out from it. To model the complex structural information, we propose a multi-layer attention based on the stack information, previous action and input sentence.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1678–1683
Copenhagen, Denmark, September 7–11, 2017. ©2017 Association for Computational Linguistics

The multi-layer attention aims to capture multiple word dependencies in partial trees for action prediction.

We evaluate our model on English and Chinese datasets. Experimental results show that our proposed model can significantly outperform the basic seq2seq model with 1.87 UAS (English) and 1.61 UAS (Chinese), matching the state-of-the-art parsing performance. With 4 models ensembled, we obtain further improvements with accuracies of 94.16 UAS (English) and 87.97 UAS (Chinese).

2 Neural Model for Sequence-to-Sequence Learning

In this work, we follow the encoder-decoder architecture proposed by Bahdanau et al. (2015). The whole architecture can be divided into three components: encoder, decoder and attention.

Encoder: The encoder reads in the source sentence $X = (x_1, x_2, \dots, x_T)$ and transforms it into a sequence of hidden states $h = (h_1, h_2, \dots, h_T)$, using a bi-directional recurrent neural network that is usually implemented as Gated Recurrent Unit (GRU) (Cho et al., 2014) or Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997).

Attention Mechanism: The context vector c_i is a weighted sum of the hidden states $(h_1, h_2, \dots, h_T)$ with the coefficients $\alpha_{i,1}, \alpha_{i,2}, \dots, \alpha_{i,T}$ computed by

$$\alpha_{i,t} = \frac{\exp(e_{i,t})}{\sum_k \exp(e_{i,k})} \tag{1}$$

$$e_{i,t} = v_a^\top \tanh(W_a z_{i-1} + U_a h_t) \tag{2}$$

where v_a,W_a, U_a are the weight matrices.

Decoder: The decoder uses another recurrent neural network to generate a corresponding target sequence $Y = (y_1, y_2, \dots, y_{T'})$ based on the encoded sequence of hidden state h. At each time i, the conditional probability of target symbol y_i is computed by

$$z_i = \text{RNN}([y_{i-1}; c_i], z_{i-1}) \tag{3}$$

$$p(y_i|y_{<i}, h) = \text{softmax}(g(y_{i-1}, z_i, c_i)) \tag{4}$$

where g is a non-linear function, z_i is the i_{th} hidden state of the decoder, and it is calculated conditional on the previous hidden state z_{i-1}, previous target symbol y_{i-1} and source context vector c_i.

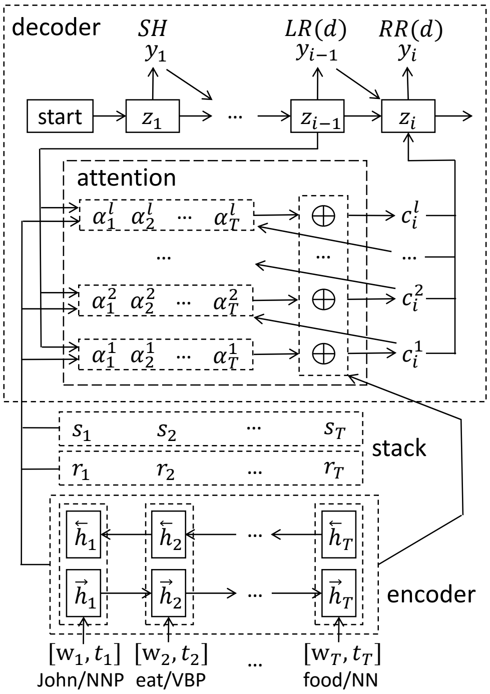
decoder
SH
y_1
$LR(d)$
y_{i-1}
$RR(d)$
y_i
start
z_1
...
z_{i-1}
z_i
attention
$\alpha_1^l \; \alpha_2^l \; \cdots \; \alpha_T^l$
c_i^l
...
$\alpha_1^2 \; \alpha_2^2 \; \cdots \; \alpha_T^2$
c_i^2
$\alpha_1^1 \; \alpha_2^1 \; \cdots \; \alpha_T^1$
c_i^1
$s_1 \; s_2 \; \cdots \; s_T$
stack
$r_1 \; r_2 \; \cdots \; r_T$
$\overleftarrow{h}_1 \; \overleftarrow{h}_2 \; \cdots \; \overleftarrow{h}_T$
$\overrightarrow{h}_1 \; \overrightarrow{h}_2 \; \cdots \; \overrightarrow{h}_T$
encoder
$[w_1, t_1]$ $[w_2, t_2]$ ... $[w_T, t_T]$
John/NNP eat/VBP food/NN

Figure 1: The architecture of sequence-to-sequence parsing model. SH, LR(d), RR(d) denote the SHIFT, LEFT-ARC(d), RIGHT-ARC(d) transitions in arc-standard algorithm and d is arc-label.

3 Sequence-to-Sequence Parsing Model

Transition-based dependency parsing conceptualizes the process of transforming a sentence into a dependency tree as a sequence of actions. It can be formulated as a sequence-to-sequence problem, and seq2seq framework can be applied. Compared with other tasks, such as machine translation, dependency parsing not only considers the previous action and input sentence, but also requires many structure information, such as the subtree structure during the parsing. Such information plays an important role in transition-based dependency parsing, so traditional methods adopt a stack to save structure information and design different type of features (word unigram, bigram, trigram, ...) to model them. However, vanilla seq2seq models have no explicit structure to model these necessary structure information. To better leverage the structure information, we extend the basic seq2seq architecture with a simulated stack and multi-layer attention, as illustrated in Figure 1. The main structure (encoder, decoder and attention part in Figure 1) of our parsing model is detailed below.

Encoder: As shown in the encoder part of Figure 1, to utilize POS tag information, each word w_i is additionally represented by x_i, the concatenation of two vectors corresponding to w_i's lexical and POS tag t_i embedding: $x_i = [W_e * e(w_i); W_t * e(t_i)]$, where $e(w_i)$ and $e(t_i)$ are one-hot vector representations of token w_i and its POS tag t_i, W_e and W_t are embedding matrices. The rest part of the encoder is the same with the basic seq2seq model.

Attention Mechanism: We improve the attention part in two aspects: introduction of stack information and multi-layer attention structure.

Stack information, which plays an essential role in the conventional algorithm, is simulated with two binary vectors $s = (s_1, \dots, s_T)$ and $r = (r_1, \dots, r_T)$ to record the state of each word w_i and initialized to zero. When parser pushes word w_i into stack, s_i is assigned to 1, while r_i is assigned to 1 only if word w_i is removed from stack. Intuitively, at each time step i in the decoding phase, stack information serves as an additional input to the attention model, which provides complementary information of that the source words is in the stack or not. We expect the stack information would guide the attention model to focus more on words in the stack. More formally, the coefficients $\alpha_1, \alpha_2, \dots, \alpha_T$ used in attention mechanism can be rewritten as

$$\alpha_{i,t} = \frac{\exp(e_{i,t}) * (1 - r_t)}{\sum_k \exp(e_{i,k}) * (1 - r_k)} \tag{5}$$

$$e_{i,t} = v_a^\top \tanh(W_a z_{i-1} + U_a h_t + S_a s_t) \tag{6}$$

where S_a is the weight matrix.

To extract complex structure information to help action prediction, we apply a l-layers network structure for attention mechanism as shown in the attention part of Figure 1. To further enhance connection between adjacent layers, we replace the state z_{i-1} in Equation 6 by the concatenation of z_{i-1} and context vector c_i^{m-1} at each layer $m(m > 1)$. The Equation 6 can be rewritten as:

$$e_{i,t}^m = v_a^\top \tanh(W_a^m [z_{i-1}; c_i^{m-1}] + U_a h_t + S_a s_t) \tag{7}$$

where W^m is the weight matrix. With this network structure, we obtain different context vectors $(c_i^1, c_i^2, \dots, c_i^l)$, and the final context vector c_i, which is considered as complex context information, is replaced by the concatenation of those vectors: $c_i = [c_i^1; c_i^2; \dots; c_i^l]$.

Decoder: Unlike machine translation and text summarization in which seq2seq model is widely applied, a sequence of action in dependency parsing must satisfy some constraints so that they can generate a dependency tree. Following the arc-standard algorithm (Nivre, 2004), the precondition can be categorized as 1) SHIFT(SH): There exists at least one word that is not pushed into the stack; 2) LEFT-ARC(LR(d)) and RIGHT-ARC(RR(d)): There are at least two words in the stack. These two constraints can be defined as indicator functions

$$I(y_i) = \begin{cases} 0 & y_i = \text{SH}, W_c \le 0 \\ 0 & y_i = \text{LR(d) or RR(d)}, S_c < 2 \\ 1 & \text{otherwise} \end{cases} \tag{8}$$

where S_c represents the number of words in the stack and W_c is the number of source words that are not pushed into the stack. To introduce these constraints, the conditional probability of each target symbol y_i can be rewritten as

$$p(y_i | y_{<i}, h) = \frac{\exp(g_i) * I(y_i)}{\sum_k \exp(g_k) * I(y_k)} \tag{9}$$

where g_i is the ith element of $g(y_{i-1}, z_i, c_i)$.

4 Experiments

In this section, we evaluate our parsing model on the English and Chinese datasets. Following Dyer et al. (2015), Stanford Dependencies (de Marneffe and Manning, 2008) conversion of the Penn Treebank (PTB) (Marcus et al., 1993) and Chinese Treebank 5.1 (CTB) are adopted. We leverage the arc-standard algorithm for our dependency parsing. In addition, we limit the vocabulary to contain up to 20k most frequent words and convert remaining words into the <unk> token.

4.1 Setup

For our model, 3-layers GRU is used for encoder and decoder. The dimension of word embedding is 300, the dimension of POS-tag/action embedding is 32, and the size of hidden units in GRU is 500. 3-layers attention structure is adopted in our model. Following Chen and Manning (2014); Dyer et al. (2015), we used 300-dimensional pretrained GloVe vectors (Pennington et al., 2014) to initialize our word embedding matrix. Other model parameters are initialized using a normal distribution with a mean of 0 and a variance of

Parser	PTB-SD				CTB			
	Dev		Test		Dev		Test	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
Z&N11	-	-	93.00	90.95	-	-	86.00	84.40
C&M14	92.20	89.70	91.80	89.60	84.00	82.40	83.90	82.40
ConBSO	-	-	91.57	87.26	-	-	-	-
Dyer15	93.20	90.90	93.10	90.90	87.20	85.90	87.20	85.70
Weiss15	-	-	93.99	92.05	-	-	-	-
K&G16	-	-	93.99	91.90	-	-	87.60	86.10
DENSE	**94.30**	91.95	94.10	91.90	87.35	85.85	87.84	86.15
seq2seq	92.02	89.10	91.84	88.84	86.21	83.80	85.80	83.53
Our model	93.65	91.52	93.71	91.60	87.28	85.30	87.41	85.40
Ensemble	94.24	**92.01**	**94.16**	**92.13**	**88.06**	**86.30**	**87.97**	**86.18**

Table 1: Results of various state-of-the-art parsing systems on English dataset (PTB with Stanford Dependencies) and Chinese dataset (CTB). The numbers reported from different systems are taken from: Z&N11 (Zhang and Nivre, 2011); C&M14 (Chen and Manning, 2014); ConBSO (Wiseman and Rush, 2016); Dyer15 (Dyer et al., 2015); Weiss15 (Weiss et al., 2015); K&G16 (Kiperwasser and Goldberg, 2016); DENSE (Zhang et al., 2017).

$\sqrt{6/(d_{row} + d_{col})}$, where d_{row} and d_{col} are the number of rows and columns (Glorot and Bengio, 2010). Our models are trained on a Tesla K40m GPU and optimized with vanilla SGD algorithm with mini-batch size 64 for English dataset and 32 for Chinese dataset. The initial learning rate is set to 2 and will be halved when unlabeled attachment scores (UAS) on the development set do not increase for 900 batches. To alleviate the gradient exploding problem, we rescale the gradient when its norm exceeds 1. Dropout (Srivastava et al., 2014) is applied to our model with the strategy recommended in Zaremba et al. (2014) and the dropout rate is 0.2. For testing, beam search is employed to find the best action sequence with beam size 8. For evaluation, we report unlabeled (UAS) and labeled attachment scores (LAS) on the development and test sets. Following Chen and Manning (2014), the punctuation is excluded from the evaluation.

4.2 Main Results

Table 1 lists the accuracies of our parsing models, compared to other state-of-the-art parsers. For the baseline, seq2seq model employs the same encoder and decoder network structure with our model. We can see that our proposed model can significantly outperform the basic seq2seq model with 1.87 UAS (English) and 1.61 UAS (Chinese) improvements on the test set. This demonstrates the effectiveness of our proposed multi-layer attention mechanism. Besides, our model achieves better UAS accuracy than Z&N11, C&M14, ConBSO and Dyer15 on development and test set, while slightly lower than Weiss15, K&G16 and DENSE. Weiss15 adopts a structured training procedure which can be easily applied to our model as well, and it will further improve the performance of our model. K&G16 uses 11 bidirectional LSTM vectors as features, which will be fed to a transition-based parser. It suggests a new direction that combines our model with feature engineering of the traditional transition-based parser to gain better performance. DENSE formalizes dependency parsing as head selection and applies MST algorithms to correct non-tree outputs, while our model doesn't require any post-processing at test time. Dozat and Manning (2016) use deep biaffine attention instead of traditional attention in the graph-based architectures of K&G16, achieving 95.74 UAS and 89.30 UAS on PTB-SD and CTB datasets respectively. For ensemble, we train 4 models using the same network with different random initialization. When we ensemble these 4 models, we simply average the output probabilities from different models and obtain the better result with accuracies of 94.16 UAS (English) and 87.97 UAS (Chinese) as shown in the Table 1.

4.3 Impact of l

The hyper-parameter l represents the number of layers in our proposed multi-layer attention mech-

	Dev		Test	
	UAS	LAS	UAS	LAS
seq2seq	92.02	89.10	91.84	88.84
$l = 1$	92.85	90.44	92.70	90.40
$l = 2$	93.30	91.13	93.21	90.98
$l = 3$	**93.65**	**91.52**	**93.71**	**91.60**
$l = 4$	93.49	91.29	93.42	91.24

Table 2: Impact of l on English PTB dataset.

anism. Larger l would bring more capacity, but lead to more computational complexity and aggravate the risk of over-fitting.

We conduct a group of experiments to investigate the impact of l. The results are shown in Table 2. Seq2seq model can be viewed as a special case of our model without any stack information. With $l = 1$, we can see that the introduction of stack information can strongly improve the parsing performance, especially for LAS. When l is small ($l < 4$), the general trend is that larger l leads to better result. However, further increasing l bring slightly damages to the parsing performance due to the over-fitting problem.

Although larger l would bring more capacity, multiple layers structure will double the training time compared with the vanilla seq2seq. In our implementation, our model costs about 500 seconds for a round of training data on English PTB dataset, while the vanilla costs about 260 seconds.

4.4 Additional Results

We perform some ablation experiments in order to quantify the effect of the different components on our models. As shown in Table 3, the POS-tag information plays the most important role in our model. We note that, different from Dyer et al. (2015), we don't utilize an external word embedding to tackle OOV problem, and it may cause our model to be more dependent on the POS-tag information. For s and r vectors, same as discussion in last section, we find that the introduction of stack information can strongly improve the parsing performance.

5 Conclusion

In order to leverage structure information for seq2seq based dependency parsing, in this paper, we propose a stack based multi-layer attention method, in which, stack is simulated with two binary vectors, and multi-layer attention is introduced to capture multiple word dependencies in partial trees. Experimental results demonstrate that our proposed model significantly outperforms the basic seq2seq model, and achieves a state-of-the-art parsing performance.

	Dev		Test	
	UAS	LAS	UAS	LAS
Our model	93.65	91.52	93.71	91.60
–pretraining	93.19	90.92	93.22	91.11
–POS	92.73	89.86	92.57	90.05
–s vector	93.18	90.68	93.02	90.89
–r vector	93.16	90.90	93.27	91.02

Table 3: Impact of the different components on English PTB dataset.

In the future, we plan to apply our approach in more languages and other transition-based system, such as arc-eager or arc-hybrid. Another direction we are interested in is to train our model with complex training approaches proposed in Weiss et al. (2015) and Andor et al. (2016).

Acknowledgments

We appreciate Dongdong Zhang, Nan Yang, Shuangzhi Wu and Qingyu Zhou for the fruitful discussions. We also thank the anonymous reviewers for their careful reading of our paper and insightful comments.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. *CoRR*, abs/1603.06042.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the Third International Conference on Learning Representations (ICLR)*.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics.

James Cross and Liang Huang. 2016. Incremental parsing with minimal features using bi-directional lstm. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 32–37, Berlin, Germany. Association for Computational Linguistics.

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

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *ACL*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256.

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

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *CoRR*, abs/1603.04351.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19:313–330.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57. Association for Computational Linguistics.

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

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *ACL*.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas. Association for Computational Linguistics.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *CoRR*, abs/1409.2329.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 665–676, Valencia, Spain. Association for Computational Linguistics.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *ACL*.