Transition-based Neural RST Parsing with Implicit Syntax Features

Nan Yu, **Meishan Zhang** and **Guohong Fu***
School of Computer Science and Technology, Heilongjiang University, China
yunan.hlju@gmail.com,
mason.zms@gmail.com,
ghfu@hotmail.com

Abstract

Syntax has been a useful source of information for statistical RST discourse parsing. Under the neural setting, a common approach integrates syntax by a recursive neural network (RNN), requiring discrete output trees produced by a supervised syntax parser. In this paper, we propose an implicit syntax feature extraction approach, using hidden-layer vectors extracted from a neural syntax parser. In addition, we propose a simple transition-based model as the baseline, further enhancing it with dynamic oracle. Experiments on the standard dataset show that our baseline model with dynamic oracle is highly competitive. When implicit syntax features are integrated, we are able to obtain further improvements, better than using explicit Tree-RNN.

1 Introduction

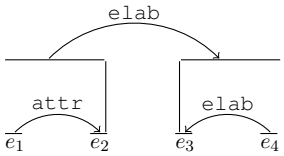
Discourse parsing is an important task in natural language processing (NLP), which aims to identify the relations between text units in documents. It has received great attention (Sagae, 2009; Hernault et al., 2010; Feng and Hirst, 2012; Joty et al., 2013; Feng and Hirst, 2014; Ji and Eisenstein, 2014; Heilman and Sagae, 2015; Li et al., 2016; Wang et al., 2017; Braud et al., 2017; Liu and Lapata, 2017), especially by performing the task based on rhetorical structure theory (RST) (Mann and Thompson, 1988). The RST-based parsing represents a document by a hierarchical tree, where leaf nodes are basic text units referred as elementary discourse unit (EDU), and non-terminal nodes define the discourse relations between adjacent tree nodes. Figure 1 shows an example, where each discourse relation has two parts, specifying its label and indicating its nuclearity, respectively.

Early studies adopt traditional statistical models for this task, using sophisticated manually-designed discrete features (Soricut and Marcu, 2003; Sagae, 2009; Hernault et al., 2010; Feng and Hirst, 2012; Joty et al., 2013; Feng and Hirst, 2014; Heilman and Sagae, 2015; Wang et al., 2017). Recently, inspired by the success of neural network models in NLP (Collobert et al., 2011; Devlin et al., 2014), several neural models for RST discourse parsing have been proposed as well (Li et al., 2014; Li et al., 2015b; Li et al., 2016; Braud et al., 2016; Braud et al., 2017; Liu and Lapata, 2017). Compared with statistical models, neural models exploit low-dimensional dense features, being able to avoid the feature sparsity problem, and on the other hand well-designed neural network structures such as long short term memory (LSTM) (Schmidhuber and Hochreiter, 1997) are capable of capturing high-order compositional features as well as global features automatically. In addition, we can use pre-trained neural word embeddings (Mikolov et al., 2013) on large scale corpus for neural network initialization. These characteristics show that neural models are promising for RST discourse parsing.

Intuitively, syntax is a potential avenue for the task, as it offers long-distance syntax relations between sentential words as well as key components of sentences. Syntax features have been demonstrated helpful in statistical models with human designed features (Soricut and Marcu, 2003; Sagae, 2009; Hernault et al., 2010; Feng and Hirst, 2012; Joty et al., 2013; Feng and Hirst, 2014; Heilman and Sagae, 2015). For neural network models, recursive neural network is a natural choice to represent tree-structural syntax trees globally. Only Li et al. (2015b) apply such a neural structure to incorporate syntax trees for

*Corresponding author.

Proceedings of the 27th International Conference on Computational Linguistics, pages 559–570
Santa Fe, New Mexico, USA, August 20-26, 2018.



e_1: American Telephone & Telegraph Co. said it
e_2: will lay off 75 to 85 technicians here , effective Nov. 1.
e_3: The workers install , maintain and repair its private branch exchanges,
e_4: which are large intracompany telephone networks.

Figure 1: An example of RST discourse tree, where $\{e_1, e_2, e_3, e_4\}$ are EDUs, `attr` and `elab` are discourse relation labels, and arrows indicate the nuclearities of discourse relations.

RST discourse parsing. Other studies still adopt discrete syntax features proposed by statistical models, feeding them into neural network models (Braud et al., 2016; Braud et al., 2017).

The above approaches model syntax trees in an explicit way, requiring discrete syntax parsing outputs as inputs for RST parsing. These approaches may suffer from the error propagation problem. Syntax trees produced by a supervised syntax parsing model could have errors, which may propagate into discourse parsing models. The problem could be extremely serious when inputs of discourse parsing have different distributions with the training data of the supervised syntax parser. Recently, Zhang et al. (2017) suggest an alternative method, which extracts syntax features from a Bi-Affine dependency parser (Dozat and Manning, 2016), and the method gives competitive performances on relation extraction. It actually represents syntax trees implicitly, thus it can reduce the error propagation problem.

In this work, we investigate the implicit syntax feature extraction approach for RST parsing. In addition, we propose a transition-based neural model for this task, which is able to incorporate various features flexibly. We exploit hierarchical bi-directional LSTMs (Bi-LSTMs) to encode texts, and further enhance the transition-based model with dynamic oracle. Based on the proposed model, we study the effectiveness of our proposed implicit syntax features. We conduct experiments on a standard RST discourse TreeBank (Carlson et al., 2003). First, we evaluate the performance of our proposed transition-based baseline, finding that the model is able to achieve strong performances after applying dynamic oracle. Then we evaluate the effectiveness of implicit syntax features extracted from a Bi-Affine dependency parser. Results show that the implicit syntax features are effective, giving better performances than explicit Tree-LSTM (Li et al., 2015b). Our codes will be released for public under the Apache License 2.0 at `https://github.com/yunan4nlp/NNDisParser`.

In summary, we mainly make the following two contributions in this work: (1) we propose a transition-based neural RST discourse parsing model with dynamic oracle, (2) we compare three different syntactic integration approaches proposed by us. The rest of the paper is organized as follows. Section 2 describes our proposed models including the transition-based neural model, the dynamic oracle strategy and the implicit syntax feature extraction approach. Section 3 presents the experiments to evaluate our models. Section 4 shows the related work. Finally, section 5 draws conclusions.

2 Transition-based Discourse Parsing

We follow Ji and Eisenstein (2014), exploiting a transition-based framework for RST discourse parsing. The framework is conceptually simple and flexible to support arbitrary features, which has been widely used in a number of NLP tasks (Zhu et al., 2013; Dyer et al., 2015; Zhang et al., 2016). In addition, a transition-based model formalizes a certain task into predicting a sequence of actions, which is essential similar to sequence-to-sequence models proposed recently (Bahdanau et al., 2014). In the following, we first describe the transition system for RST discourse parsing, and then introduce our neural network model by its encoder and decoder parts, respectively. Thirdly, we present our proposed dynamic oracle strategy aiming to enhance the transition-based model. Then we introduce the integration method of implicit syntax features. Finally we describe the training method of our neural network models.

2.1 The Transition-based System

The transition-based framework converts a structural learning problem into a sequence of action predictions, whose key point is a transition system. A transition system consists of two parts: states and actions. The states are used to store partially-parsed results and the actions are used to control state transitions.

Step	Stack	Queue	Action	Relation
1	$\varnothing$	e_1, e_2, e_3, e_4	SH	$\varnothing$
2	e_1	e_2, e_3, e_4	SH	$\varnothing$
3	e_1, e_2	e_3, e_4	RD (attr, SN)	$\varnothing$
4	$e_{1:2}$	e_3, e_4	SH	$\widehat{e_1\mathbf{e_2}}$
5	$e_{1:2}, e_3$	e_4	SH	$\widehat{e_1\mathbf{e_2}}$
6	$e_{1:2}, e_3, e_4$	$\varnothing$	RD (elab, NS)	$\widehat{e_1\mathbf{e_2}}$
7	$e_{1:2}, e_{3:4}$	$\varnothing$	RD (elab, SN)	$\widehat{e_1\mathbf{e_2}}, \widehat{\mathbf{e_3}e_4}$
8	$e_{1:4}$	$\varnothing$	PR	$\widehat{e_1\mathbf{e_2}}, \widehat{\mathbf{e_3}e_4}, \widehat{e_{1:2}\mathbf{e_{3:4}}}$

Table 1: An example of the transition-based system for RST discourse parsing.

The initial state is an empty state, and the final state represents a full result. There are three kinds of actions in our transition system:

- **Shift** (SH), which removes the first EDU in the queue onto the stack, forming a single-node subtree.
- **Reduce** (RD) (l,d), which merges the top two subtrees on the stack, where l is a discourse relation label, and $d \in \{\texttt{NN}, \texttt{NS}, \texttt{SN}\}$ indicates the relation nuclearity (nuclear (N) or satellite (S)).
- **Pop Root** (PR), which pops out the top tree on the stack, marking the decoding being completed, when the stack holds only one subtree and the queue is empty.

Given the RST tree as shown in Figure 1, it can be generated by the following action sequence: {SH, SH, RD (attr, SN), SH, SH, RD (elab, NS), RD (elab, SN), PR}. Table 1 shows the decoding process in detail. By this way, we naturally convert RST discourse parsing into predicting a sequence of transition actions, where each line includes a state and next step action referring to the tree.

2.2 Encoder-Decoder

Previous transition-based RST discourse parsing studies exploit statistical models, using manually-designed discrete features (Sagae, 2009; Heilman and Sagae, 2015; Wang et al., 2017). In this work, we propose a transition-based neural model for RST discourse parsing, which follows an encoder-decoder framework. Given an input sequence of EDUs $\{e_1, e_2, ..., e_n\}$, the encoder computes the input representations $\{\mathbf{h}_1^e, \mathbf{h}_2^e, ..., \mathbf{h}_n^e\}$, and the decoder predicts next step actions conditioned on the encoder outputs.

2.2.1 Encoder

We follow Li et al. (2016), using hierarchical Bi-LSTMs to encode the source EDU inputs, where the first-layer is used to represent sequencial words inside of EDUs, and the second layer is used to represent sequencial EDUs. Given an input sentence $\{w_1, w_2, ..., w_m\}$, first we represent each word by its form (e.g., w_i) and POS tag (e.g. t_i), concatenating their neural embeddings. By this way, the input vectors of the first-layer Bi-LSTM are $\{\mathbf{x}_1^w, \mathbf{x}_2^w, ..., \mathbf{x}_m^w\}$, where $\mathbf{x}_i^w = emb(w_i) \oplus emb(t_i)$, and then we apply Bi-LSTM directly, obtaining:

$$\{\mathbf{h}_1^w, \mathbf{h}_2^w, ..., \mathbf{h}_m^w\} = \text{Bi-LSTM}(\{\mathbf{x}_1^w, \mathbf{x}_2^w, ..., \mathbf{x}_m^w\}) \quad (1)$$

The second-layer Bi-LSTM is built over sequential EDUs. We should first obtain a suitable representation for each EDU, which is composed by a span of words inside a certain sentence. Assuming an EDU with its words by $\{w_s, w_{s+1}, ..., w_t\}$, after applying the first-layer Bi-LSTM, we obtain their representations by $\{\mathbf{h}_s^w, \mathbf{h}_{s+1}^w ..., \mathbf{h}_t^w\}$, then we calculate the EDU representation by average pooling:

$$\mathbf{x}^e = \frac{1}{t-s+1} \sum_{s}^{t} \mathbf{h}_k^w \quad (2)$$

When the EDU representations are ready, we apply the second-layer Bi-LSTM directly, resulting:

$$\{\mathbf{h}_1^e, \mathbf{h}_2^e, ..., \mathbf{h}_n^e\} = \text{Bi-LSTM}(\{\mathbf{x}_1^e, \mathbf{x}_2^e, ..., \mathbf{x}_n^e\}) \quad (3)$$

Algorithm 1 Dynamic Oracle.

$S \leftarrow empty$;
$a_g \leftarrow Oracle(S, T)$;
$a_p \leftarrow Predict(S)$;
while S is not $terminal$ **do**
    $Train(S, a_g)$;
    if $pick\_gold > \alpha$ **then**
        $S \leftarrow Apply(S, a_g)$;
    else
        $S \leftarrow Apply(S, a_p)$;

2.2.2 Decoder

The decoder part is to predict the next-step action based on a given state. As mentioned before, each transition state has a sequence of subtrees on the stack and a sequence of unprocessed EDUs in the queue. We simply choose the top three stack subtrees (s_0, s_1, s_2) and the first EDU (q_0) in the queue as major clues for prediction, applying a feed-forward neural layer to calculate next-step action scores:

$$\mathbf{o} = \mathbf{W}(\mathbf{h}^{sbt}_{s_0} \oplus \mathbf{h}^{sbt}_{s_1} \oplus \mathbf{h}^{sbt}_{s_2} \oplus \mathbf{h}^{e}_{q_0}) \tag{4}$$

where $\mathbf{o}$ is the output scores and $\mathbf{W}$ is a model parameter.

The hidden vector $\mathbf{h}^{e}_{q_0}$ is obtained directly from the encoder outputs. The hidden vectors $\mathbf{h}^{sbt}_{s_0}$, $\mathbf{h}^{sbt}_{s_1}$ and $\mathbf{h}^{sbt}_{s_2}$ are representations of partially-parsed subtrees. For efficiency, we exploit average pooling over the covering EDUs $\{e_i, e_{i+1}, ..., e_j\}$ of a subtree s to derive its representation:

$$\mathbf{h}^{sbt}_{s} = \frac{1}{j-i+1} \sum_{i}^{j} \mathbf{h}^{e}_{k} \tag{5}$$

2.3 Dynamic Oracle

The transition-based model converts discourse parsing into an incremental action prediction problem. For a given state, we require its answer for training. When the state follows the traces of gold-standard actions, its answer is exactly the next gold-standard action. Majority work trains a classifier based on gold-standard state-answer pairs (Ji and Eisenstein, 2014; Heilman and Sagae, 2015; Braud et al., 2017; Wang et al., 2017). However, we usually have error states during the test stage, which are resulted by historical error actions. These error states have never been seen during training, which makes prediction difficult for them. The phenomenon brings inconsistency between training and testing.

Here we adopt dynamic oracle to alleviate the above problem, which has been firstly exploited by Goldberg and Nivre (2012) for dependency parsing. The main idea is to add error states and their oracle actions into the training corpus randomly. On the one hand, by exploring a percentage of error states, the transition-based model is able to handle these states more confidently. On the other hand, by defining oracle actions over error states, the model is able to make best choices even when errors occur.

In this work, we suggest a similar dynamic oracle strategy for our transition-based neural model. Algorithm 1 shows the pseudo codes. During training, we explore into predicted states randomly by a probability α. At each step, we generate a random number $pick\_gold$ between [0,1]. If the value is less than α, we use the predicted state as the next-step input, making the training process consistent with the testing. The oracle function is defined by minimizing the future decoding loss referring to gold-standard RST trees, which is defined as follows:

$$a_g = \begin{cases} \text{RD}, \ s.t. \ \widehat{s_0 s_1} \text{ is a subtree in G (relation labels and nuclearities are neglected)} \\ \text{SH}, \ \text{otherwise,} \end{cases} \tag{6}$$

where s_0, s_1 are top two stack subtrees, and G is the set of gold-standard subtrees. Simply speaking, if s_0 and s_1 can be composed as a subtree according to the gold-standard RST-tree, we apply RD, and otherwise we apply SH.

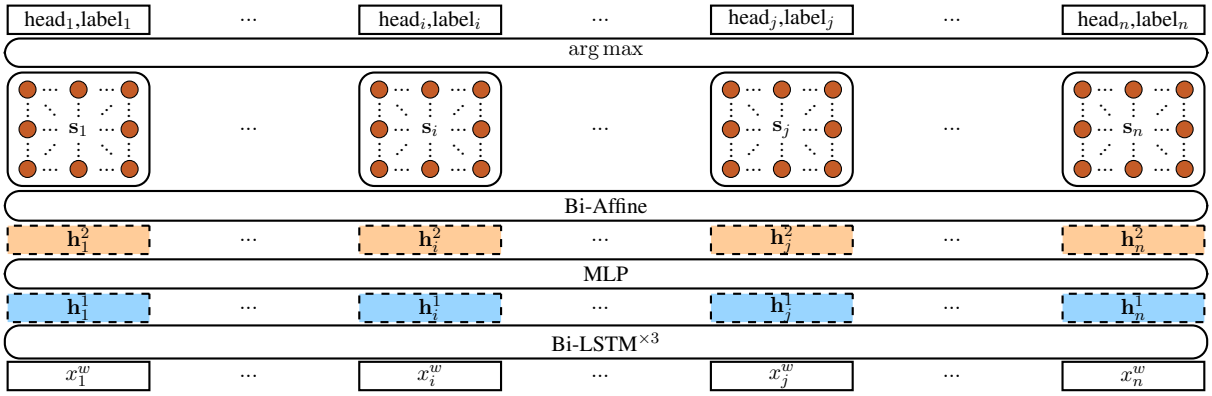


Figure 2: The framework of Bi-Affine neural parser.

2.4 Syntax Features

Intuitively, syntax information could be a valuable source for RST parsing (Soricut and Marcu, 2003; Sagae, 2009; Hernault et al., 2010; Feng and Hirst, 2012; Joty et al., 2013; Feng and Hirst, 2014; Heilman and Sagae, 2015). Previously, the information is integrated into RST discourse parsing by manually-designed discrete features, except the work of Li et al. (2015b), which exploits a bi-directional Tree-LSTM to model syntax trees.

In addition, all previous studies use deterministic parsing outputs as inputs for RST discourse parsing, which may suffer from error propagation problem since syntax parsing outputs can have errors. To alleviate this problem, Zhang et al. (2017) suggest an alternative method, which extracts intermediate hidden outputs of a neural parsing model as inputs for relation-extraction. The method exploits an implicit way to incorporate syntax features without the the need of parsing outputs. Inspired by their work, we investigate the same method for RST discourse parsing.

We follow Zhang et al. (2017) to use the Bi-Affine dependency neural parser of Dozat and Manning (2016) for syntax feature extraction. Zhang et al. (2017) formalize the Bi-Affine parser as an encoder-decoder architecture, and exploit the encoder outputs as inputs for relation extraction. Here we study the Bi-Affine parser in deep, examining other hidden layers as well. Next we will introduce the Bi-Affine parser briefly, and then describe our implicit syntax feature extraction method.

2.4.1 The Bi-Affine Parser

The Bi-Affine neural dependency parser has achieved the top performances (Dozat and Manning, 2016). As shown by Figure 2, the model has four components: (1) a three-layer Bi-LSTM (Bi-LSTM$^{\times 3}$) over the input sentential words; (2) a multi-layer perceptron (MLP) layer; (3) a Bi-Affine layer; (4) an $\arg\max$ decoder. Their outputs are $\{\mathbf{h}_1^1, \mathbf{h}_2^1 ..., \mathbf{h}_n^1\}$, $\{\mathbf{h}_1^2, \mathbf{h}_2^2, ..., \mathbf{h}_n^2\}$, $\{\mathbf{s}_{ij}^l, i \in [1,n], j \in [1,n], l \in L$ (The set of dependency labels.)$\}$, and $\{(\text{head}_1, \text{label}_1), ..., (\text{head}_n, \text{label}_n)\}$, respectively, which are computed by follow formulas:

$$
\begin{aligned}
&\{\mathbf{h}_1^1, \mathbf{h}_2^1, ..., \mathbf{h}_n^1\} = \text{Bi-LSTM}^{\times 3}(\{\mathbf{x}_1^w, \mathbf{x}_2^w, ..., \mathbf{x}_n^w\}) \\
&\{\mathbf{h}_1^2, \mathbf{h}_2^2, ..., \mathbf{h}_n^2\} = \text{MLP}(\{\mathbf{h}_1^1, \mathbf{h}_2^1, ..., \mathbf{h}_n^1\}) \\
&\mathbf{s}_{i,j}^l = \text{Bi-Affine}(\mathbf{h}_i^2, \mathbf{h}_j^2, l) \\
&\text{head}_i, \text{label}_i = argmax_{\{h,l\}}(\mathbf{s}_{i,h}^l)
\end{aligned}
\tag{7}
$$

Noticeably, the above formulas are only an approximate description of the Bi-Affine parser. For more details, one can refer to their paper.

2.4.2 Implicit Syntax Feature Extraction

Zhang et al. (2017) formalize the Bi-Affine parser as an encoder-decoder model, where the encoder part includes the neural structures ending by the Bi-LSTM$^{\times 3}$, and they use the encoder outputs for implicit syntax feature extraction. Additionally, by carefully examining, we find that the MLP outputs have

similar attributes to the Bi-LSTM$^{\times 3}$ outputs, both of which are aligned with the input sentential words perfectly, being closely related to the words at the aligned positions. In addition, the MLP outputs are closer to the final syntax parsing outputs, thus are able to encode syntax information more sufficiently.

Concretely, for a given sentence $\{w_1, w_2, ..., w_n\}$, we feed it into the Bi-Affine parser, extracting the encoder outputs $\{\mathbf{h}_1^1, \mathbf{h}_2^1, ..., \mathbf{h}_n^1\}$ or the MLP outputs $\{\mathbf{h}_1^2, \mathbf{h}_2^2, ..., \mathbf{h}_n^2\}$, respectively. Then we concatenate them with the encoder outputs of our transition-based neural RST parsing model. The resulting vectors are used as inputs for the decoder part. The overall syntax incorporation process can be formalized as follows:

$$\{\mathbf{h}'_1, \mathbf{h}'_2, ..., \mathbf{h}'_n\} = \{\mathbf{h}_1^w \oplus \mathbf{h}_1^*, \mathbf{h}_2^w \oplus \mathbf{h}_2^*, ..., \mathbf{h}_n^w \oplus \mathbf{h}_n^*\} \tag{8}$$

where $*$ could be either 1 or 2, denoting the Bi-LSTM$^{\times 3}$ or the MLP outputs.

2.5 Training

We follow the majority work of transition-based deterministic neural models, using a cross-entropy loss plus with an l_2 regularization as our training objective. Given a state S and its oracle action a, we calculate the decode outputs by using equation (4), and then apply a softmax operation to obtain the oracle action probalility: $p_{a_g} = \frac{exp(\mathbf{o}_{a_g})}{\sum_{a' \in A} exp(\mathbf{o}_{a'})}$, which is then fed into the cross-entropy function:

$$L(\Theta) = -log(p_{a_g}) + \frac{\lambda}{2}||\Theta||^2 \tag{9}$$

where Θ is the set of model parameter and λ is an l_2 regularization factor. We train our model by online learning with mini-batch, updating model parameters by using Adam algorithm (Kingma and Ba, 2014).

3 Experiments

3.1 Settings

3.1.1 Data

We follow previous studies (Ji and Eisenstein, 2014; Li et al., 2014; Feng and Hirst, 2014; Heilman and Sagae, 2015; Li et al., 2016; Wang et al., 2017; Braud et al., 2017), conducting experiments by using the RST discourse Treebank (Carlson et al., 2003). It has annotated 385 documents, where 347 for training and the remaining 38 for testing. All the documents are collected from Wall Street Journal (WSJ), belonging to newswire gene. We use 18 coarse-grained relations in our experiments. For preprocessing, we exploit the StanFord CoreNLP toolkit[1](Manning et al., 2014) to lemmatize words and get their POS tags. To facilitate parameter tuning, we select 35 documents from the training corpus randomly as the development corpus. All experiments are conducted on manually segmented EDUs.

3.1.2 Evaluation

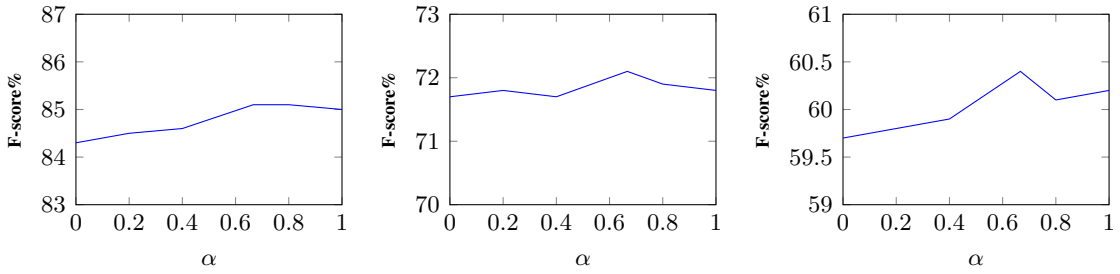
We adopt standard evaluation methods (Marcu, 2000) to test model performances, including three metrics `Span`, `Nuclearity` and `Relation`. The `Span` evaluates the capability of predicting tree skeletons, the `Nuclearity` evaluates tree skeletons and nuclearity indications, and the `Relation` evaluates tree skeletons together with discourse relations, while ignoring nuclearities. The `Full` evaluates the tree skeletons together with both nuclearity indications and discourse relations. We follow (Morey et al., 2017) to report the micro-averaged F scores.

3.1.3 Hyper-Parameters

There are several hyper-parameters in our proposed models. We set the values according to their development performances. We set sizes of all hidden vectors (e.g. the dimension of word embeddings, the Bi-LSTM outputs and etc.) by 200. The word embeddings are initialized by pre-trained GloVe embeddings[2](Pennington et al., 2014). For training, the l_2 regularization factor, the mini-batch size, the initial learning rate, the dropout probability and the max norm of gradient clipping are set by 10^{-6}, 8, 0.001,

[1]https://stanfordnlp.github.io/CoreNLP/

[2]http://nlp.stanford.edu/data/wordvecs/glove.6B.zip



(a) Span (b) Nuclearity (c) Relation

Figure 3: The influence of the dynamic oracle strategy in different α.

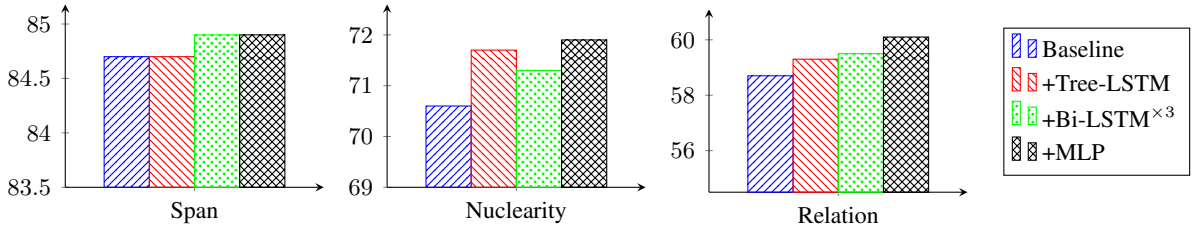


Figure 4: The influence of different syntax feature integration methods.

0.5 and 10, respectively. We train models iteratively on the entire training corpus by 60 rounds, and use the best-performance iteration on the development corpus as the final model.

3.2 Development Results

We conduct two sets of development experiments to show important factors of our proposed model, which are related to dynamic oracle and syntax features, respectively.

3.2.1 Dynamic Orcale

First, we examine the strategy of dynamic oracle. As shown in Algorithm 1, there is a hyper-parameter α to control the exploration, which is used to produce extra training instances. Here we study how α influences the model performances. Figure 3 shows the development performances with respect to α. The performances are relatively stable surrounding 0.7, where all metrics under both settings are close to their peak values. Thus we exploit $\alpha = 0.7$ as the final setting.

3.2.2 Syntax Features

There are two choices of our implicit syntax feature extraction approach. We can use either the encoder Bi-LSTM$^{\times 3}$ outputs or the MLP outputs as described in section 2.4. In addition, we implement a bi-directional dependency based Tree-LSTM (Teng and Zhang, 2017) as well for comparisons. Figure 4 shows the results. First, we find that syntax features[3] are very useful for RST discourse parsing, which is consistent with previous observations (Soricut and Marcu, 2003; Sagae, 2009; Braud et al., 2016). Second, the implicit syntax feature extraction method is slightly better than explicit Tree-LSTM. In particular, the model can achieve the best performances by using the MLP outputs. Thus we exploit the MLP outputs as the final implicit syntax features in our RST discourse parsing model.

[3] We use the English PTB corpus to train the Bi-Affine parser, which achieves 95.88% UAS and 94.16% LAS.

Model	Span	Nuclearity	Relation	Full
(Hayashi et al., 2016)	82.6	66.6	54.6	54.3
(Surdeanu et al., 2015)	82.6	67.1	55.4	54.9
(Joty et al., 2015)	82.6	68.3	55.8	55.4
(Feng and Hirst, 2014)	84.3	69.4	56.9	56.2
(Braud et al., 2016)	79.7	63.6	47.7	47.5
(Li et al., 2016)	82.2	66.5	51.4	50.6
(Braud et al., 2017)	81.3	68.1	56.3	56.0
(Ji and Eisenstein, 2014)	82.0	68.2	57.8	57.6
Baseline	85.0	71.0	57.6	57.1
Our Final Model	**85.5**	**73.1**	**60.2**	**59.9**
Human	88.3	77.3	65.4	64.7

Table 2: Final micro-averaged F scores on the test corpus. The results of other models are borrowed from Morey et al. (2017).

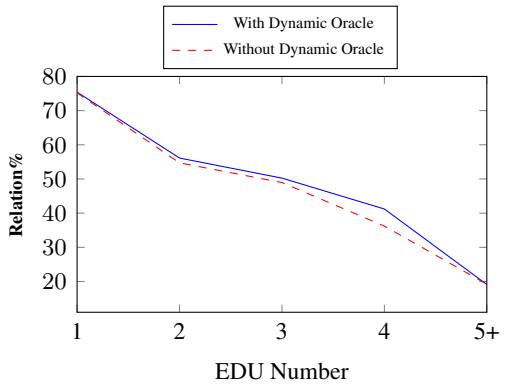


Figure 5: The span-level F-measures with respect to the number of EDU.

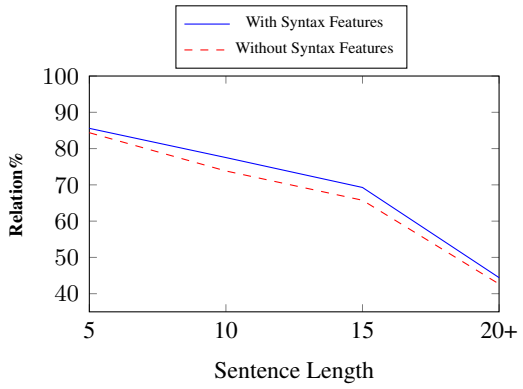


Figure 6: The span-level F-measures with respect to the span length.

3.3 Final Results

Table 2 shows the final results of our proposed neural models on the test corpus. Our baseline model achieves a `Span` F-measure of 85.0 and a `Nuclearity` F-measure of 71.0, which are exceed most models. When dynamic oracle and implicit syntax features are exploited, our final model achieves 60.2 on the `Relation` F-measure, resulting overall improvements 60.2 - 57.6 = 2.6. The `Span`, `Nucleartiy` and `Full` metrics have similar tendencies as well.

We compare the proposed neural model with other state-of-the-art systems as well. (Hayashi et al., 2016) is a greedy bottom-up parser with a linear SVM classification. (Surdeanu et al., 2015) is also a greedy bottom-up parser, which uses the perceptron for tree skeleton building and nuclearity indication, the logistic regression for relation labelling. (Joty et al., 2015) is a two-stage (intra-sentential then multi-sentential parsing) parser with dynamic conditional random field models, and (Feng and Hirst, 2014) is a two-stage parser with linear-chain conditional random field models. (Braud et al., 2016) is a sequence-to-sequence hierarchical neural parser, and (Li et al., 2016) is an attention-based hierarchical neural parser. (Ji and Eisenstein, 2014) is a transition-based statisitical model with the representation learning. (Braud et al., 2017) is a transition-based neural model by embedding and composing a number of manually-designed sophisticated atomic features. As shown by Table 2, we can see that transition-based models are able to achieve state-of-the-art performances under both discrete and neural settings, demonstrating the robustness of this framework. Our final model is able to achieve competitive results compared with these systems.

3.4 Analysis

In this subsection, we perform several analysis experiments on the test corpus to illustrate the benefits from dynamic oracle and syntax features, respectively.

First, in order to understand the influence of dynamic oracle, we evaluate the `Relation` performances with respect to the EDU number. In our transition-based model, a subtree with more EDUs requires more transition actions to be produced. As introduced in section 2.3, our neural model with dynamic oracle is robust to error input states. Figure 5 shows the comparison results. We can find that the model with dynamic oracle performs better on spans of the EDU number between [2,4], which is consistent with our intuitions. However, when the EDU number increases to 5+, both settings perform very poorly.

Second, we investigate the benefits by using our proposed syntax features. Intuitively, the dependency parsing outputs exploit tree structures to re-organize sentential words, thus a number of long-distance word pairs are connected directly. We would expect that spans covering a long range of words could be better handled with syntax features. We verify this assumption by comparing performances with respect to the word number in span. As shown by Figure 6, we can see that syntax features do improve the performances of spans containing more words. The observation is consistent with our assumption.

4 Related Work

RST discourse parsing has been studied intensively since early (Marcu, 1997; Marcu, 1999; Soricut and Marcu, 2003). Initial work focuses on statistical models by using manually-designed feature (Soricut and Marcu, 2003; Sagae, 2009; Hernault et al., 2010; Feng and Hirst, 2012; Joty et al., 2013; Feng and Hirst, 2014; Heilman and Sagae, 2015; Wang et al., 2017). Among these studies, transition-based models have achieved state-of-the-art performances (Sagae, 2009; Ji and Eisenstein, 2014; Heilman and Sagae, 2015; Wang et al., 2017). Recently, neural network models have been investigated. Braud et al. (2017) have proposed a transtion-based neural model by using several well-designed atomic features, embedding them directly and then feeding them into feed-forward neural networks. In this work, we suggest hierarchical Bi-LSTMs, following Li et al. (2016) to encode documents, which are more popular in document modeling (Li et al., 2015a; Chen et al., 2016).

Syntax features have been demonstrated useful for RST discourse parsing by a number of studies (Soricut and Marcu, 2003; Sagae, 2009; Hernault et al., 2010; Feng and Hirst, 2012; Joty et al., 2013; Feng and Hirst, 2014; Heilman and Sagae, 2015). The majority work exploits manually-designed features to achieve this goal. The work of Li et al. (2015b) have compared a Tree-LSTM structrue over syntax trees with a sequential LSTM over input sentences, finding that the syntax tree does not show better performances. In this work, we propose a different implicit feature extraction to represent syntax information, using them as a supplementary for RST discourse parsing.

The exploration of dynamic oracle has been proposed for transition-based dependency parsing (Goldberg and Nivre, 2012), bringing significantly better performances for dependency parsing. Our dynamic oracle is mainly inspired by this work, and we apply it on RST discourse parsing. To our knowledge, it is the first work of transition-based RST parsing by using dynamic oracle.

Zhang et al. (2017) suggest a new method of integrating syntax features implicitly. First they extract a sequence of hidden vectors from a supervised neural dependency parsing model, and then feed these implicit syntax features into a neural relation extraction model. The method has been also applied to targeted sentiment analysis (Gao et al., 2017). We follow their work to investigate the implicit syntax feature extraction for RST discourse parsing, and improve this method accordingly for our task.

5 Conclusion

In this work, we investigated an implicit syntax feature extraction method for neural RST discourse parsing. First, we proposed a transition-based neural baseline, and further enhanced it with a dynamic oracle mechanism. Second, we examined the implicit syntax feature extraction method proposed by Zhang et al. (2017) and suggested use outputs of a different neural layer of a Bi-Affine dependency parsing model (Dozat and Manning, 2016). We conducted experiments on standard RST discourse TreeBank (Carlson et al., 2003) to evaluate our proposed models. Results show that our transition-based parser is very competitive after applying dynamic oracle. Further, we find that the proposed implicit syntax features are highly effective, better than explicit Tree-LSTMs.

Acknowledgments

We thank the anonymous reviewers for their constructive comments, which help to improve the paper. This work is supported by National Natural Science Foundation of China (NSFC) grants 61672211 and 61602160, Natural Science Foundation of Heilongjiang Province (China) grant F2016036, Special business expenses in Heilongjiang Province (China) grant 2016-KYYWF-0183.

References

[Bahdanau et al.2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[Braud et al.2016] Chloé Braud, Barbara Plank, and Anders Søgaard. 2016. Multi-view and multi-task training of rst discourse parsers. In *Proceedings of COLING 2016, the 26th ICCL: Technical Papers*, pages 1903–1913.

[Braud et al.2017] Chloé Braud, Maximin Coavoux, and Anders Søgaard. 2017. Cross-lingual rst discourse parsing. *arXiv preprint arXiv:1701.02946*.

[Carlson et al.2003] Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, pages 85–112. Springer.

[Chen et al.2016] Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on EMNLP*, pages 1650–1659.

[Collobert et al.2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

[Devlin et al.2014] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the ACL (Volume 1: Long Papers)*, volume 1, pages 1370–1380.

[Dozat and Manning2016] Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

[Dyer et al.2015] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

[Feng and Hirst2012] Vanessa Wei Feng and Graeme Hirst. 2012. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the ACL: Long Papers-Volume 1*, pages 60–68. Association for Computational Linguistics.

[Feng and Hirst2014] Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the ACL (Volume 1: Long Papers)*, volume 1, pages 511–521.

[Gao et al.2017] Yuze Gao, Yue Zhang, and Tong Xiao. 2017. Implicit syntactic features for target-dependent sentiment analysis. In *Proceedings of the Eighth IJCNLP Processing (Volume 1: Long Papers)*, pages 516–524, Taipei, Taiwan, November. Asian Federation of Natural Language Processing.

[Goldberg and Nivre2012] Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. *Proceedings of COLING 2012*, pages 959–976.

[Hayashi et al.2016] Katsuhiko Hayashi, Tsutomu Hirao, and Masaaki Nagata. 2016. Empirical comparison of dependency conversions for rst discourse trees. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 128–136.

[Heilman and Sagae2015] Michael Heilman and Kenji Sagae. 2015. Fast rhetorical structure theory discourse parsing. *arXiv preprint arXiv:1505.02425*.

[Hernault et al.2010] Hugo Hernault, Helmut Prendinger, Mitsuru Ishizuka, et al. 2010. Hilda: A discourse parser using support vector machine classification. *Dialogue & Discourse*, 1(3).

[Ji and Eisenstein2014] Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the ACL (Volume 1: Long Papers)*, volume 1, pages 13–24.

[Joty et al.2013] Shafiq Joty, Giuseppe Carenini, Raymond Ng, and Yashar Mehdad. 2013. Combining intra-and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of the 51st Annual Meeting of the ACL (Volume 1: Long Papers)*, volume 1, pages 486–496.

[Joty et al.2015] Shafiq Joty, Giuseppe Carenini, and Raymond T Ng. 2015. Codra: A novel discriminative framework for rhetorical analysis. *Computational Linguistics*, 41(3):385–435.

[Kingma and Ba2014] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Li et al.2014] Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on EMNLP*, pages 2061–2069.

[Li et al.2015a] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015a. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

[Li et al.2015b] Jiwei Li, Minh-Thang Luong, Dan Jurafsky, and Eudard Hovy. 2015b. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185*.

[Li et al.2016] Qi Li, Tianshi Li, and Baobao Chang. 2016. Discourse parsing with attention-based hierarchical neural networks. In *Proceedings of the 2016 Conference on EMNLP*, pages 362–371.

[Liu and Lapata2017] Yang Liu and Mirella Lapata. 2017. Learning contextually informed representations for linear-time discourse parsing. In *Proceedings of the 2017 Conference on EMNLP*, pages 1289–1298.

[Mann and Thompson1988] William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

[Manning et al.2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL System Demonstrations*, pages 55–60.

[Marcu1997] Daniel Marcu. 1997. The rhetorical parsing of natural language texts. In *Proceedings of the 35th Annual Meeting of the ACL and Eighth Conference of EACL*, pages 96–103. Association for Computational Linguistics.

[Marcu1999] Daniel Marcu. 1999. A decision-based approach to rhetorical parsing. In *Proceedings of the 37th annual meeting of the ACL on Computational Linguistics*, pages 365–372. Association for Computational Linguistics.

[Marcu2000] Daniel Marcu. 2000. *The theory and practice of discourse parsing and summarization*. MIT press.

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[Morey et al.2017] Mathieu Morey, Philippe Muller, and Nicholas Asher. 2017. How much progress have we made on rst discourse parsing? a replication study of recent results on the rst-dt. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1319–1324.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on EMNLP*, pages 1532–1543.

[Sagae2009] Kenji Sagae. 2009. Analysis of discourse structure with syntactic dependencies and data-driven shift-reduce parsing. In *Proceedings of the 11th ICPT*, pages 81–84. Association for Computational Linguistics.

[Schmidhuber and Hochreiter1997] Jürgen Schmidhuber and Sepp Hochreiter. 1997. Long short-term memory. *Neural Comput*, 9(8):1735–1780.

[Soricut and Marcu2003] Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of NAACL on Human Language Technology-Volume 1*, pages 149–156. Association for Computational Linguistics.

[Surdeanu et al.2015] Mihai Surdeanu, Tom Hicks, and Marco Antonio Valenzuela-Escarcega. 2015. Two practical rhetorical structure theory parsers. In *Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: Demonstrations*, pages 1–5.

[Teng and Zhang2017] Zhiyang Teng and Yue Zhang. 2017. Head-lexicalized bidirectional tree lstms. *Transactions of the ACL*, 5:163–177.

[Wang et al.2017] Yizhong Wang, Sujian Li, and Houfeng Wang. 2017. A two-stage parsing method for text-level discourse analysis. In *Proceedings of the 55th Annual Meeting of the ACL (Volume 2: Short Papers)*, volume 2, pages 184–188.

[Zhang et al.2016] Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *Proceedings of the 54th Annual Meeting of the ACL (Volume 1: Long Papers)*, volume 1, pages 421–431.

[Zhang et al.2017] Meishan Zhang, Yue Zhang, and Guohong Fu. 2017. End-to-end neural relation extraction with global optimization. In *Proceedings of the 2017 Conference on EMNLP*, pages 1730–1740.

[Zhu et al.2013] Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. 2013. Fast and accurate shift-reduce constituent parsing. In *Proceedings of the 51st Annual Meeting of the ACL (Volume 1: Long Papers)*, volume 1, pages 434–443.