Text Sentiment Analysis based on Fusion of Structural Information and Serialization Information

Ling Gan and **Houyu Gong**
College of Computer Science and Tecnology
Chongqing University of Posts and Telecommunications
Chongqing 400065, China
ganling@cqupt.edu.cn,gonghouyu_b103@163.com

Abstract

Tree-structured Long Short-Term Memory (Tree-LSTM) has been proved to be an effective method in the sentiment analysis task. It extracts structural information on text, and uses Long Short-Term Memory (LSTM) cell to prevent gradient vanish. However, though combining the LSTM cell, it is still a kind of model that extracts the structural information and almost not extracts serialization information. In this paper, we propose three new models in order to combine those two kinds of information: the structural information generated by the Constituency Tree-LSTM and the serialization information generated by Long-Short Term Memory neural network. Our experiments show that combining those two kinds of information can give contributes to the performance of the sentiment analysis task compared with the single Constituency Tree-LSTM model and the LSTM model.

1 Introduction

Text sentiment analysis, namely Opinion mining, is an important research direction in the field of Natural Language Processing (NLP). It aims to extract the author's subjective information from text and provide useful values for us. In recent years, there were more and more researchers paying attention to the study of text sentiment analysis.

Up to date, a variety of methods have been developed for improving the performance of sentiment analysis models. The distributed representation for words has been proposed in 2003 (Bengio et al., 2003). This model trained by two kinds of three-layer neural networks generates vectors to represent words. Glove, which is the improvement of the model mentioned above, has been proposed in 2014 (Pennington et al., 2014). The improved representation of words gives contribution to the research of NLP tasks including sentiment analysis, and are used as the input of sentiment analysis models. There are several kinds of deep learning methods to extract text features. Sequential models such as Recurrent Neural Network (Schmidhuber, 1990), Bidirectional Recurrent Neural Networks (Member et al., 1997), Long Short-Term Memory (Hochreiter and Schmidhuber, 2012) and Gated Recurrent Unit (Cho et al., 2014) mainly extract serialization information of text. Multi-layer sequential models have also been proposed for sentiment analysis (Wang et al., 2016)(Tang et al., 2015)(He et al., 2016)(Yang et al., 2016). Tree-structured models extract structural information. The first tree-structured model named Recursive Neural Network has been proposed in 2012 (Socher et al., 2012b), followed by more models such as Matrix-Vector Recursive Neural Network (Socher et al., 2012a) and Recursive Neural Tensor Network (Socher et al., 2013). In 2015, Tree-structured Long Short-Term Memory Neural Network (Tree-LSTM), which combines the LSTM cell and tree-structured models, has been proposed and it outperforms the traditional LSTM and tree-structured neural networks (Le and Zuidema, 2015)(Tai et al., 2015)(Zhu et al., 2015). Different from the traditional tree-structured model, Tree-LSTM uses the LSTM cell to control the information from bottom to top so that it can effectively prevent the vanishing gradient problem.

As mentioned above, though combining the LSTM cell, Tree-LSTM does not really combine the structural information and serialization information. In this paper, we introduce three models: Tree-Composition LSTM (TC-LSTM), Leaf-Tree LSTM (LT-LSTM) and Leaf-Composition-Tree L-

STM (LCT-LSTM). Those models combine those two kinds of information and experiments show that they perform better than the traditional LSTM and the Constituency Tree-LSTM model.

The remainder of this paper is organized as follows: Section 2 introduces the LSTM and Tree-LSTM model which are related to our work. Section 3 introduces the models proposed in this paper. Experimental results are shown in Section 4 and in Section 5 we give the conclusions and future work.

2 Background

2.1 Long Short-Term Memory

Recurrent Neural Network (RNN) (Schmidhuber, 1990) encodes text information according to time. Giving a sentence, its words are encoded by the model in chronological order. For example, x_t represents the vector of input word at time step t, the hidden unit of time step t can be calculated as follows:

$$h_t = tanh(Wx_t + Uh_{t-1} + b). \quad (1)$$

h_t represents the hidden layer at time step t, h_{t-1} represents the hidden state at time step $t-1$, W is the weight matrix of the input layer, U is the weight matrix between h_{t-1} and h_t, b represents the bias and $tanh$ is the activation function which can normalize output information:

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \quad (2)$$

At each time step, the hidden layer produces an output layer:

$$o_t = softmax(Vh_t + b). \quad (3)$$

Usually, the output of the final time step can be used to represent the feature of a sentence. Then, after forward propagation, the weights of model are trained by the backward propagation. Traditional RNN model has the problem of gradient vanish. Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 2012) Neural Network has effectively solved the problem. The model has four gates which help to selectively forget or remember information. A memory cell has also been added to memory information transmitted over time steps. The information calculated by an LSTM unit can be shown as follows (Tai et al., 2015):

$$i_t = \sigma(W^{(i)}x_t + U^i h_{t-1} + b^i), \quad (4)$$

$$f_t = \sigma(W^{(f)}x_t + U^f h_{t-1} + b^f), \quad (5)$$

$$o_t = \sigma(W^{(o)}x_t + U^o h_{t-1} + b^o), \quad (6)$$

$$u_t = tanh(W^{(u)}x_t + U^u h_{t-1} + b^u), \quad (7)$$

$$c_t = i_t * u_t + f_t * c_{t-1}, \quad (8)$$

$$h_t = o_t * tanh(c_t). \quad (9)$$

Here, i_t, f_t, u_t, o_t denote the four gates, c_t is the memory cell, $*$ represents element-wise multiplication. Intuitively, input gate (i_t) and update gate (u_t) denote how much the memory cell update information, forget gate (f_t) determines how much the memory cell forget history information and output gate (o_t) controls how much the hidden unit get information from the cell. σ represents the sigmoid function. The weights of different layers are different but they are shared at each time step in the same layer.

2.2 Tree-LSTM

Dependency Tree-LSTM and Constituency Tree-LSTM are two types of Tree-LSTM structures. We discuss the latter because it achieves a better performance in the sentiment analysis task (Tai et al., 2015). Constituency Tree-LSTM includes three types of layers. Input layer includes the leaf nodes, it consists of the words in the sentence, each word is represented by a vector. Composition layer acts as the hidden layer which composes the information flowing from the leaf nodes to the root node. Each composition unit can be seen as the structural feature of its leaf nodes. The final composition node (root node) represents the structural feature of the whole sentence, it is the input of output layer. LSTM cell is used to control the information flowed bottom-up. Different from the cell in sequential models, the hidden information of a composition node comes from their two child nodes:

$$i_j = \sigma(W^{(i)}x_j + \sum_{l=1}^{N} U_l^{(i)} h_{jl} + b^{(i)}), \quad (10)$$

$$f_{jk} = \sigma(W^{(f)}x_j + \sum_{l=1}^{N} U_{kl}^{(f)} h_{jl} + b^{(f)}), \quad (11)$$

$$o_j = \sigma(W^{(o)}x_j + \sum_{l=1}^{N} U_l^{(o)} h_{jl} + b^{(o)}), \quad (12)$$

$$u_j = tanh(W^{(u)}x_j + \sum_{l=1}^{N} U_l^{(u)} h_{jl} + b^{(u)}), \quad (13)$$

$$c_j = i_j * u_j + \sum_{l=1}^{N} f_{jl} * c_{jl}, \quad (14)$$

$$h_j = o_j * tanh(c_j). \quad (15)$$

It is worth noting that, in the input layer, the information composing the gates only include the input words (x_j) but do not have hidden information (h_{jl}). In the composition layer and output layer, only hidden information from two sub nodes participates in the construction of gates. The structure of Constituency Tree-LSTM model is shown in Figure 1.

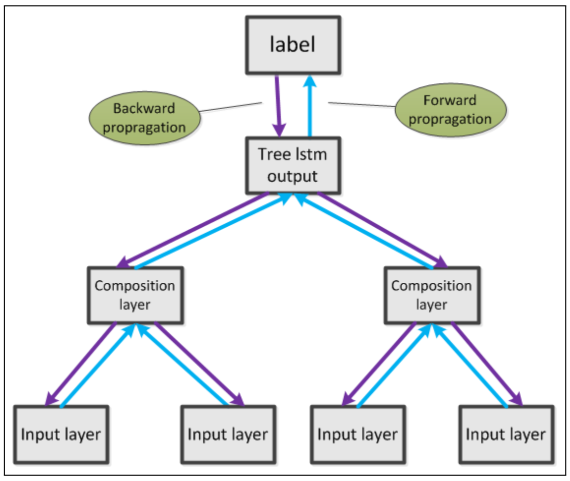

Figure 1: Constituency Tree-LSTM model. The upward arrows represent the direction of forward propagation, the downward arrows represent the direction of the backward propagation.

3 Our model

In this section, we discuss three new models which combine the structural information generated by tree-structured model (Constituency Tree-LSTM) and serialization information generated by sequential model (LSTM).

3.1 TC-LSTM

Constituency Tree-LSTM uses composition node at the root of the tree to represent the feature of sentence. We propose a new model named Tree-Composition LSTM (TC-LSTM) which generates a new feature taking all the leaf nodes, composition nodes and their sequential information into account. Firstly, we use postorder traversal to get all the nodes in the tree, and those nodes are treated as a sequence. The sequence contains not only the words in the sentence, but also the structural information in their parent composition nodes. Then we put the sequence into LSTM model for training, thus obtaining the serialization information of those hidden nodes with structural information. It is worth noting that, though sharing the same hidden nodes, in our first proposed model, the weights of the original Tree-LSTM module and the new added LSTM module are trained independently. The input word vectors firstly perform forward propagation on the Tree-LSTM, and then, the sequence mentioned above is obtained. Then the forward propagation of the sequence is performed on the LSTM model. The output error and gradient of the two modules are obtained through the training label separately. Finally, the backward propagation performs independently of each other. The gradients of back propagation updating the word vectors of input layer only flows from Tree-LSTM module.

TC-LSTM model is shown in Figure 2. After training, we only use the output of LSTM module to test on test data in order to verify the performance of sequential information extracted from tree-structured model.

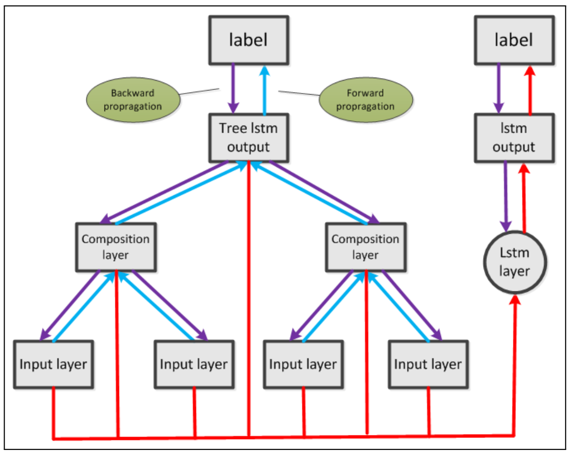

Figure 2: TC-LSTM model. New LSTM module is added to the original Tree-LSTM model. They shared the same hidden nodes but trained separately. Only the output of LSTM module is used to do the prediction.

3.2 LT-LSTM

We propose the Leaf-Tree LSTM (LT-LSTM) model to give a combination of the structural information and sequential information of a sentence. Similar to TC-LSTM, this model has two modules but it only has one output. The first module is the same to Tree-LSTM, and the second module is the LSTM module which takes the leaf nodes of the tree, namely only the words of a sentence as input. During the forward propagation, we add the output of two modules and take the result as the output of the whole model. Gradient of the whole model is generated by the output, and then, assigned to the output of two modules for their backward propagation. Figure 3 shows the structure of the LT-LSTM model.

The output represents the combination of those two kinds of information: structural information and serialization information. Correspondingly, The gradient of the output layer contains the error information of the Tree-LSTM module and L-STM module. Letting the gradient propagate top-down through those two modules can make them learn from each other, thus having the chance to make the comprehensive performance of the whole model better.

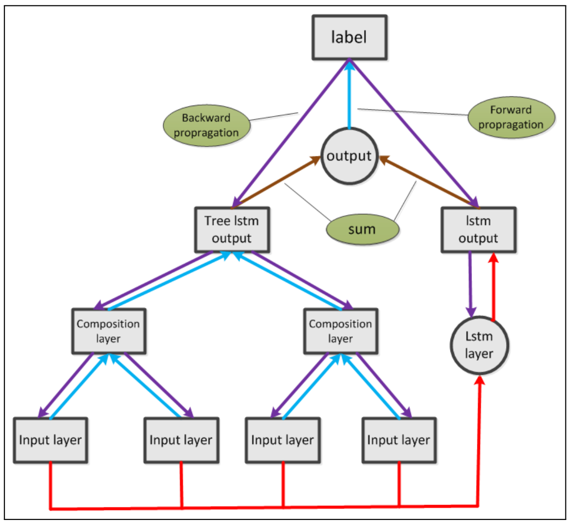

Figure 3: LT-LSTM model. The new added LST-M model only takes the leaf nodes as input. The output represent the fusion of two kinds of information, and its gradient trains the whole model.

3.3 LCT-LSTM

The third model proposed by us is Leaf-Composition-Tree LSTM (LCT-LSTM).Different from the LT-LSTM and TC-LSTM, this model not only takes the composition nodes into consideration when building the LSTM layer, but also makes sum of the outputs of two modules mentioned above. That is, LCT-LSTM can be seen as the composition of TC-LSTM and LT-LSTM for the reason of not only building the sequential feature for the composition nodes which contain the structural information, but also combining the sequential feature and the structural feature of the input sentence. The structure of LCT-LSTM is shown in Figure 4.

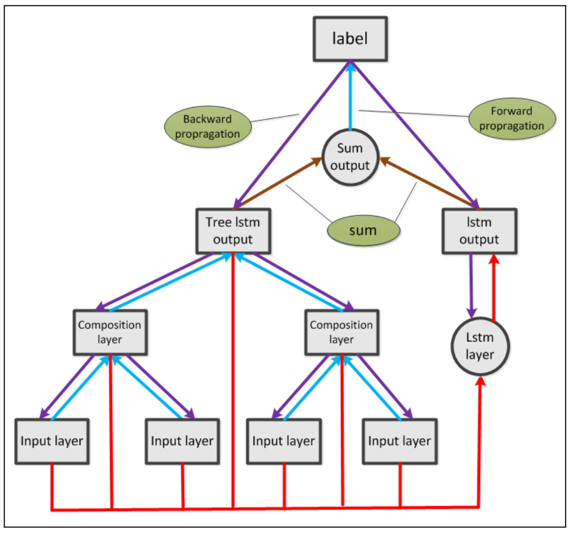

Figure 4: LCT-LSTM model.

4 Experiment

4.1 Dataset

We evaluate our proposed models on the Stanford Sentiment Tree Bank (SST) dataset, which contains sentences collected from movie reviews. The sentences in the dataset are split into three parts: 8544 for training, 1101 for development and 2210 for test. SST dataset has two classification tasks, one for fine-grained classification (five categories: very negative, negative, neutral, positive, and very positive) and the other for binary classification (two categories: negative and positive). The fine-grained subtask is evaluated on 8544/1101/2210 splits, and the binary classification is evaluated on 6920/872/1821 splits (there are fewer sentences because the neutral examples are excluded). Every sentence in the dataset is processed into tree structure, and every phrase (corresponding to the nodes in the tree) in the sentence is also labeled.

4.2 Hyperparameters and Training Details

We use the Glove vectors of 300 dimension (Pennington et al., 2014) to represent the input words. Word embeddings are fine-tuned during training and the learning rate used for the input layer is 0.1, for the other layers is 0.05. Adagrad

	$\theta - all$		
Models	F	B	$\theta - com$
LSTM	271955	271653	271200
Tree-LSTM	317555	317253	316800
TC-LSTM	499510	498906	498755
LT-LSTM	499510	498906	498755
LCT-LSTM	499510	498906	498755

Table 1: Parameters of models. $\theta - all$ represents the number of all the parameters in a model for F (fine-grained) tasks and B (binary tasks). $\theta - com$ represents parameters of composition layer. TC-LSTM, LT-LSTM and LCT-LSTM have the same number of parameters but they are trained in different ways.

algorithm is used for training, the minibatch size set by us is 25, L2-regularization is used for each batch using the value of 1e-4, and the dropout for the output layer is 0.5.

The dimension of the input layer is the same as the word vector, and the hidden layer consisted of tree nodes has the dimension of 150. For the sequential module, both of the inputs and the hidden layer have the dimension of 150, the vectors of leaf nodes are projected into the 150 dimension when put into the sequential part. The numbers of parameters for all the models are shown in Table 1.

Every model is trained on the training set for 20 epochs, and tested on the development set for validation after finishing every epoch. We choose the parameters performing best among them to do the evaluation on the test set. For every model, we repeat experiments for 8 times and take the average of their results as the final performance of the model.

4.3 Baseline

The models proposed by us fuse the structural information and serialization information, so we compare those models with other models which do not combine those two kinds of information. We choose the Constituency Tree-LSTM, LSTM and BiLSTM mentioned in 2015 (Tai et al., 2015) as the baseline models, other tree-structure models such as RNN, MV-RNN and RNTN are also used for comparison.

4.4 Result

The results of experiment are shown in Table 2. We use accuracy to measure the performance of models.

Models	Fine-grained	Binary
LSTM (Tai et al., 2015)	46.4	84.9
Bi-LSTM (Tai et al., 2015)	49.1	87.5
RNN (Socher et al., 2013)	43.2	82.4
MV-RNN (Socher et al., 2013)	44.4	82.9
RNTN (Socher et al., 2013)	45.7	85.4
Constituency Tree-LSTM	50.7	88.0
TC-LSTM	49.6	88.2
LT-LSTM	**51.0**	88.5
LCT-LSTM	50.9	**88.7**

Table 2: The result of accuracy on the test set. Fine-grained represents the five-category classification and the Binary represents the positive/negative classification.

From Table 2, we can see that on the whole, the models fusing structural and serialization information outperform other models which do not combine those two kinds of information. LT-LSTM achieves the best performance among our compared models in the fine-grained subtask and LCT-LSTM has the best performance in the binary subtask. TC-LSTM performs slightly better than Constituency Tree-LSTM in the binary subtask but worse than fine-grained subtask, but it still performs better than other single sequential models and tree-structured models.

We find that building the serialization feature for the nodes in tree-structure (TC-LSTM) does not really help the tree-structural models, but fusing the structural information and serialization information gives help to it. While fusing, adding the hidden nodes containing the structural information to the sequential model (LCT-LSTM) performs better in the binary subtask, but slightly worse in the fine-grained subtask compared to the model does not do so (LT-LSTM).

5 Conclusion

In this paper, we propose three new models in order to explore the effect of fusing the structural and sequential information. We evaluate our models on the Stanford Sentiment Tree Bank (SST). Experiments show that fusing the structural information and sequential information is an effective way to improve the performance of models proposed before. Future work can be focused on finding better ways to fusing those two features. Other models, such as the Bidirectional Long Short-Term Memory (BiLSTM), Bidirectional Tree-LSTM (Teng and Zhang, 2016) and TreeGRU (Kokkinos and Potamianos, 2017) can be used in place of the tree-structured model and the sequential model used in our models. Attention mechanism (Luong et al., 2015) can also be used to do some improvement.

References

Yoshua Bengio, Rjean Ducharme, Pascal Vincent, Christian Jauvin, K Jaz, Thomas Hofmann, Tomaso Poggio, and John Shawe-Taylor. 2003. Journal of machine learning research 3 (2003) 1137–1155 submitted 4/02; published 2/03 a neural probabilistic language model .

Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Computer Science* .

Yunchao He, Liang Chih Yu, Chin Sheng Yang, K. Robert Lai, and Weiyi Liu. 2016. Yzu-nlp team at semeval-2016 task 4: Ordinal sentiment classification using a recurrent convolutional network. In *International Workshop on Semantic Evaluation*. pages 251–255.

Sepp Hochreiter and Schmidhuber. 2012. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Filippos Kokkinos and Alexandros Potamianos. 2017. Structural attention neural networks for improved sentiment analysis .

Phong Le and Willem Zuidema. 2015. Compositional distributional semantics with long short term memory. *Computer Science* .

Minh Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. *Computer Science* .

Member, IEEE, Mike Schuster, and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*. pages 1532–1543.

Schmidhuber. 1990. Recurrent networks adjusted by adaptive critics .

R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank .

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012a. Semantic compositionality through recursive matrix-vector spaces. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. pages 1201–1211.

Richard Socher, Chiung Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2012b. Parsing natural scenes and natural language with recursive neural networks. In *International Conference on Machine Learning, ICML 2011, Bellevue, Washington, Usa, June 28 - July*. pages 129–136.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *Computer Science* 5(1):: 36.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Conference on Empirical Methods in Natural Language Processing*. pages 1422–1432.

Zhiyang Teng and Yue Zhang. 2016. Bidirectional tree-structured lstm with head lexicalization .

Jin Wang, Liang Chih Yu, K. Robert Lai, and Xuejie Zhang. 2016. Dimensional sentiment analysis using a regional cnn-lstm model. In *Meeting of the Association for Computational Linguistics*. pages 225–230.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 1480–1489.

Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. 2015. Long short-term memory over tree structures .