Transition-Based Discourse Parsing with Multilayer Stack Long Short Term Memory

Yanyan Jia, Yansong Feng*, Bingfeng Luo, Yuan Ye, Tianyang Liu, and Dongyan Zhao

Institute of Computer Science & Technology, Peking University, Beijing, China
{*jiayanyan*, *fengyansong*, *bingfeng_luo*, *pkuyeyuan*, *ltyang*, *zhaodongyan*}@*pku.edu.cn*

Abstract. Discourse parsing aims to identify the relationship between different discourse units, where most previous works focus on recovering the constituency structure among discourse units with carefully designed features. In this paper, we propose to exploit Long Short Term Memory (LSTM) to properly represent discourse units, while using as few feature engineering as possible. Our transition based parsing model features a multilayer stack LSTM framework to discover the dependency structures among different units. Experiments on RST Discourse Treebank show that our model can outperform traditional feature based systems in terms of dependency structures, without complicated feature design. When evaluated in discourse constituency, our parser can also achieve promising performance compared to the state-of-the-art constituency discourse parsers.

1 Introduction

The task of discourse parsing is to identify the coherence relationship between discourse units which is important for many NLP tasks such as sentiment analysis [31], text summarization [24], question-answering [11] and so on.

Previously, constituency based discoursing parsing method [18, 10, 22] acts as the dominant parsing approach, though suffering from the high complexity and local maximum problem. One noteworthy work is [23]. They first apply dependency parsing to discourse since dependency trees contain much fewer nodes and with Rhetorical Structure Theory (RST)[25] analyzing the relations between element discourse unites(EDUs) is feasible and straightforward. In RST framework, text spans or EDUs are marked with nucleus or satellite according to their importance. The nucleus span is core of the discourse with essential information and the satellite span gives supporting evidence to the nucleus. However, [23] used Eisner Algorithm [8]and Maximum Spanning Tree Algorithm [26]which are both graph-based approaches. They suffer from two main problems. Firstly, they need sophisticated features to represent EDUs, as no grammatical or morphology information could help; Secondly, they suffer from Graph-based parsing method's $O(n^3)$ time complexity.

Since deep learning methods have been booming recently and various models have been proposed such as Long Short Term Memory (LSTM) [17], Gated Recurrent Unit(GRU) [5], attention-based models [4] and enormous varieties. These models have gained significant results in many areas like sequence labeling, question answering and speech recognition. Hence, to address the first inefficient and complex feature engineering problem, we resort to neural network model to gain better performance with fewer features. To be specific, we propose to use LSTM to encode the long term parsing state with the memory-gate architecture using as few features as possible.

Furthermore, inspired by [2], they do sentence-level parsing by modeling characters instead of words and gain good performance in morphologically rich languages. We propose a new multilayer stack LSTM discourse parsing model with novel word based and word/pos based EDU representation methods which give each EDU a unique surface form or portray. However, intra-sentence character modeling for words and inter-sentence word modeling for EDUs are absolutely different tasks and the later is more difficult. Firstly, words have surface form, part of speech tag and the morphology spelling includes only 26 characters. But what do EDUs have? EDUs could contain diverse number of words from the over six hundred thousand words vocabulary taking English for example. There are out of vocabulary words and massive low term frequency words making EDU representation more difficult. Secondly, the number of words in EDUs could be more than the number of letters in the words and the order of the words arranged in the EDUs could be more diverse.

To solve the second problem, we adopt a transition-based dependency parsing model. Though time complexity decline to $O(n)$ with the transition-based parsing approach, most generally the overall accuracy will be worse than the graph-based parsing method, since the transition-based parsing works in a greedy way using the local optimization to gain the global optimization along with an error propagation problem. Hence this is a great challenge to our model and EDU representation method. However, encouragingly, our method gains a better result than the graph-based model with the same or even less features. Besides, this better result to some extent shows LSTM model's high capacity in discourse parsing.

The contribution of this paper is threefold. First, with our novel EDU representation method, each EDU obtains a unique "face" and this surface form helps saving human effort in designing various features. Second, we propose a novel multilayer LSTM model for dependency discourse parsing with encouraging results both in dependency structure and discourse constituency. Third, we initially propose to use LSTM to do dependency discourse parsing.

2 Related Work

Recently, LSTMs have been widely used in multiple NLP tasks with various structures. [30] proposed the tree structure LSTM and used it to predict semantic relevance. [13] investigated the use of Deep Bidirectional LSTM (DBLSTM) in a standard neural network-HMM hybrid system. [7] used a multilayer LSTM to parse the intra-sentence relation. All these tasks take advantage of LSTM's memory-gate mechanism which makes it easier to capture useful information both local and global. So, following [7], we adopt a stack LSTM to do the discourse parsing.

However, many previous works make great efforts in designing multiple features to represent EDUs. [23]used 6 sets of sophisticated features including 15 kinds of features and resources such as WordNet to gain a state-of-art accuracy. [16] used SVM with a greedy bottom-up way to do discourse segmentation and relation labeling. For encoding textual organization they listed 13 kinds of features along with dominance set as defined in [29]. [19] separated discourse parsing into two stages and used two Conditional Random Fields(CRFs) to do the intra-sentential parsing and multi-sentential parsing. They organized the features used in their parsing model into several sets including 8 organizational features, 4 text structural features, 8 n-gram features, 5 dominance set features, 8 lexical chain features, 2 contextual features and 2 substructure features.

Hence, we try to use as few feature engineering as possible to save human effort. With our stack LSTM model we gain an encouraging improvement compared with graph-based model when we use the same or even less features. Furthermore, our multilayer LSTM model with the novel EDU embedding method gives a new direction in modeling EDUs and could be improved in many ways. With this potential method people do not need to rack brains to design features elaborately.

3 Long Short Term Memory Theory

LSTM is designed to cope with the problem of vanishing or exponentially growing gradient over long sequence inherent in recurrent neural networks (RNNs). The typical LSTM cell contains an extra memory "cell" (c) and three kinds of multiplicative gates, the input gate (i), output gate (o) and forget gate (f). The memory cell collects inputs from the input gate and then pass them to the forget gate and finally to the output gate. This mechanism makes the useless information "forgotten" and only the helpful proportion of the current input remained. The computation functions are given below:

$$i_t = \sigma(W_{xi} \times x_t + W_{hi} \times h_{t-1} + W_{ci} \times c_{t-1} + b_i)$$

$$f_t = \sigma(W_{xf} \times x_t + W_{hf} \times h_{t-1} + W_{cf} \times c_{t-1} + b_f)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_{xc} \times x_t + W_{hc} \times h_{t-1} + b_c)$$

$$o_t = \sigma(W_{xo} \times x_t + W_{ho} \times h_{t-1} + W_{co} \times c_t + b_o)$$

$$h_t = o_t \odot tanh(c_t)$$

Here σ is the logistic sigmoid function, $\odot$ denotes the element-wise multiplication operator; b is the bias and t is the time step, W is the weight matrices and h is the hidden state vector. See details in [9]. In this paper we use the multilayer stack LSTM to encode the parsing state. Details will be discussed in Section 5.

4 Transition-based Parsing

Our parser is based on the arc-standard transition inventory[27]. There are four kinds of actions: SHIFT, SWAP, REDUCE-RIGHT and REDUCE-LEFT as shown in Table 1. We use three stacks: Buffer(B), Stack(S), Action(A) to load the input sequence, the dependency tree fragment and the history parsing actions. During decision making stage, when SHIFT operation is selected, the top element in B is popped and then pushed into S. Reduce action pop off two tree fragments from S(represented with m and n) and combine them into a new tree fragment represented with g_r(M,N) or g_r(N,M) depending on the direction of attachment(REDUCE-RIGHT or REDUCE-LEFT), which is then pushed back into S waiting to be processed next. Here m and n denote the tree fragments, M and N denotes the corresponding embeddings. With relation r, we use a recursive neural network g to compose the representations of the two subtrees. The resulting vector embeds the tree fragment in the same space as EDUs. The composition detail was thoroughly explored in prior work [28].

Stack_t	Buffer_t	Action	Stack_{t+1}	Buffer_{t+1}	Dep
S	(M,m),B	SHIFT	(M,m),S	B	_
(M,m),(N,n),S	B	SWAP	(M,m),S	(N,n),B	_
(M,m),(N,n),S	B	REDUCE-RIGHT(r)	(g_r(M,N),m),S	B	$m \xrightarrow{r} n$
(M,m),(N,n),S	B	REDUCE-LEFT(r)	(g_r(N,M),n),S	B	$n \xrightarrow{r} m$

Table 1. Transition actions of the parser

5 Method

5.1 EDU representation

Features used in this paper are listed below:
(a) First, second and last word of the EDU;
(b) First, second and last word's POS tag of the EDU;
(c) Paragraph ID, inter paragraph order, sentence ID, inter sentence order(two form);
(d) 61 frequently used Conjunctions(not listed for space limitation);
(e) Whether the two top EDUs in S and B belong to the same sentence or paragraph;

5.1.1 Standard EDU Embedding Method

To represent the surface form of the input EDUs, we use the features listed in Section 5.1 to encode their property. Here we concatenate two kinds of vectors. Firstly, learned vector representations for each feature extracted from the EDU. Secondly, a fixed vector representation(t) from other language models as pre-trained embedding. Here, we use Paragraph Vector [20] to provide the pre-trained embeddings of the EDUs and these vectors will not change in our model. In the Paragraph Vector framework the fixed length feature representations could encode variable length texts with an unsupervised algorithm. Hence we use it to encode the variable length EDUs. In this Framework, our model do not has a feature number limitation. However, since in this work we fight for using as fewer features as possible to gain high performance and saving human effort, we do not use too many sophisticated features as other works listed in section 2.

We apply a linear map (F) to the resulting vector and passed through a component-wise ReLU as Equation 1. Here, take first word of the EDU(w_1), POS tag of the first word(p_1) as example and fea_1 to fea_n could be any feature, b is the bias and t is the pre-trained embedding(optional).

$$x = max\{0, F(w_1, p_1, fea_1 ... fea_n, t) + b\} \tag{1}$$

5.1.2 Word Based and Word/POS Based EDU Embedding Method

We compute the continuous space vector embeddings of EDUs using bidirectional LSTMs [14] in the EDU embedding layer, multilayer LSTM details will be discussed in Section 5.3. Here, we discuss the word based embedding method of EDUs. This process is shown in Figure:1. For example, when parsing the EDU: "He works in a small company named "OPOG".", the bidirectional LSTM reads the EDU twice, once in forward order and the other in reverse. Each word element is represented with an LSTM cell and we concatenate the two EDU vectors with the opposite directions to represent the EDU. Then the bidirectional LSTM embedding of the EDU is concatenated with other extracted feature representations. The equation is represented with Equation 2. Here, $\overrightarrow{e}$ means the forward order EDU representation and $\overleftarrow{e}$ is the reverse order EDU representation, b is the bias, fea_1 and fea_n mean the standard features extracted from the EDU. However we do not use many sophisticated features and the experiment detail will be discussed in section 6.

Additionally, we propose a word/pos based embedding method of EDUs. Words with low term frequency will be replaced with the pos tag of the corresponding word as shown in Figure:1. We view words appear once in the corpus as low term frequency words. For example in "He works in a small company named "OPOG".", here, OPOG is a low term frequency word and we replace it with it's pos tag "NN". The computation is also done in EDU embedding layer as the word based EDU embedding method.

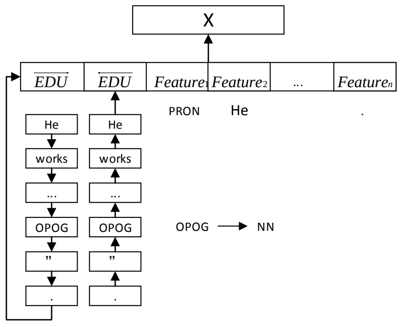

Fig. 1. Word based and Word/POS based EDU Embedding Method

$$x = max\{0, F(\overrightarrow{e}, \overleftarrow{e}, fea_1...fea_n) + b\} \tag{2}$$

5.2 Stack LSTM

Following [7] we use stack LSTM to encode the states. There is a "stack pointer" (marked with "top" in Figure 2) with each LSTM and it determines which cell in the L-STM provides c_t and h_t when computing $t+1$ time step memory cell contents. The stack LSTM provides PUSH and POP operations. POP operation moves the stack pointer to the previous element that could be placed in any location in the stack. PUSH operation adds a new element to the stack pointed by the previous top element (In Figure 2 the "output3" element marked with "top" is pointed by the stack pointer). The output vector of the top element could be viewed as the "summary" of the contents with the current stack configuration and in this paper only the output vector of the top element is processed. For example, when parsing the paragraph shown in Figure 2, the three golden divided EDUs (1. "President Bush insists"; 2. "it would be a great tool"; 3. "for curbing budget deficit.") serve as the input to the LSTM represented with the rectangle in the lowest row. Memory cells and gates illustrated with the oval are located in the middle row. The upper row gives the output of the LSTM represented with rectangle.

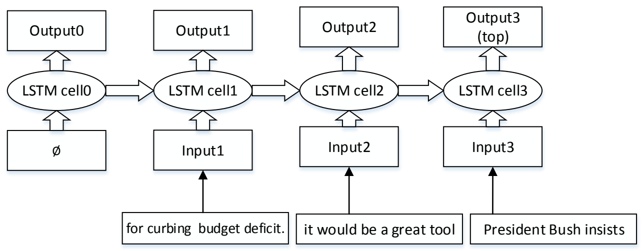

Fig. 2. Stack LSTM

5.3 Multilayer Stack LSTM Discourse Parsing Model

Our arc-standard transition-based parsing model is equipped with three stacks(A, B, S) and takes the EDUs,actions and their intermediate embeddings as inputs. Each stack is a stack LSTM that provides embedding of their current contents. As Figure 3 , stack B contains the embedding of the input EDUs in reverse order of the discourse. Stack S contains the partially constructed dependency subtrees with their embeddings. Stack A is used to preserve the history actions along with the corresponding relations taken by the parser. Actions used here include REDUCE-LEFT(RL), REDUCE-RIGHT(RR) ,SHIFT(SH) and SWAP(SW). Hence, in Figure 3, "RR(cause)" means the action is REDUCE-RIGHT and the relation is "cause".

Notably, our model is a multilayer LSTM composed of three layers, EDU embedding layer, the lower layer and the higher layer. As illustrated in Figure 3, we encode the EDU: "for curbing budget deficit." in EDU embedding layer and details are discussed in Section 5.1.2. Besides word based and word/POS based EDU embedding method, there can be many potential pre-processing method to encode EDUs.

Besides, based on the equations in section 3, here, x_t is the input to the lower layer(gray rectangle) and h_t of the lower layer acts as the input to the higher layer(white rectangle). Output is produced from h_t at the top layer.

5.4 Parsing with Multilayer Stack LSTM Model

Initially, stack S and stack A only contain the empty symbol($\emptyset$). The discourse to be parsed is located in B with reading order from top to the bottom (the "ROOT" symbol). The parsing unit is EDU. At each time step, based on the configurations of B, S and A, the parser computes the representation of the current stack state, predicts the action to take and updates the stacks. Until S contains the full parse tree rooted with the "ROOT" symbol and an empty symbol, B only contains the empty symbol and A filled with all the shift-reduce actions and relations taken by the parser, the parsing process completes.

The action predicting formulas are listed below. In Equation 3, s_t, b_t and a_t are the LSTM embedding of the three stacks; W is a parameter matrix to be learned and c is the bias. These parameters pass through a rectified linear unit (ReLU) nonlinearity [12] to compute the parser state representation at time *t* represented by p_t.

$$p_t = max\{0, W(s_t, b_t, a_t) + c\} \tag{3}$$

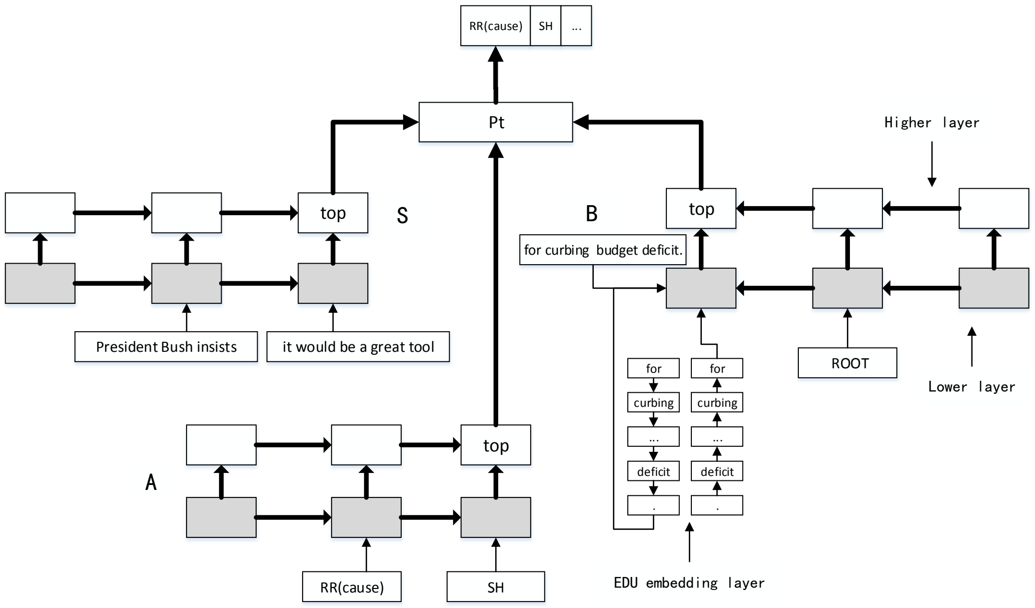

Fig. 3. Discourse parsing example of the paragraph: "President Bush insists it would be a great tool for curbing budget deficit."

Then we apply an affine transform to the embedding of p_t and transfer it to the softmax layer to produce a distribution over parsing decisions(actions and relations) as shown in Equation 4. The parsing actions could be the four kinds of actions listed in Table 1. Here, g_z is a row vector representing the output embedding of the parser action z. b_z is the bias of action z and $A(S, B)$ is the set of the feasible actions that could be taken given the current state of the stacks. According to the chain rule, the probability of the parsing actions z conditioned on the input could be represented as Equation 5.

$$p(z_t|p_t) = \frac{exp(g_{z_t} \times p_t + b_{z_t})}{\sum_{z' \in A(S,B)} exp(g_{z'} \times p_t + b_{z'})} \tag{4}$$

$$p(z|w) = \prod_{t=1}^{|z|} p(z_t|p_t) \tag{5}$$

5.5 Composition Functions

As discussed in section 4, we use recursive neural network as our composition function to encode the dependency tree fragment in stack S in the same vector space as EDU embeddings. We apply the composition function to the <head(h), modifier(d), relation(r)> triples. We concatenate the vectors of the head, modifier and relation, then apply a linear operator and a component-wise nonlinearity. See equation 6 , we use the parser action(such as syntactic relation and direction of attachment) to encode the relation vector.

$$c = tanh(0, V(h, d, r) + b) \tag{6}$$

6 Experiment

6.1 Data

RST Discourse Treebank [3] is a corpus annotated in the framework of RST theory which contains 385 documents from the Wall Street Journal. To make fair comparison with [23] , we follow them to use 380 documents to do the experiments, 342 for training and 38 for testing. Totally, there are 21111 EDUs and 8272 sentences in detail. Every document contains 55 EDUs in average and every sentence contains 2.55 EDUs in average. In this paper we use the 111 fine-grained relations and gain the POS tags using NLTK maxent_treebank_pos_tagger.

6.2 Results and Discussion

6.2.1 Results with Standard EDU Embedding Method

With the standard EDU embedding method in Section 5.1.1, we list results in Table 2. Here we cite results of [23] with their feature sets 1 and 2[1] including 8 features. We listed the methods for comparison, "Eisner" and "MST" are algorithms in [23]. "EDUVEC" means paragraph vector of the EDU and used only in this sub-section, all the experiments in other sub-sections exclude the EDUVEC. "FW" means first word of the EDU, "FP" means the first word's POS tag, "LW" means the last word of the EDU and "Feature only" means we exclude EDUVEC and only use features in the standard EDU embedding way. Here LAS means labeled accuracy, with relation and head. UAS means unlabeled accuracy, with head only.

We can see in Table 2 with only the EDUVEC and no features, the UAS rises encouragingly to 38.79%. This is a good proof for the high power of LSTM model to do discourse parsing. Compared with [23] we use only two features, first word of the EDU(FW), first word's POS of the EDU(FP) in their large feature set 1 and set 2, the UAS rises to 50.04% and both the UAS and LAS are higher than their better Eisner method by 12.61% and 2.39%. When we include the last word(LW) the UAS and LAS are higher than Eisner by 16.91% and 4.87%.

This means our stack LSTM model could gain much higher result than [23], even when we use much fewer features. LSTM discourse parsing model with standard EDU embedding method could gain a much higher result with fewer features over graph-based parsing method even with transition-based framework. That is a solid proof to LSTM model's suitability for discourse parsing task especially when people want to save efforts in designing sophisticated features.

6.2.2 Results with Word Based and Word/POS based EDU Embedding Method

With the method in Section 5.1, the results are listed in Table 3. We use feature sets: a, b, c, d listed in Section 5.1 and "Feature only" means we use the standard EDU

[1] In Table 2 row ID 1 and row ID 2, we use results of [23] when they do experiment with their whole feature set 1 and feature set 2. We list their two feature sets below:

(1) WORD: The first one word, the last one word, and the first bigrams in each EDU, the pair of the two first words and the pair of the two last words in the two EDUs are extracted as features.

(2) POS: The first one and two POS tags in each EDU, and the pair of the two first POS tags in the two EDUs are extracted as features.

ID	Method	Features	UAS	LAS
1	Eisner	8 features(set 1+set 2)	0.3743	0.2421
2	MST	8 features(set 1+set 2)	0.2080	0.1300
3	EDUVEC	NONE	**0.3879**	0.0356
4	EDUVEC+FW	FW	**0.5029**	**0.2467**
5	EDUVEC+FP	FP	**0.3968**	0.1745
6	Feature only	FW+FP	**0.5004**	**0.2660**
7	EDUVEC+FW+FP	FW+FP	**0.5141**	0.2523
8	Feature only	FW+FP+LW	**0.5434**	**0.2908**

Table 2. Comparison with Graph-based Discourse Parsing Method

embedding method in section 5.1.1, here it is the baseline. "WME" means the word based EDU embedding method discussed in Section 5.1.2. Though this is a good direction in embedding EDUs, but there are 19262 tokens in the vocabulary, including upper or lower case of the same words, numbers, special symbols and low term frequency words. These will inevitably give bad effect on the accuracy. So we replace various numbers in the vocabulary with the same token and only use the lower case of the word, hence 15696 tokens remain and the UAS raises to 57.78%. Both the UAS and LAS are higher than the baseline by 1.05% and 2.09%. These give a potential on other EDU pre-processing method. "WPE" means the word/POS based EDU embedding method discussed in Section 5.1.2. When we replace words with low term frequency (tokens appear once in vocabulary) with their POS tags, the vocabulary size decline to 8256 tokens. The UAS rises to 58.61% and 1.88% higher than the baseline.

These results show our new multilayer stack LSTM discourse parsing model with novel EDU embedding method is stronger than the baseline (two layer stack LSTM model without the EDU embedding layer). Our novel EDU embedding method is a potential improvement direction in discourse parsing since this method gives each EDU an portray and could be improved in many ways because besides words' surface form and POS tag, other information could be used in the same way and encode the EDUs into our multilayer stack LSTM discourse parsing model. And this method is useful especially when people do not want to or can not design complex features.

ID	Method	Features	UAS	LAS
1	Feature only	a+b+c+d	0.5673	0.3065
2	Feature +WME	a+b+c+d	**0.5778**	**0.3274**
3	Feature +WPE	a+b+c+d	**0.5861**	**0.3214**

Table 3. Parsing Results with Word based and Word/Pos based EDU Embedding Method

6.2.3 Comparison with structured perceptron based dependency discourse parsing

In order to verify the ability of our multilayer LSTM discourse parser, we implement a dependency discourse parser in perceptron based shift-reduce framework with early update strategy [6] as our baseline. The input EDUs are initially sent to a queue and the algorithm removes the EDUs and pushes them into a stack which stores the temporary dependency structures of the processed part of EDUs according to the transition rules. The algorithm terminates when the queue is empty. With an arc-eager style, the perceptron based parser have four actions: Left-ARC, Right-Arc, Reduce and Shift, details could resort to [1]. Further, We use four perceptrons to predict the score for each transition action and train them in a greedy way. Before each transition, the perceptron predicts the best action instead of the LSTM. We use the features listed in Section 5.1(set a, b, c, d) and use 72 feature templates including the first, second and third element's fea_i of the 12 features both in the queue and the stack. This is the method "Perceptron" in Table 4. "LSTM" means the stack LSTM with standard EDU embedding method in Section 5.1.1, and "WME" and "WPE" have the same meaning as Table 3.

As seen in Table 4, with the standard EDU embedding method, the two layer LSTM discourse parser has an UAS of 56.73%, about 3.62% higher than the baseline but the LAS is lower for 0.55%. But with our word/POS based EDU embedding method in Section 5.1.2, our multilayer LSTM parser wins both in UAS and LAS when we use the same feature sets a, b, c, d. In order to test the performance with complex features we add the pairwise features as [23], so we add feature set e, this time both the WME and WPE raise. Finally, after tuning the dimensions of the LSTM parameters, "LSTMT+WME" gains the higher results.

Though, with few simple features our results have gap with the state-of-art, but [23] use many complex features such as bigrams, pairwise words, pairwise POS tags and pairwise intra sentence/paragraph position, length of EDU, dominate nodes, semantic similarity from WordNet that we all exclude. Our focus is to verify the LSTM like deep learning method's ability in saving human feature design effort especially with the EDU's given "face" (word based and word/POS based EDU embedding method in Section 5.1.2). And our WPE and WME have already win the perceptron based parser(with 72 feature template) using the same four feature sets.

ID	Method	Features	UAS	LAS
1	Perceptron	72 feature templates	0.5311	0.3120
2	LSTM	a+b+c+d	**0.5673**	0.3065
3	LSTM+WME	a+b+c+d	**0.5778**	**0.3274**
4	LSTM+WPE	a+b+c+d	**0.5861**	**0.3214**
5	LSTM+WME	a+b+c+d+e	**0.5968**	**0.3453**
6	LSTM+WPE	a+b+c+d+e	**0.5929**	**0.3316**
7	LSTMT+WME	a+b+c+d+e	**0.6142**	**0.3410**

Table 4. Comparison with perceptron based dependency discourse parser

6.2.4 Performance in discourse constituency

We also evaluate our parser in term of discourse constituency where we evaluate the parsing performance with F measure [15] of the blank tree structure(S), the tree structure with nuclearity indication (N) and the tree structure with rhetorical relation indication but no nuclearity indication (R). To compare our dependency parsing results with constituency works, we convert the dependency trees to constituency trees. In Table 5, other discourse parsing methods include: (1)"Eisner": the state-of-art dependency parser with their full 6 sets features. (2)"Perceptron": our perceptron based dependency parser as discussed in Section 6.2.3 with 72 feature templates designed from feature sets a,b,c,d. (3)"HILDA": SVM based constituency discourse parser [16]. (4)"LeThanh": multi-level rule based constituency parser [21]. (5) "Marcu": decision tree based constituency parser [15]. "LSTM+WPE(abcd)" is our multilayer stack LSTM dependency parser with word/POS based embedding method and feature set a, b, c, d. "LSTM+WME(abcde)" is our word based embedding method with feature set a, b, c, d, e. With few features, "LSTM+WME(abcde)" wins all the other parsers except Eisner (Eisner uses their full 6 sets more complex features, we only use our smaller five simple feature sets) although we are not designed specially for constituency parsing.

ID	Method	S	N	R
1	Eisner	83.4	73.8	57.8
2	LSTM+WME(abcde)	**80.90**	**66.07**	**51.37**
3	LSTM+WPE(abcd)	**79.89**	**63.66**	**49.56**
4	Perceptron	76.22	59.98	44.99
5	HILDA	72.3	59.1	47.8
6	LeThanh	53.7	47.1	39.9
7	Marcu	44.8	30.9	18.8

Table 5. Comparison in discourse constituency

7 Conclusion

In this paper, we first propose to use LSTM for discourse parsing, with this method people could use as few feature engineering as possible to gain high performance. Then we propose a novel multilayer stack LSTM dependency discourse parsing model with word based and word/POS based EDU representation which are potential improvement direction in discourse parsing. With this multilayer LSTM discourse parsing model, we gain better performance than perceptron based discourse parser and graph-based model under the same or even worse feature condition. Even when evaluated in discourse constituency, our results are still encouraging compared to the state-of-the-art constituency discourse parsers.

8 Acknowledgments

We would like to thank Sujian Li, Liang Wang, and the anonymous reviewers for their helpful feedback. This work is supported by National High Technology R&D Program of China (Grant No.2015AA015403, 2014AA015102), Natural Science Foundation of China (Grant No. 61202233, 61272344, 61370055) and the joint project with IBM Research. For any correspondence, please contact Yansong Feng.

References

1. Abney S P, J.M.: Memory requirements and local ambiguities of parsing strategies. Journal of Psycholinguistic Research 20 (1991)
2. Ballesteros, M., Dyer, C., Smith, N.A.: Improved transition-based parsing by modeling characters instead of words with lstms. In: EMNLP 2015, Lisbon, Portugal. pp. 349–359 (2015)

3. Carlson, L., Marcu, D., Okurowski, M.E.: Building a discourse-tagged corpus in the framework of rhetorical structure theory. In: SIGdial Workshop. pp. 1–10 (2001)
4. Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., Bengio, Y.: Attention-based models for speech recognition. CoRR abs/1506.07503 (2015)
5. Chung, J., Gülçehre, Ç., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR abs/1412.3555 (2014)
6. Collins, M., Roark, B.: Incremental parsing with the perceptron algorithm. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain. pp. 111–118 (2004)
7. Dyer, C., Ballesteros, M., Ling, W., Matthews, A., Smith, N.A.: Transition-based dependency parsing with stack long short-term memory. In: ACL 2015, Beijing, Volume 1. pp. 334–343 (2015)
8. Eisner, J.: Three new probabilistic models for dependency parsing: An exploration. In: COLING 1996, August 5-9, 1996. pp. 340–345 (1996)
9. Eyben, F., Böck, S., Schuller, B.W., Graves, A.: Universal onset detection with bidirectional long short-term memory neural networks. In: ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010. pp. 589–594 (2010)
10. Feng, V.W., Hirst, G.: A linear-time bottom-up discourse parser with constraints and post-editing. In: ACL 2014, Baltimore, MD, USA, Volume 1. pp. 511–521 (2014)
11. Ferrucci, D.A., Brown, E.W., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J.W., Nyberg, E., Prager, J.M., Schlaefer, N., Welty, C.A.: Building watson: An overview of the deepqa project. AI Magazine 31(3), 59–79 (2010)
12. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011. pp. 315–323 (2011)
13. Graves, A., Jaitly, N., Mohamed, A.: Hybrid speech recognition with deep bidirectional L-STM. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013. pp. 273–278 (2013)
14. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks 18(5-6), 602–610 (2005)
15. Hahn, U.: The theory and practice of discourse parsing and summarization by daniel marcu. Computational Linguistics 28(1), 81–83 (2002)
16. Hernault, H., Prendinger, H., duVerle, D.A., Ishizuka, M.: HILDA: A discourse parser using support vector machine classification. D&D 1(3), 1–33 (2010)
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997)
18. Ji, Y., Eisenstein, J.: One vector is not enough: Entity-augmented distributed semantics for discourse relations. TACL 3, 329–344 (2015)
19. Joty, S.R., Carenini, G., Ng, R.T., Mehdad, Y.: Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In: ACL 2013, Sofia, Bulgaria, Volume 1. pp. 486–496 (2013)
20. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: ICML 2014, Beijing, China, 21-26 June 2014. pp. 1188–1196 (2014)
21. LeThanh, H.: Generating discourse structures for written texts. Proceedings of the 20th International Conference on Computational Linguistics pp. 329–335 (2004)
22. Li, J., Li, R., Hovy, E.H.: Recursive deep models for discourse parsing. In: EMNLP 2014, October 25-29, 2014, Doha, Qatar,. pp. 2061–2069 (2014)
23. Li, S., Wang, L., Cao, Z., Li, W.: Text-level discourse dependency parsing. In: ACL 2014, Baltimore, Volume 1: Long Papers. pp. 25–35 (2014)
24. Louis, A., Joshi, A.K., Nenkova, A.: Discourse indicators for content selection in summarization. In: SIGDIAL 2010 Conference, Tokyo, Japan. pp. 147–156 (2010)

25. Mann, W., Thompson, S.: Rhetorical structure theory: Toward a functional theory of text organization. In: Text. pp. 243–281 (1988)
26. McDonald, R.T., Crammer, K., Pereira, F.C.N.: Online large-margin training of dependency parsers. In: ACL 2005, University of Michigan, USA (2005)
27. Nivre, J., Scholz, M.: Deterministic dependency parsing of english text. In: COLING 2004, 23-27 August 2004, Geneva, Switzerland (2004)
28. Socher, R., Karpathy, A., Le, Q.V., Manning, C.D., Ng, A.Y.: Grounded compositional semantics for finding and describing images with sentences. TACL 2, 207–218 (2014)
29. Soricut, R., Marcu, D.: Sentence level discourse parsing using syntactic and lexical information. In: HLT-NAACL (2003)
30. Tai, K.S., Socher, R., Manning, C.D.: Improved semantic representations from tree-structured long short-term memory networks. In: ACL 2015, Beijing. pp. 1556–1566 (2015)
31. Voll, K.D., Taboada, M.: Not all words are created equal: Extracting semantic orientation as a function of adjective relevance. In: AI 2007, Gold Coast, Australia, December 2-6, 2007, Proceedings. pp. 337–346 (2007)