The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Improving Sequence-to-Sequence Constituency Parsing

Lemao Liu, Muhua Zhu, Shuming Shi
Tencent AI Lab, Shenzhen, China
{redmondliu,muhuazhu, shumingshi}@tencent.com

Abstract

Sequence-to-sequence constituency parsing casts the tree-structured prediction problem as a general sequential problem by top-down tree linearization, and thus it is very easy to train in parallel with distributed facilities. Despite its success, it relies on a probabilistic attention mechanism for a general purpose, which can not guarantee the selected context to be informative in the specific parsing scenario. Previous work introduced a deterministic attention to select the informative context for sequence-to-sequence parsing, but it is based on the bottom-up linearization even if it was observed that top-down linearization is better than bottom-up linearization for standard sequence-to-sequence constituency parsing. In this paper, we thereby extend the deterministic attention to directly conduct on the top-down tree linearization. Intensive experiments show that our parser delivers substantial improvements over the bottom-up linearization in accuracy, and it achieves 92.3 Fscore on the Penn English Treebank section 23 and 85.4 Fscore on the Penn Chinese Treebank test dataset, without reranking or semi-supervised training.

Introduction

Constituency parsing is a fundamental task in natural language processing, and it plays an important role in downstream applications such as machine translation (Galley et al. 2004; 2006) and semantic analysis (Rim, Seo, and Simmons 1990; Manning, Schütze, and others 1999). Over the decades, feature-rich linear models had been dominant in constituency parsing (Petrov and Klein 2007; Zhu et al. 2013); but they are not good at capturing the long distance dependencies due to feature sparsity. Recurrent neural networks have the advantages to address such issue, and recently there has been much work on recurrent neural models for constituency parsing (Vinyals et al. 2015; Watanabe and Sumita 2015; Dyer et al. 2016; Cross and Huang 2016).

In particular, sequence-to-sequence parsing (Vinyals et al. 2015) has been increasingly popular. Its basic idea is to linearize a parse tree into a sequence in a top-down manner (see Figure 1) and then transform parsing into a standard sequence-to-sequence learning task. The main technique inside the sequence-to-sequence parsing is a probabilistic attention mechanism, which aligns an output token to input tokens to select relevant context for better prediction as shown in Figure 2(a). This parsing model is general and easy to understand; particularly it runs in a sequential manner and thus is easy to parallelize with GPUs. However, the probabilistic attention can not guarantee the selected context is informative enough to yield satisfactory outputs. As a result, its accuracy is only comparable to the feature-rich linear models (Petrov and Klein 2007; Zhu et al. 2013), especially given that it utilizes global context.

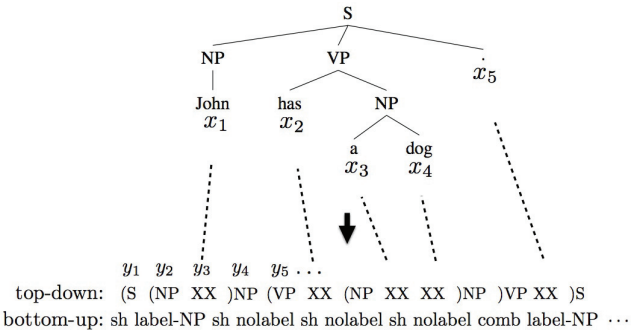
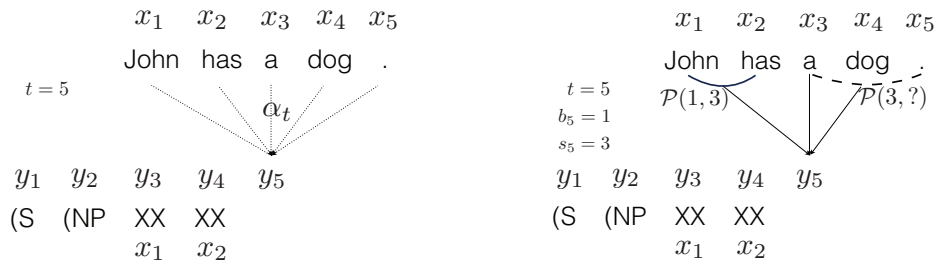


Figure 1: Top-down and bottom-up linearlization of a parse tree in sequence-to-sequence constituency parsing. The input sequence $\mathbf{x}$ is the leaves of the parse tree in the top, and the output is the linearized sequence $\mathbf{y}$ in the bottom. A dash line indicates the relation between x_i and "XX".

Ma et al. (2017) proposed a deterministic attention for sequence to sequence parsing, which defines the alignments between output and input tokens in a deterministic manner to select the relevant context. This method was able to select better context than probabilistic attention for parsing. However, their approach was conducted on the bottom up linearization (see its linearized sequence in Figure 1) and they require to binarize a parse tree, which induces the issue of ambiguity: different binarized trees may lead to the same tree. In addition, the bottom-up linearization lacks of top-down guidance such as lookahead information, which has been proved to be useful for better prediction (Roark and Johnson 1999; Liu and Zhang 2017b). As a result, their parser is still worse than the state of the art parsers in accuracy.



(a). Probabilistic attention (b). Deterministic attention

Figure 2: The figures for probabilistic attention (a) and deterministic attention (b). At the timestep $t = 5$, $y_{<5}$ have been available but y_t is unavailable and will be predicted next using context by attentions. Probabilistic attention aligns y_5 to all tokens according to a distribution α_t shown in dotted arrow lines, while deterministic attention aligns y_5 to the phrase $\mathcal{P}(1,3)$, the semi-phrase $\mathcal{P}(3,?)$ and x_3 in a deterministic manner solid shown in arrow lines.

In this paper, therefore, we aim to explore the deterministic attention directly on top of top-down linearization, with the expectation to improve the sequence-to-sequence constituency parsing. The proposed deterministic attention is inspired by the following intuition. When linearizing a parse tree in a top-down manner, it is clear that each token "XX" represents a known word in the input side by a dash line as shown in Figure 1, and thus this output token might deterministically align to that specific token in the input side rather than stochastically align to all input tokens. Respecting this intuition, we analyze the ideal alignment situations at each decoding timestep and propose a general deterministic attention criteria for context selection. Under this criteria, we propose some simple instances to deterministically specify the alignments between input and output tokens (§3). Since our deterministic attention sequentially represents alignments for a given parse tree, its training still performs in a sequential manner and thus is easy to parallelize as the standard sequence-to-sequence parsing does. Empirical experiments demonstrate that the resulting deterministic attention on top-down linearization achieves substantial gains over the model in Ma et al. (2017). Furthermore, with the help of ensemble, the proposed parser is competitive to state-of-the-art RNN parsers (Dyer et al. 2016; Stern, Andreas, and Klein 2017), which require to maintain tree structures and thus are not easy to parallelize for training.

This paper makes the following contributions:

- It analyzes the deterministic attention for sequence-to-sequence parsing on top of top-down linearization, and proposes a simple yet effective model without increasing the training time.
- On both Penn English and Chinese Treebank datasets, intensive experiments show that our parser outperforms several direct sequence-to-sequence baselines, and achieve 92.3 Fscore on English dataset and 85.4 Fscore on Chinese dataset without reranking or semi-supervised training.

Sequence-to-Sequence Parsing

Suppose $\mathbf{x} = \langle x_1, x_2, \cdots, x_{|\mathbf{x}|} \rangle$ denotes an input sequence with length $|\mathbf{x}|$; and $\mathbf{y} = \langle y_1, y_2, \cdots, y_{|\mathbf{y}|} \rangle$ denotes the output sequence which represents a linearized parse tree of $\mathbf{x}$ via a linearization method such as top-down linearization. In Figure 1, $\mathbf{x}$ is the sequence $\langle \text{John}, \text{has}, \text{a}, \text{dot}, . \rangle$ and $\mathbf{y}$ is its linearized tree sequence $\langle (\text{S}, (\text{NP}, \text{XX}, \cdots, \text{XX},)\text{S} \rangle$.

Generally, sequence-to-sequence constituency parsing directly maps an input sequence to its linearized parse tree sequence by using a neural machine translation model (NMT) (Vinyals et al. 2015; Bahdanau, Cho, and Bengio 2014). NMT relies on recurrent neural networks under the encode-decode framework including two stages. In the encoding stage, it applies recurrent neural networks to represent $\mathbf{x}$ as a sequence of vectors:

$$h_i = f(e_{x_i}, h_{i-1}),$$

where the hidden unit h_i is a vector with dimension d at timestep i, f is a recurrent function such as LSTM and GRU and e_x denotes the word embedding of x. Suppose the encoding sequence is denoted by $E^{\mathbf{x}} = \left\langle E_1^{\mathbf{x}}, E_2^{\mathbf{x}}, \cdots, E_{|\mathbf{x}|}^{\mathbf{x}} \right\rangle$. Then each $E_i^{\mathbf{x}}$ can be set as h_i from a reversed recurrent neural network (Vinyals et al. 2015) or as the concatenation of the hidden units from bidirectional recurrent neural networks (Ma et al. 2017).

In the decoding stage, it generates a linearized sequence from the conditional probability distribution defined by a recurrent neural network as follows:

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{x}; \theta) &= \prod_{t=1}^{|\mathbf{y}|} p(y_t \mid y_{<t}, E^{\mathbf{x}}) \\ &= \prod_{t=1}^{|\mathbf{y}|} \mathbf{softmax}\left(g(e_{y_{t-1}}, h'_t, c_t)\right)[y_t], \quad (1) \end{aligned}$$

with

$$h'_t = f'(h'_{t-1}, y_{t-1}, c_t)$$

where θ is the overall parameter of this model; $y_{<t} = \langle y_1, y_2, \cdots, y_{t-1} \rangle$; e_y denotes the embedding of y; g is a

projection function mapping into a vector with dimension of the output vocabulary size V; and $[i]$ denotes the i_{th} component of a vector; c_t is a context vector at timestep i and h'_t is a hidden unit specified by a RNN unit f similar to f defined in encoding stage.

As shown in Eq.(1), c_t is used to not only update the hidden unit h_t in f but also predict y_t in $\mathbf{softmax}$, and thus it is critical for NMT. In order to capture informative information in context c_t, NMT employs a probabilistic attention, which aligns the token y_t to tokens in $\mathbf{x}$. In detail, at each timestep t, c_t is defined as the weighted sum of input encodings $E^{\mathbf{x}}$ with weight α_t:

$$c_t = \phi(\alpha, E^{\mathbf{x}}) = \alpha_t^{\top} E^{\mathbf{x}}.$$

Here the weight α_t implements an *probabilistic* attention mechanism indicating that $\alpha_{t,i}$ is the alignment probability of y_t being aligned to x_i. α_t is derived as follows:

$$\alpha_t = \mathbf{softmax}\left(A(h'_{t-1}, E^{\mathbf{x}})\right), \tag{2}$$

where A is a feedforward neural network mapping h'_{t-1} and $E^{\mathbf{x}}$ into a $|\mathbf{x}|-$dimension vector; and $\mathbf{softmax}$ makes this vector to be a distribution, indicating that y_t stochastically aligns to all tokens in $\mathbf{x}$ as shown in Figure 2(a).

The Proposed Parsing Model

Despite of the success of sequence-to-sequence parsing, its context is defined by a probabilistic attention which can not guarantee that the selected context is always informative for reliable predictions. Ma et al. (2017) proposed an approach based on the deterministic attention to address this issue. Unlike the probabilistic attention, the deterministic attention only aligns to some informative tokens in the input side and thus selects their representations as the context at each prediction timestep t. In order to specify those informative tokens easily, they developed the deterministic attention on top of bottom up linearization in an indirect manner, even if it was observed that top-down linearization is better for the standard sequence-to-sequence parsing in their experiments. Inspired by this observation, this paper aims to apply the deterministic attention directly based on the top-down linearization.

Top-down Deterministic Attention

Suppose $\mathbf{x}$ is an input sequence and $\mathbf{y}$ is its top-down linearized tree. According to the tree linearization procedure, there are some natural alignment signals for "XX" tokens in $\mathbf{y}$ if $\mathbf{y}$ is given. For example, in Figure 1, y_3 = "XX" represents x_1 = "John" and thus y_3 may deterministically align to x_1 at least. For other tokens in $\mathbf{y}$ with the forms of ")" and "(", we consider the following two intuitive cases in Figure 1.

In Figure 1, y_4 = "NP)", the label "NP" in y_4 shows that there is a noun phrase "John" spanning $[1, 2)$ in the input side, and thus y_4 may align to this "NP" phrase. In this paper, this phrase is denoted by $\mathcal{P}(b_t, s_t)$, where b_t and s_t denote the beginning and stopping positions. In this case, $b_4 = 1$ and $s_4 = 2$. Similarly, y_5 = "(VP" in Figure 1, the label "VP" in y_5 indicates that the next phrase is a verb phrase starting at "has", and thus y_5 may align to this noun phrase. As the stopping position of this phrase is unknown, we call it a *semi-phrase* represented by $\mathcal{P}(b_t, ?)$ throughout this paper. In this case, $b_5 = 2$.

The above analysis is the exact case in the training stage, where the entire tree sequence $\mathbf{y}$ is given in Figure 1. However, in testing stage the entire $\mathbf{y}$ is not available but generated incrementally starting from the beginning timestep $t = 1$ as in Figure 2(b). Suppose in testing stage the decoder has already generated the tokens $\mathbf{y}_{<t}$ and reaches the timestep t. For each $k < t$ satisfying y_k="XX", we can specify some k' such that y_k represents $x_{k'}$. Then it is easy to calculate b_t and s_t in Figure 2(b), where $t = 5$, $b_5 = 1$ and $s_5 = 3$. At this moment, although y_t is not available but to be generated next, there are three choices for y_t, "XX", a form of ")" or a form of "(":

- y_t aligns to x_{s_t} if y_t is to be "XX".
- y_t aligns to the phrase $\mathcal{P}(b_t, s_t)$ if y_t is to be ")".
- y_t aligns to the semi-phrase $\mathcal{P}(s_t, ?)$ if y_t is to be "(".

Accordingly, for an unknown y_t at the timestep t, we define its alignments to the union of $\mathcal{P}(b_t, s_t)$, $\mathcal{P}(s_t, ?)$ and e_t, instead of a single alignment case. For example in Figure 2(b), even if y_5 will be "XX", its alignment is not the input token x_3 it represented, but the union of $\mathcal{P}(1, 3)$, $\mathcal{P}(3, ?)$ and x_3. This alignment criteria makes our attention strategy different from the probabilistic attention stochastically aligning to all tokens in $\mathbf{x}$ as shown in Figure 2(a).

Interestingly, Liu and Zhang (2017a) recently proposed an improved attention mechanism for sequence-to-sequence parsing. They split $\mathbf{x}$ into two parts $x_{<s_t}$ and $\mathbf{x}/x_{<s_t}$, and define two different probabilistic distributions for each parts. Therefore, their model stills fall into the framework of probabilistic attention as Vinyals et al. (2015) while ours is beyond this framework. Furthermore, our experiments will demonstrate that our model delivers substantial gains over their model (see §4).

Definition of Context Function

Instead of defining the context using the probabilistic attention, we define the context c_t for y_t based on the deterministic attention as following:

$$c_t = \phi\big(\mathcal{P}(b_t, s_t), \mathcal{P}(s_t, ?), s_t, E^{\mathbf{x}}\big), \tag{3}$$

where ϕ denotes the context function. To specify the context function ϕ, one requires to represent the phrase $\mathcal{P}(b_t, s_t)$ and the semi-phrase $\mathcal{P}(s_t, ?)$ based on the input sequence representation $E^{\mathbf{x}}$. In this paper, we employ some simple methods to represent both the phrase and the semi-phrase, based on their boundary representations.

The first one represents the phrase $\mathcal{P}(b_t, s_t)$ as the concatenation of both its boundary representations, and represents the semi-phrase $\mathcal{P}(s_t, ?)$ as the representation of s_t. Thus, ϕ is defined as follows:

$$\phi = \theta^c \cdot [E^{\mathbf{x}}_{b_t}; E^{\mathbf{x}}_{s_t-1}; E^{\mathbf{x}}_{s_t}] \tag{4}$$

where θ^c denotes the parameter of the context function and it is a component of the entire parameter θ; $[\,\cdot\,;\,\cdot\,;\,\cdot\,]$ denotes the concatentation of vectors.

The context function ϕ in Eq.(4) actually defines on the x_{b_t}, x_{s_t-1} and x_{s_t}. As x_{s_t} and x_{s_t-1} are adjacent in $E^{\mathbf{x}}$ derived from RNNs, $E^{\mathbf{x}}_{s_t}$ may contain some information from x_{s_t-1}. We thereby reduce the context function such that it depends on two words x_{b_t}, x_{s_t} as follows:

$$\phi = \theta^c \cdot [E^{\mathbf{x}}_{b_t}; E^{\mathbf{x}}_{s_t}] \tag{5}$$

Additionally, since s_t is the stopping position of $\mathcal{P}(b_t, s_t)$ and the beginning position of $\mathcal{P}(s_t, ?)$, $E^{\mathbf{x}}_{s_t}$ encodes some information of both. We further reduce the dependency of ϕ to only one word x_{s_t} as follows:

$$\phi = \theta^c \cdot E^{\mathbf{x}}_{s_t} \tag{6}$$

Please note that our representations for a phrase and a semi-phrase are defined on their boundaries rather than their entire structures, and especially we do not consider the stopping boundary for a semi-phrase at all. Thus there may be some further improvements in their representations. For example, Cross and Huang (2016) and Stern, Andreas, and Klein (2017) employ some sophisticated methods to represent a phrase in syntactic parsing. In addition, we simply employ a linear layer to define ϕ, and it is promising to adopt a GRU layer to summarize this context together as shown in Tu et al. (2017a). As the above simple method works well in our experiments, and we instead remain these advanced methods as our future work.

Detailed Model

In encoding of our model, we define the representation of x_i by using three different types of word embeddings similar to Dyer et al. (2015) and Liu and Zhang (2017a). We define e_{x_i} as the concatenation of three embeddings:

$$e_{x_i} = [\hat{e}_{x_i}; \hat{e}_{p_i}; \bar{e}_{x_i}],$$

where $\hat{e}_{x_i}$ is a word embedding of x_i, $\hat{e}_{p_i}$ is an embedding of the pos tag of x_i, and $\bar{e}_{x_i}$ is a pretrained embedding of x_i. Both $\hat{e}_{x_i}$ and e_{p_i} are randomly initialized and tuned in the training data; while $\bar{e}_{x_i}$ is fixed as a constant during the training. While for the word embedding y_t in the output side, we simply consider it as a variable and tune it during the training.

Until now, we complete the definition of our parsing model: if one substitutes the Eq.(3) with ϕ specified in Eq.(4), Eq.(5) or Eq.(6) into Eq.(1) one can obtain its entire definition.

Note that our model is on top of deterministic attention, and it requires to figure out b_t and s_t at each timestep, to which our context function is attentive. Fortunately, we can still maintain the similar complexity to the standard sequence-to-sequence parsing model (Vinyals et al. 2015) in both the training and testing stages. In training stage, for a given $\mathbf{x}$ and $\mathbf{y}$ pair, we can calculate the b_t and s_t at each timestep t in advance and then store them as an additional input sequence to calculate the log probability $P(\mathbf{y} \mid \mathbf{x})$. In this way, our neural architecture does not maintain the tree structure and thus our training procedure still performs in a sequential manner, which is easy for parallelization. This advantage makes it different from the models in Dyer et al. (2016) and Stern, Andreas, and Klein (2017), which require to explicitly maintain the tree structure in neural networks. In testing stage[1], although we have to calculate both b_t and s_t on-the-fly, its additional complexity is neglected compared with the calculation of neural models involving in large matrix multiplications.

Experiments

Datasets

The experiments are conducted on both English and Chinese constituent parsing tasks. For English task, we use the benchmark of WSJ sections in Penn Treebank (PTB) (Marcus, Marcinkiewicz, and Santorini 1993), and we follow the standard splits: the training data ranges from section 2 to section 21; the development data is section 24; and the test data is section 23. For Chinese parsing task, we use the Penn Chinese Treebank (CTB) of the version 5.1 (Xue et al. 2005). The training data includes the articles 001-270 and articles 440-1151; the development data is the articles 301-325; and the test data is the articles 271-300. The statistics of these datasets are shown in Table 1. We pretrained the English word embeddings on the AFP portion of English Gigaword and Chinese word embeddings on Wikipedia corpus, using the skip-gram model in the word2vec toolkit with the default settings (Mikolov et al. 2013).

	English(PTB)			Chinese(CTB)		
	train	dev	test	train	dev	test
Sents	39,831	1,346	2,416	18,072	352	348
Toks	2,430k	84k	145k	1,611k	22k	25k

Table 1: The Corpus statistics of Penn English Treebanks (PTB) and Chinese Treebanks (CTB).

Settings

The part-of-speech tags for both English and Chinese datasets are obtained by using the Stanford tagger (Toutanvoa and Manning 2000) with 10-way jackknifing. In both English and Chinese tasks, the words with frequency less than 9 are replaced by UNK, leading to an English vocabulary sizes of 6870 and a Chinese vocabulary of 4967.

The training objective is optimized by a stochastic gradient descent algorithm on the training data and its optimized epoch is observed on the development data. In the SGD algorithm with a learning rate schema of an exponential decay. The parameters θ of our models are randomly initialized from the uniform distribution as suggested by Glorot and Bengio (2010). The overall hyper-parameters are shown in Table 2 without further tuned in our experiments.

As there are randomnesses in our training algorithm, we independently run the algorithm for five times and the final results was reported as the averaged results of these runs. The ensemble model includes these five independent models as its individual model.

[1]Compared with testing stage, training efficiency is the bottleneck for neural models in practice, because training usually takes several days.

Parameter	Value
LSTM Layers	1
Hidden unit dim	512
Trained word embedding dim	256
English pretrained word embedding dim	100
Chinese pretrained word embedding dim	200
POS tag embedding dim	128
Dropout rate	0.3
English batch size	10
Chinese batch size	5
Beam size in search	10

Table 2: Hyper-parameters used in our models.

Results on PTB

Model	F_1
DA_1	90.2
DA_2	90.2
DA_3	90.3

Table 3: F_1 score of different context functions on the PTB development data. DA_i denotes the model with regarding to the i_{th} context function in Section §3.

In Table 3, we compare the proposed modes based on different context functions ϕ as presented in Section §3. From this table, we can see that there are no significant differences among these three models. Therefore, in the rest of this paper, we only report the results of DA_3 for simplicity. Note that DA_1 includes more information than the other two models, since its context function ϕ defines on top of both the phrase $\mathcal{P}(b_t, s_t)$ and the semi-phrase $\mathcal{P}(s_t, ?)$. However, this does not lead to more gains accordingly. One possible reason is that the simple linear layer in Eq.(4) can not make full use of these additional information; and another reason may be that the phrase representation method is not strong enough to make them differentiable.

Figure 3 shows the F_1 score and model score according to different beam sizes on the development dataset. From this figure, we can see that our parser achieves decent F_1 score of 89.8 even with the greedy decoding, i.e. beam size 1. As the beam size increases, there are modest improvements but the accuracy peaks at the beam size of 80, achieving F_1 score of 90.3; and this case is similar for model score. In addition, we found that our accuracy almost stays at the same level when using a large beam size. This result is very different from the findings of NMT system on machine translation task, where the translation accuracy decreases largely with a large beam size as reported in Tu et al. (2017b) and Koehn and Knowles (2017). This may be caused by our deterministic attention in parsing, which is unsuitable to MT except the probabilistic attention. To trade-off the parsing accuracy and efficiency, in the rest of this experiments, the beam size is set to be 10.

Table 4 shows the comparison of the proposed model over other sequence-to-sequence parsing models, under the single model setting. Our model achieves significant gains over

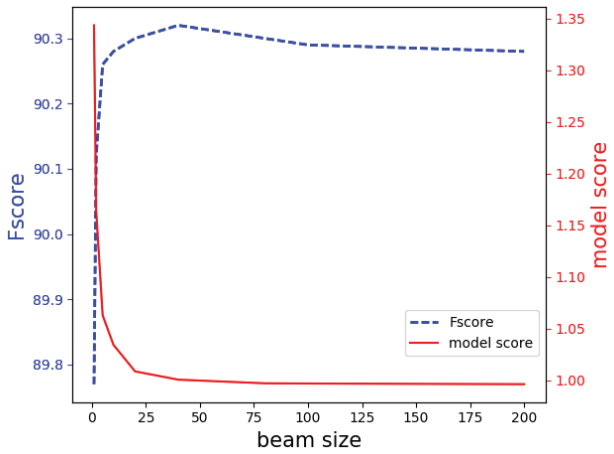


Figure 3: F_1 score (dashed curve with the left y axis) and model score (solid curve with the right y axis) with respect to different beam sizes during decoding on the development set.

Model	F_1
Vinyals et al. (2015) (single)	88.3
NMT-top-down (single)	87.2
Ma et al. (2017) (single)	88.6
Liu and Jiang (2017a)	90.5
This paper (single)	91.2

Table 4: Comparison between our model and other sequence-to-sequence parsers on English PTB test dataset.

the models with probabilistic attention on the top-down linearization, i.e., Vinyals et al. (2015) and NMT baselines. Similarly, compared with Ma et al. (2017), which is with the deterministic attention on the bottom-up linearization, our model obtains substantial improvements as well. In addition, compared with the fine-grained probabilistic attention model (Liu and Zhang 2017a), our model still obtains a gain of 0.7 F_1 score. Note that both (Liu and Zhang 2017a) and our model include the POS tag information and external word embeddings, while the other three models do not involve these additional information and thus one might argue that their comparison is unfair.

Model	F_1
Vinyals et al. (2015)	88.3
Ma et al. (2017)	88.6
This paper (Ablated)	90.1

Table 5: The F_1 of a single seq2seq parsing models on English PTB test data without POS tags and External embeddings.

To make an apple-to-apple comparison, we omit the POS tags and external word embedding from our model, and compare it with these sequence-to-sequnce models again. As

Model	F_1
Vinyals et al. (2015) (ensemble)	90.5
NMT-top-down (ensemble)	89.8
Ma et al. (2017) (ensemble)	90.6
This paper (ensemble)	92.3
This paper (single)	91.2

Table 6: Comparison with direct sequence-to-sequence parsers on English PTB Section 23 dataset.

shown in Table 5, without the pretrained word embedding and pos tags, the accuracy of our model drops about 1 point from 91.2 to 90.1. Fortunately, our model still obtains substantial improvements over the baselines. Our model outperforms Vinyals et al. (2015) with 1.8 F_1 score, which demonstrates that deterministic attention is better than probabilistic attention in constituent parsing. In addition, Our model achieves improvements of 1.5 F_1 score over Ma et al. (2017). This fact shows that top-down linearization is better than bottom-up linearization under the settings of sequence-to-sequence parsing.

Note that Vinyals et al. (2015) tried to introduce POS tags in the decoding phase, by replacing "XX" with its corresponding POS tag, but no performance improvement was observed. In this paper, we introduce the POS tags in the encoding phase and found they are useful according to our experiments. This finding is consistent with that in state-of-the-art neural parsers (Zhu et al. 2013; Liu and Zhang 2017b; Dyer et al. 2016; Stern, Andreas, and Klein 2017), all of which involve the POS tag information. Furthermore, to the best of our knowledge, it is the first time that a single model without POS tags and any other external resources achieves competitive accuracy on the PTB dataset.

Ensemble is an effective technique for recurrent neural networks,and thus we implement an ensemble of five models for further improvements. For comparison, we report the ensemble results of Vinyals et al. (2015), NMT-top-down, and Ma et al. (2017). Note that the first two baselines include five individual models as ours, and the third baseline includes ten individual models. The results are depicted in Table 6. From this table, we can see that the ensemble boosts our accuracy from 91.2 to 92.3 F_1 score, and it still outperforms the ensemble models including at least the same number of individual models.

We also compare our models with various state-of-the-art parsers as shown in Table 7. Several observations can be obtained from the table. Firstly, our single model outperforms the fully-supervised feature-rich models with gains up to 0.8 F_1 score, and it can match the performance of several neural models such as Cross and Huang (2016). Secondly, our model is better than the other sequence-to-sequence baselines except Vaswani et al. (2017), which relies on a probabilistic attention but a different model architecture. It is possible and would be interesting to integrate our idea on their model architecture. Thirdly, our single model is competitive to the best fully-supervised neural models such as Dyer et al. (2016), Stern, Andreas, and Klein (2017) and Liu and Zhang (2017b) in parsing accuracy. Since our model is sequential and does not require the tree-structured networks as Dyer et al. (2016) and Stern, Andreas, and Klein (2017), its training is easier to parallelize and scale to the large training datasets. In addition, as the parser proposed by Liu and Zhang (2017b) is essentially a feature-rich model with an additional RNN feature, in this sense our model is promising to be integrated into a feature-rich model for further improvements. Finally, our ensemble model delivers a decent result compared to the reranking and semi-supervised models. On the other hand, our model is orthogonal to both of these models. For instance, our model can provide a better kbest candidates for reranking with these notable generative models in Dyer et al. (2016) and Choe and Charniak (2016).

Model	F_1
fully-supervised	
Petrov and Klein (2007)¶	90.1
Socher et al. (2013)‡	90.4
Zhu et al. (2013)¶	90.4
Liu and Zhang (2017a)†	90.5
Watanabe and Sumita (2015)†	90.7
Durrett and Klein (2015)†	91.1
Cross and Huang (2016)†	91.3
Dyer et al. (2016)†	91.7
Stern, Andreas, and Klein (2017)†	91.7
Liu and Zhang (2017b) ‡	91.7
Petrov (2010) (ensemble)¶	91.9
Vaswani et al. (2017)†	91.3
Vinyals et al. (2015) (ensemble)†	90.5
Ma et al. (2017) (ensemble)†	90.7
This paper (ensemble)†	92.3
This paper (single)†	91.2
reranking	
Charniak and Johnson (2005)¶	91.5
Huang (2008)¶	91.7
Choe and Charniak (2016)†	92.6
Dyer et al. (2016)†	93.3
Kuncoro et al. (2017)†	93.6
semi-supervised	
Wang, Mi, and Xue (2015)†	90.7
Zhu et al. (2013)¶	91.3
Vinyals et al. (2015)†	92.8
Choe and Charniak (2016)†	93.8

Table 7: Comparison with state-of-the-art parsers on English PTB test data.¶, † and ‡ denote feature-rich models, neural models and hybrid feature-rich and neural models.

Results on CTB

We additionally train parsers on the CTB and testify its accuracy on the test data. We list our final results and compare them to several other papers in Table 8. Our single model is competitive to Watanabe and Sumita (2015). Although it falls short of the score achieved by the structured neural networks in Dyer et al. (2016) and the hybrid model in Liu and Zhang (2017b), such gap can be narrowed by a simple en-

Model	F_1
fully-supervised	
Petrov and Klein (2007)¶	83.3
Zhu et al. (2013)¶	83.2
Wang et al. (2015)‡	83.2
Watanabe and Sumita (2015)†	84.3
Dyer et al. (2016)†	84.6
Liu and Jiang (2017b)‡	85.5
This paper (single)†	84.1
This paper (ensemble)†	85.4
reranking	
Charniak and Johnson (2005)¶	82.3
Dyer et al. (2016)†	86.9
semi-supervised	
Zhu et al. (2013)¶	85.6
Wang and Xue (2014)¶	86.3
Wang et al. (2015)‡	86.6

Table 8: Comparison with state-of-the-art parsers on CTB test data. ¶, † and ‡ denote feature-rich models, neural models and hybrid feature-rich and neural models.

semble technique.

Errors on PTB

error type	bottom-up	top-down	reduction
PP Attach	850	668	182 (21%)
1-word Span	687	489	198 (29%)
Unary	555	412	143 (25%)
NP Int	464	335	129 (28%)
Clause Att	376	311	65 (17%)
Different label	374	269	105 (28%)
Mod Attach	317	264	53 (17%)
Co-ordination	379	225	154 (41%)
UNSET add	291	216	75 (26%)
NP Attachment	216	161	55 (25%)
Other	413	304	109 (26%)

Table 9: Error types between bottom-up and top-down deterministic attentions on PTB.

We used the toolkit in Kummerfeld et al. (2012) to analyze the parsing errors on different error types.[2] Table 9 lists the error results between a single model in Ma et al. (2017) and a single of our model with POS tags and external word embeddings. Generally, both sequence-to-sequence models suffer most errors from the types of PP attachment, 1-word span and unary. Compared with the parser in Ma et al. (2017), our improved parser can largely reduce these errors, and the Coordination errors are particularly reduced by about 40%.

[2]https://code.google.com/p/berkeley-parser-analyser/

Related Work

Constitency parsing is a traditional NLP task and there have been many attempts to model this task. Over the decades, feature-rich linear models have been widely employed in constituency parsing. For example, Petrov and Klein (2007) proposed a log-linear model with hand-crafted features based on the CKY parsing agenda. Zhu et al. (2013) and Wang and Xue (2014) integrated more abundant features into the shift-reduce parsing agenda. These feature-rich models are efficient for training, but they are unable to make the full use of the long-distance features due to feature sparsity.

Neural models are appealing to avoid the feature sparsity by mapping discrete features into dense vectors and provide a natural way to make use the large context for parsing action predictions (Vinyals et al. 2015; Watanabe and Sumita 2015; Cross and Huang 2016; Dyer et al. 2016). For instance, Dyer et al. (2016) proposed a stack-LSTM to represent the history action predictions into a tree structure; Stern, Andreas, and Klein (2017) proposed a variant neural network based on the CKY structure; and both of the notable models achieved the new state of the art parsing accuracy so far. Unlike these explicit tree-structured models, there are some sequential neural models by casting constituency parsing as an instance of sequence-to-sequence learning (Vinyals et al. 2015; Liu and Zhang 2017b; Vaswani et al. 2017). Similar to these models, our parsing model is essentially a sequential models; whereas our model is based on the deterministic attention as Ma et al. (2017) rather than the probabilistic attention in their models.

Conclusion and Future Work

This paper proposed an improved sequence-to-sequence constituency parser. Its key technique is a deterministic attention mechanism, which is able to select informative context from the input side at each timestep along the top-down linearized sequence of a parse tree. Empirical experimental results show that our parser achieves substantial improvements over many sequence-to-sequence baseline parsers, and obtains 92.3 Fscore on the Penn English Treebank section 23 dataset and 85.4 Fscore on the Chinese Treebank test dataset, without reranking and semi-supervised training.

In the future, as we utilized a simple method to represent a phrase and a semi-phrase, it is promising to investigate more sophisticated methods for their representations in our deterministic attention mechanism. In addition, since our parser is trained in a sequential manner, it would be interesting to train it on a large mount of corpus by tri-training, which is potential to further boost the parsing accuracy (Vinyals et al. 2015; Choe and Charniak 2016).

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473.

Charniak, E., and Johnson, M. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, 173–180.

Choe, D. K., and Charniak, E. 2016. Parsing as language modeling. In *Proceedings of EMNLP*, 2331–2336.

Cross, J., and Huang, L. 2016. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *Proceedings of EMNLP*, 1–11.

Durrett, G., and Klein, D. 2015. Neural crf parsing. In *Proceedings of ACL-IJCNLP*, 302–312.

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 ACL-IJCNLP*, 334–343.

Dyer, C.; Kuncoro, A.; Ballesteros, M.; and Smith, N. A. 2016. Recurrent neural network grammars. In *Proceedings of NAACL*, 199–209.

Galley, M.; Hopkins, M.; Knight, K.; and Marcu, D. 2004. What's in a translation rule? In *Proceedings of NAACL-HLT*.

Galley, M.; Graehl, J.; Knight, K.; Marcu, D.; DeNeefe, S.; Wang, W.; and Thayer, I. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of COLING-ACL*, 961–968.

Glorot, X., and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 249–256.

Huang, L. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-HLT*, 586–594.

Koehn, P., and Knowles, R. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, 28–39.

Kummerfeld, J. K.; Hall, D.; Curran, J. R.; and Klein, D. 2012. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *Proceedings of EMNLP-CoNLL*, 1048–1059.

Kuncoro, A.; Ballesteros, M.; Kong, L.; Dyer, C.; Neubig, G.; and Smith, N. A. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of EACL*, 1249–1258.

Liu, J., and Zhang, Y. 2017a. Encoder-decoder shift-reduce syntactic parsing. In *Proceedings of IWPT*.

Liu, J., and Zhang, Y. 2017b. Shift-reduce constituent parsing with neural lookahead features. *TACL* 5:45–58.

Ma, C.; Liu, L.; Tamura, A.; Zhao, T.; and Sumita, E. 2017. Deterministic attention for sequence-to-sequence constituent parsing. In *Proceedings of the AAAI*, 3237–3243.

Manning, C. D.; Schütze, H.; et al. 1999. *Foundations of statistical natural language processing*, volume 999. MIT Press.

Marcus, M. P.; Marcinkiewicz, M. A.; and Santorini, B. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.* 19(2):313–330.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Petrov, S., and Klein, D. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL-HLT*.

Petrov, S. 2010. Products of random latent variable grammars. In *Proceedings of NAACL-HLT*, 19–27.

Rim, H.-C.; Seo, J.; and Simmons, R. F. 1990. Transforming syntactic graphs into semantic graphs. In *Proceedings of ACL*, 47–53.

Roark, B., and Johnson, M. 1999. Efficient probabilistic top-down and left-corner parsing. In *Proceedings of ACL*, 421–428.

Socher, R.; Bauer, J.; Manning, C. D.; and Andrew Y., N. 2013. Parsing with compositional vector grammars. In *Proceedings of ACL*, 455–465.

Stern, M.; Andreas, J.; and Klein, D. 2017. A minimal span-based neural constituency parser. In *Proceedings of ACL*, 818–827.

Toutanvoa, K., and Manning, C. D. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 63–70.

Tu, Z.; Liu, Y.; Lu, Z.; Liu, X.; and Li, H. 2017a. Context gates for neural machine translation. *TACL* 5:87–99.

Tu, Z.; Liu, Y.; Shang, L.; Liu, X.; and Li, H. 2017b. Neural machine translation with reconstruction. In *Proceedings of the AAAI*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. *CoRR*.

Vinyals, O.; Kaiser, Ł.; Koo, T.; Petrov, S.; Sutskever, I.; and Hinton, G. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, 2773–2781.

Wang, Z., and Xue, N. 2014. Joint pos tagging and transition-based constituent parsing in chinese with non-local features. In *Proceedings of ACL*, 733–742.

Wang, Z.; Mi, H.; and Xue, N. 2015. Feature optimization for constituent parsing via neural networks. In *Proceedings of ACL*, 1138–1147.

Watanabe, T., and Sumita, E. 2015. Transition-based neural constituent parsing. In *Proceedings of the ACL*, 1169–1179.

Xue, N.; Xia, F.; Chiou, F.-d.; and Palmer, M. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Nat. Lang. Eng.* 11(2):207–238.

Zhu, M.; Zhang, Y.; Chen, W.; Zhang, M.; and Zhu, J. 2013. Fast and accurate shift-reduce constituent parsing. In *Proceedings of ACL*, 434–443.