Journal of Intelligent & Fuzzy Systems 39 (2020) 333–340
DOI:10.3233/JIFS-191171
IOS Press

Attention-based LSTM, GRU and CNN for short text classification

Shujuan Yu*, Danlei Liu, Wenfeng Zhu, Yun Zhang and Shengmei Zhao
College of Electronic and Optical Engineering, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu Province, China

Abstract. Text classification is a fundamental task in Nature Language Processing(NLP). However, with the challenge of complex semantic information, how to extract useful features becomes a critical issue. Different from other traditional methods, we propose a new model based on two parallel RNNs architecture, which captures context information through LSTM and GRU respectively and simultaneously. Motivated by the siamese network, our proposed architecture generates attention matrix through calculating similarity between the parallel captured context information, which ensures the effectiveness of extracted features and further improves classification results. We evaluate our proposed model on six text classification tasks. The result of experiments shows that the ABLGCNN model proposed in this paper has the faster convergence speed and the higher precision than other models.

Keywords: Long short term memory, gated recurrent unit, convolutional neural network, attention mechanism, text classification

1. Introduction

Following the trend of machine learning, deep learning theory has made great progress in natural language processing (NLP) tasks, such as answer selection [26], document summarization [17], text classification [30, 31] and so on. The most popular deep learning architectures are Convolution Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). CNNs have good performance on features extraction by convolution kernels, while RNNs are often used to capture the flexible context information.

Within NLP tasks, the complexity of contextual information is a critical issue. Therefore, how to capture useful semantic information has attracted many researchers. According to the different characteristics of CNNs and RNNs, some researchers modified architectures by combining the two classical models. For example, on the basis of TextCNN [31], Lai et al. proposed TextRCNN [24] network by connecting Bi-LSTM to CNNs pooling layer, which not only reserves advantages of selecting discriminative features through the max-pooling layer, but also expands the range of captured contextual information through the recurrent structure.

Another popular breakthrough in feature extraction is the attention mechanism, firstly applied in NLP by Bahdanau [7]. It achieved excellent performances in machine translation [8, 16] by measuring the different parts contribution to the whole, which is in accordance with human's linguistics cognition. One of the representative works is ABCNN [27], which firstly incorporates attention theory into CNNs. With the aims of solving the interdependence problem between two different sentences, ABCNN was established on two weight-sharing CNNs to model sentences pairs. Compared with Bi-CNN mentioned in [27], the results show that

*Corresponding author. Shujuan Yu, College of Electronic and Optical Engineering, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu Province, China. E-mail: yusj@njupt.edu.cn.

computing attention weight on the convolution output achieves considerable improvement on sentence pairs modeling.

It is worth mentioning that the basic Bi-CNN model in ABCNN was highly motivated by siamese architecture [32], which does also inspire our innovation. Technically, siamese architecture was firstly proposed for calculating the similarity between two patterns to verity signatures. Attracted by its excellent performance on calculating similarities, this structure has been widely applied in the field of computer vision and NLP, like object tracking [33], human identification [34], face recognition [35] and so on. While in the field of NLP, siamese network has achieved satisfying performance on calculating semantic textual similarity [36, 37]. However, to the best of our knowledge, siamese architecture has not been widely studied in the text classification tasks, which stir up our inspiration.

2. Model

As shown in Fig. 1, the overall model consists of five parts: Input Layer, LSTM and GRU Layer, Attention Layer, Convolution and Attention-based Pooling Layer, and Output Layer. The details of each component are described in the following sections.

2.1. Input layer

In the sample shown in Fig. 1, the input sentence matrix consists of 5 words, represented as s for the length of sentence. Each word is represented as a d-dimension vector pre-trained by Word2vec embedding [25] or Glove embedding [19], where $d = 300$. Therefore, the sentence can be represented as a feature map of dimension $d \times s$.

2.2. LSTM and GRU layers

RNNs, as a network being calculated over time, performs the same calculations at each time node and relies on the previous time state result. With the addition of hidden layer, RNNs consider the sequence information between words, which is more suitable for NLP tasks. Long short term memory (LSTM), as a variant of RNN, was firstly proposed by Hochreiter and Schmidhuber [22], which mainly overcomes the problem of gradient disappearance through the gate structure. In order to simplify LSTM model without influencing the effect, Cho proposed Gated recurrent unit (GRU) [13] model, which adaptively captures dependencies at different time scales using loop blocks. Given a sequence $S = \{s_0, s_1, ..., s_l\}$, both the LSTM and GRU models need to process the previous word vector results to perform the current word vector calculation.

Figure 2(a) shows the structural block of the LSTM model. It mainly consists of three gates and a storage unit. The three gates are input gate, forget gate and output gate, which control the flow of data for input, storage and output respectively. The activation function f and g in Fig. 2(a) represent the tanh and sigmoid function respectively. In the LSTM model, when the matrix x_t at time t is given, we first consider the forget gate f_t, which controls how many units of the last moment will be transmitted to the current moment.

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1} \oplus x_t\right] + b_f\right) \tag{1}$$

In Equation (1), $\oplus$ presents the stitching of matrix, W_f and Pb_f represent the forget gate weight matrix and the bias respectively. h_{t-1} represents the output of the previous state. σ is the sigmoid function.

The next step is to consider the input gate, which is primarily responsible for the input of sequence at current state.

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1} \oplus x_t\right] + b_i\right) \tag{2}$$

$$\widetilde{c_t} = \tan h\left(W_c \cdot \left[h_{t-1} \oplus x_t\right] + b_c\right) \tag{3}$$

$$c_t = f_t{*}c_{t-1} + i_t{*}\widetilde{c_t} \tag{4}$$

In Equations (2), (3) and (4), i_t is the first part of the calculation, which mainly considers how much information of the current input will be saved in the storage unit. $\widetilde{c_t}$ is the second part of the calculation, which mainly controls how much new information in the input will be saved. c_t is the formula for neuron renewal, which depends on the input gate and the output of forget gate.

Finally, the output gate gets the corresponding output and passes it to the next moment.

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1} \oplus x_t\right] + b_o\right) \tag{5}$$

$$h_t = o_t{*}\tan h\left(c_t\right) \tag{6}$$

The GRU model is shown in Fig. 2(b). The main difference from the LSTM model is that the GRU model does not have storage units. The calculations

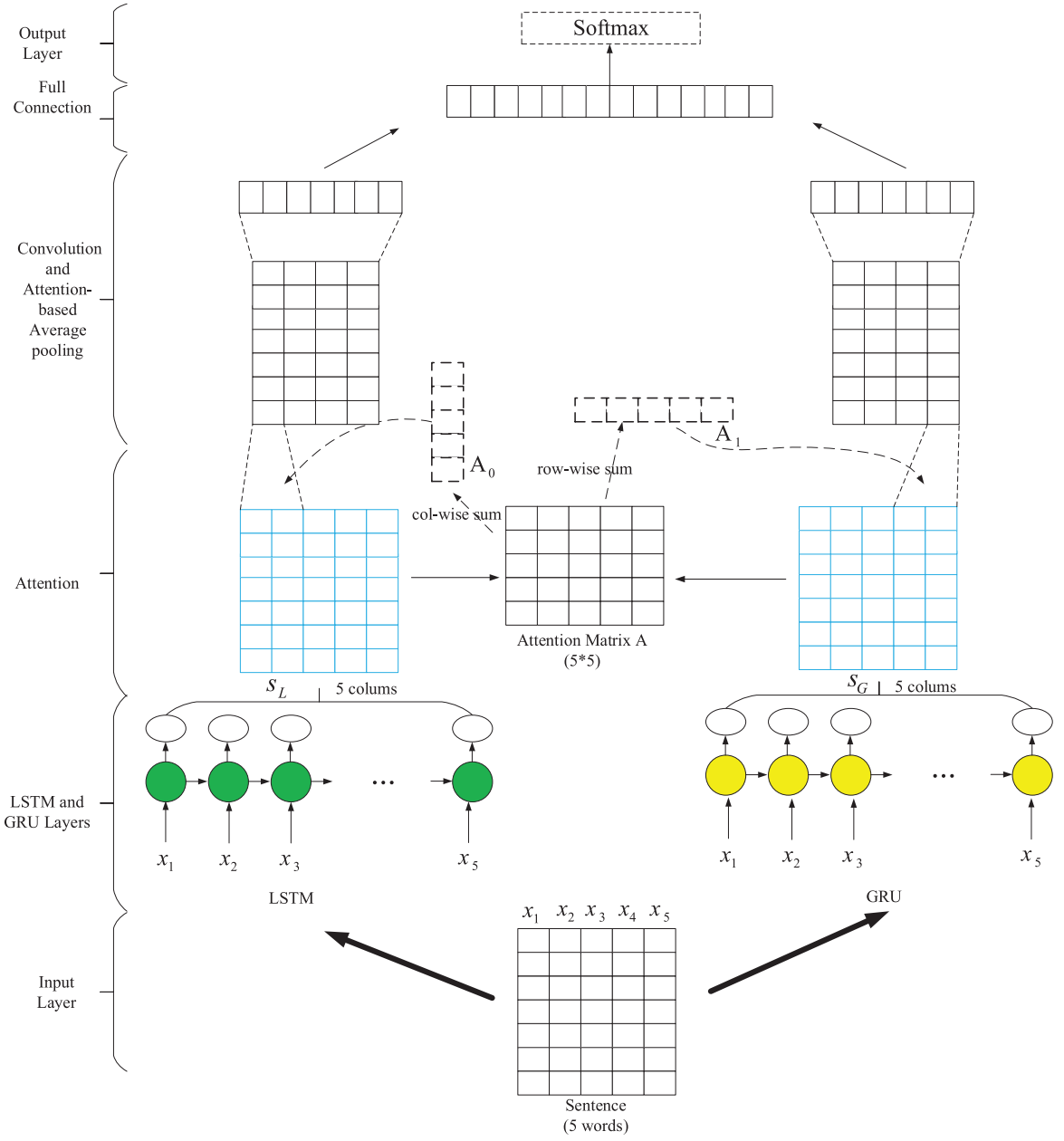

Fig. 1. ABLGCNN architecture.

are as follows:

$$r_t = \sigma\left(W_r \cdot \left[h_{t-1} \oplus x_t\right]\right) \tag{7}$$

$$z_t = \sigma\left(W_z \cdot \left[h_{t-1} \oplus x_t\right]\right) \tag{8}$$

$$\widetilde{h_t} = \tan h\left(W_{\tilde{h}} \cdot \left[r_t \otimes h_{t-1} \oplus x_t\right]\right) \tag{9}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t} \tag{10}$$

In Equations (7), (8), (9) and (10), $\otimes$ represents the multiplication of corresponding elements between the matrices. The reset gate r_t determines the number of units updated from all units' previous activation h_{t-1} in the same layer. And the update gate z_t determines the number of units updated from its activation. At last, GRU activation unit is generated from the previous activation unit and the current candidate unit.

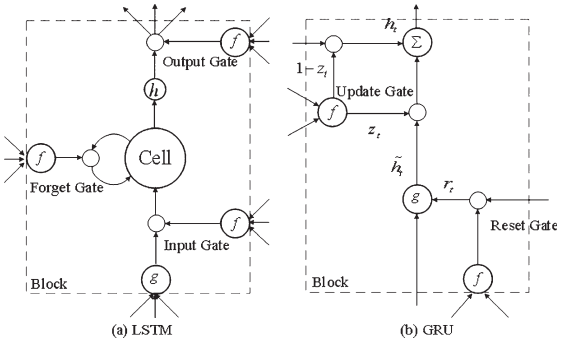

Fig. 2. Illustration for the structures of LSTM and GRU.

As the main contribution in this paper, the input feature map of $d \times s$ is transferred to both LSTM and GRU respectively and simultaneously for capturing text information. As shown in Fig. 1, the output is marked as s_L and s_G respectively, where each column represents a neural unit. It is worth noting that we set the same number of neurons in LSTM and GRU for obtaining the same output size from LSTM and GRU, which ensures the further similarity calculation.

2.3. Attention layer

Adopting the idea from ABCNN-2 [27], we calculate the similarity between two parallel RNNs results and then generate the attention matrix $\mathbf{A}$. Specifically, the i-th row in $\mathbf{A}$ represents the attention distribution of the i-th neuron in s_L with respect to s_G, while the j-th column represents the attention distribution of the j-th neuron in s_G with respect to s_L. Therefore, the attention matrix $\mathbf{A}$ can be regarded as a feature map of s_L in the row and a feature map of s_G in the column. Since the attention matrix $\mathbf{A}$ compares all the elements between s_L and s_G, the large quantity of parameters will lead to overfitting. In order to avoid the problem, the proposed model sums each neural unit value, as the row of attention matrix $\mathbf{A}$ is summed by M attention vector, while the column is summed by N attention vector. This method effectively focuses on each neural unit and derives a single attention weight for that unit.

More formally, let $F_{\text{model}} \in \mathbb{R}^{n \times s}$ be the output of the LSTM and GRU models (model $\in \{LSTM, GRU\}$) where s represents the sentence length and n represents the number of hidden neurons set in LSTM and GRU. Then we define the attention matrix $\mathbf{A} \in \mathbb{R}^{s \times s}$ as follows:

$$\mathbf{A}_{i,j} = Euclidean\left(F_{LSTM}\left[:, i\right], F_{GRU}\left[:, j\right]\right) \tag{11}$$

Note that $Euclidean\,(\cdot)$ is Euclidean distance, where $Euclidean\,(x, y) = \frac{1}{1+|x-y|}$. Based on the calculated attention matrix $\mathbf{A}$, the attention weight $a_{0,j} = \sum \mathbf{A}\left[j, :\right]$ of the j-th neural unit in s_L and the attention weight $a_{1,j} = \sum \mathbf{A}\left[:, j\right]$ of the j-th neural unit in s_G can be generated.

2.4. Convolution and attention-based average pooling

Since LSTM and GRU have captured text information, feature vectors can generate a text matrix. In order to extract more meaningful information, the proposed model performs convolution and pooling operations. Based on the attention mechanism layer, the convolution layer captures features through a convolution kernel with a window size $n \times w$. For example, a new feature c_m is generated from a window of words $x_{m:m+w-1}$ is follows:

$$c_i = f\left(U \cdot x_{m:m+w-1} + b_c\right) \tag{12}$$

In Equation (12), b_c is a bias term and $f\,(\cdot)$ is a nonlinear function like tanh function. The convolution kernel is applied to each possible word window in the sentence $\{x_{1:w}, x_{2:w}, ..., x_{s-w+1:s}\}$ and generates a feature map as follows:

$$c = c_1 \oplus c_2 \oplus, ..., \oplus c_{s-w+1}, c \in \mathbb{R}^{n \times s-w+1} \tag{13}$$

Normally, on the basis of feature mapping, neural network models usually perform maximum pooling or average pooling operations. The proposed model in this paper applies the average pooling operation based on the attention mechanism.

$$F_i^p\left[:, j\right] = \sum_{k=j:j+w} a_{i,k} c_i\left[:, k\right], j = 1...s_i \tag{14}$$

In Equation (14), $F_i^p \in \mathbb{R}^{n \times s_i}$ and i represents the LSTM or GRU model. The output matrix produced by the pooling operation is of the same size as the convolution input matrix, so that multiple convolution pool modules can be stacked to extract more abstract features.

2.5. Output layer

As shown in Fig. 1, the fully-connection layer follows the pooling layer. The outputs are passed to a softmax classifier to predict the semantic relation

Table 1
Summary statistics for the datasets

Data	$C.$	L	M	$Train$	$Test$	$\|V\|$	$\|V_{pre}\|$
MR	2	20	59	9595	1050	19346	16548
SST-1	5	18	51	9645	2210	17863	16262
Subj	2	23	65	9000	1000	21233	17803
CR	2	19	77	3396	350	5219	4978
MPQA	2	3	31	9544	1061	6539	6386
TREC	6	10	33	5452	500	9592	9125

C: number of target classes. L: average sentence length. M: maximum sentence length. $Train/Test$: train/test dataset size. $|V|$: vocabulary size. $\left|V_{pre}\right|$: number of words after pre-trained.

label $\hat{y}$ from a discrete set of classes Y. The calculations are as follows:

$$\hat{p}(y|s) = softmax(W_s \cdot O + b_s) \tag{15}$$

$$\hat{y} = \arg\max_{y} \hat{p}(y|s) \tag{16}$$

Finally, we minimize the categorical cross-entropy loss as following calculation:

$$J(\theta) = -\frac{1}{K}\sum_{k=1}^{K} y_k \log\left(\hat{y}_k\right) + \lambda\theta^2 \tag{17}$$

In Equation (17), K is the number of target classes, $y_k \in \mathbb{R}^K$ is the label in one-hot representation, $\hat{y}_k \in \mathbb{R}^K$ the estimated probability for each class by softmax, and λ is an L2 regularization hyper-parameter. In training, this proposed model uses Adam optimizer with mini-batch.

3. Datasets and experimental setup

We compare our model with others on various datasets. The summary statistics of datasets are shown in Table 1.

MR: Movie reviews [5]. Each review is a sentence. It is a two-categorical data set containing negative and positive reviews.

SST-1: Stanford Sentiment Treebank. It is an extension of MR from [21]. The dataset refines the labels, including very negative part, negative part, neutral part, positive part and very positive part.

Subj: Subjectivity dataset [4] where the task is to classify a sentence as being subjective or objective.

CR: Customer reviews of various products which include cameras, MP3 and so on [15]. It is also a two-categorical dataset which contains negative and positive reviews.

MPQA: Opinion polarity detection subtask of the MPQA dataset [10].

TREC: Question classification dataset [28]. The tasks involves classifying a question into six question types (abbreviation, description, entity, human, location, numeric value).

3.1. Word embedding

Pre-trained word embedding in unlabeled corpus enables better generalization of training datasets [12]. In order to get more practical results, our experiments utilize the Glove embedding trained by [19] and Word2Vec embedding from Google News. The words not present in the pre-trained words are initialized by randomly sampling from uniform distribution in [–0.1, 0.1]. The dimensionality of each word is 300 and the text vectors were trained through the continuous bag-of-words architecture [25].

3.2. Hyper-parameter settings

We randomly select 10% of the data as the test set. The evaluation metric of the five datasets is accuracy. MR and MPQA also use the loss as the metric. The final hyper-parameters settings are as follows.

The dimension of GloVe embedding and Word2Vec is 300. The hidden units of LSTM [22], GRU [13], BLSTM [1] and RCNN [24] are 128. In TextCNN [31] and ABLGCNN (ours), we use 128 convolutional filters and each window sizes is (3, sentence length). The mini-batch size is set as 50, the learning rate of Adam is 0.01 and L2 penalty with coefficient is 10^{-3}. For regularization, we apply the dropout operation [9] with dropout rate of 0.5 for the LSTM, GRU and Convolution Layer.

4. Results

4.1. Comparative analysis of test set accuracy

The experiment results of Glove embedding and Word2vec embedding are shown in Table 2 and Table 3 respectively. The comparison shows that the classification accuracy of the results of two word embedding methods are not much different.

Comparing the accuracy of LSTM, LSTM-Att, BLSTM and BLSTM-Att, it is found that attention mechanism improves the classification results obviously, which is according with its good performance on feature extraction.

Table 2
Results of our ABLGCNN model against other models (word embeddings are Glove)

Model	MR	SST1	SUBJ	CR	MPQA	TREC
LSTM	79.24	44.31	92.9	81.43	84.28	81.00
BLSTM	80.29	45.36	93.8	82.28	84.95	80.40
TextCNN	74.57	38.27	91.4	80.29	83.81	81.60
RCNN	77.71	**46.41**	92.6	82.57	84.48	81.40
LSTM-Att	80.67	44.68	93.0	83.14	85.43	81.20
BLSTM-Att	80.67	45.591	93.7	82.85	84.76	81.40
ABCNN	80.48	45.55	92.70	82.24	85.14	84.00
ABLGCNN (ours)	**81.33**	46.32	**93.9**	**83.71**	**85.62**	**84.40**

LSTM: Long short-term memory [5]. **BLSTM:** Bidirectional LSTM networks for improved phoneme classification and recognition [1]. **TextCNN:** Convolutional neural networks for sentence classification [31]. **RCNN:** Recurrent convolutional neural networks for text classification [24]. **LSTM-Att:** Attention-based LSTM for Aspect-level Sentiment Classification[29]. **BLSTM-Att:** Improved by Integrating Bidirectional LSTM with Two-dimensional Max pooling [18]. **ABCNN:** Attention-based Convolutional Neural Network [27].

Table 3
Results of our ABLGCNN model against other models (word embeddings are Word2vec)

Model	MR	SST1	SUBJ	CR	MPQA	TREC
LSTM	79.14	45.36	90.70	81.14	85.52	90.80
BLSTM	79.05	45.77	91.90	81.42	87.81	91.00
TextCNN	77.33	40.41	90.50	82.28	84.86	92.20
RCNN	79.52	43.63	93.40	81.14	87.62	92.60
LSTM-Att	80.28	45.54	92.70	81.71	87.62	92.00
BLSTM-Att	79.71	46.05	92.90	81.43	**88.00**	**93.80**
ABCNN	80.48	46.09	92.30	82.14	87.08	91.80
ABLGCNN (ours)	**81.33**	**46.59**	**93.6**	**82.57**	87.90	**93.80**

LSTM: Long short-term memory [5]. **BLSTM:** Bidirectional LSTM networks for improved phoneme classification and recognition [1]. **TextCNN:** Convolutional neural networks for sentence classification [31]. **RCNN:** Recurrent convolutional neural networks for text classification [24]. **LSTM-Att:** Attention-based LSTM for Aspect-level Sentiment Classification[29]. **BLSTM-Att:** Improved by Integrating Bidirectional LSTM with Two-dimensional Max pooling [18]. **ABCNN:** Attention-based Convolutional Neural Network [27].

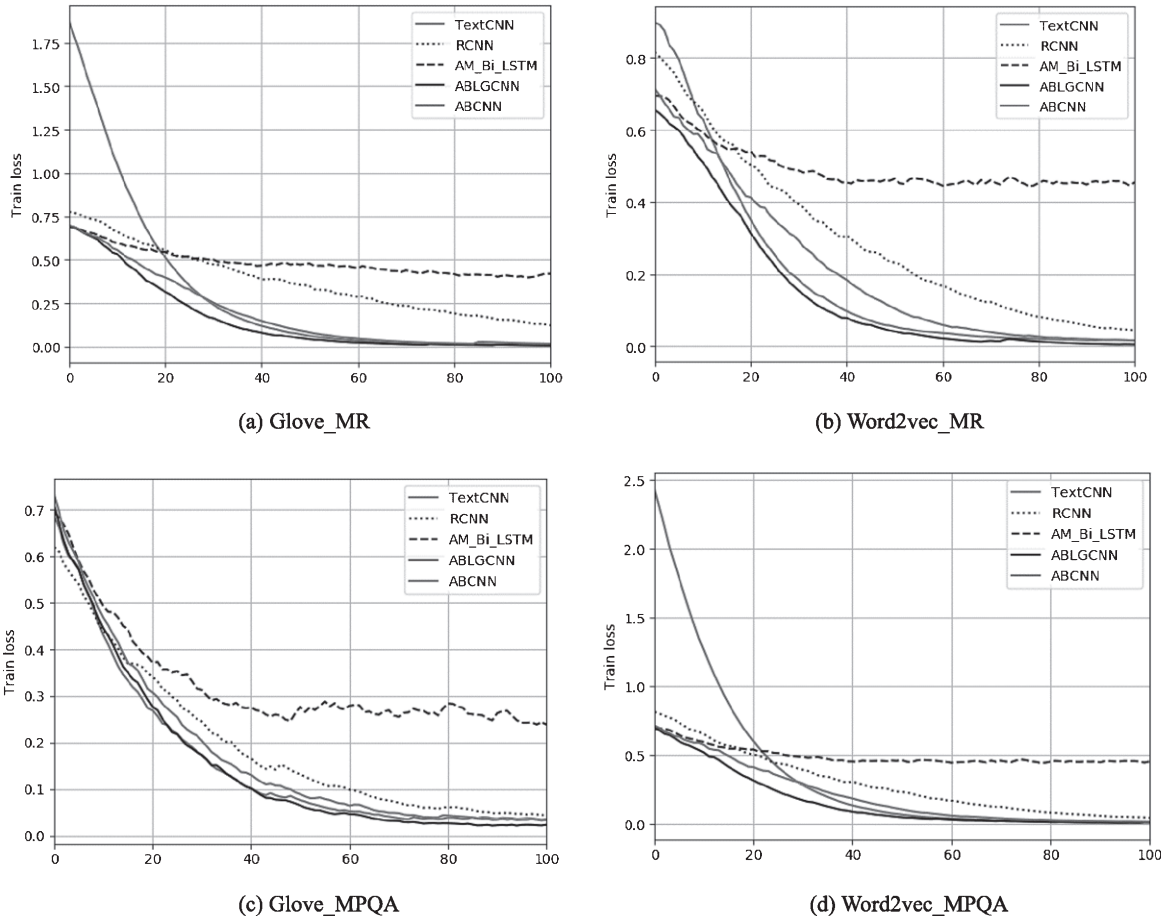

Fig. 3. Loss value of the training set of MR and MPQA.

Comparing the accuracy of TextCNN and RCNN, it is found that the classification of TextCNN is relatively low because of its simplicity in model and the lack of consideration in context relationship, while RCNN effectively improves the performance with the addition of recurrent network.

Compared with other models, the ABLGCNN model proposed in this paper shows the best performance of classification accuracy. As ABCNN model has already achieved quite considerable result, our model technologically adds the parallel RNNs network. With the advantage of RNNs in capturing context information, the attention matrix is generated on the basis of the calculated similarity, which further ensures the importance of extracted feature effectively and increases the classification accuracy.

4.2. Comparative analysis of training set convergence speed

In order to give results analysis from the different aspects, we also conduct experiments on training set convergence performance. It should be noted that for better presenting comparison results, we remain four contrast models on MR and MPQA datasets. The loss values of all models are smoothing processed to avoid the problem possibly caused by shock loss. The number of iterations is 10000. In order to present the convergence changes clearly, we record one point every a hundred iterations.

As shown in Fig. 3, ABLGCNN converges quickly with the notable decreasing slope, while ABCNN shows a slight weaker convergence speed. Same as the disappointing performance in accuracy, the convergence speed is quite poor, which further proves that attention-based Bi-LSTM is not suitable for short text classification. TextCNN shows unstable converging performance in the four experiment. Combined with its unstable classification acuracy, we consider that this simple model is easily caused over-fitting problem. While RCNN shows a relatively good convergence speed, which further proves the effectiveness of adding recurrent network.

From the above two perspectives, the ABLGCNN model not only is of high accuracy in the testing set, but also converges quickly in the training set.

5. Conclusion

In this paper, we propose a new model ABLGCNN for short text classification. The main contribution is the application of LSTM and GRU networks in parallel to capture context information and to calculate the attention weights for extracting more feature information on the basis of parallel results. The experimental results show that ABLGCNN outperforms than other models for its higher classification results and quicker convergence speed.

Though the new model proves its potentials for short text classification, the large quantity of parameters makes it unsuitable for long text classification. An interesting and possible direction is to connect multiple network models in parallel with the attention mechanism.

Acknowledgment

This research was supported by National Natural Science Foundation of China (No. 61871234, No. 61302155).

References

[1] A. Graves, S. Fernández and J. Schmidhuber, Bidirectional LSTM networks for improved phoneme classification and recognition, *In ICANN* , (2005), pp. 799–804.

[2] A.M Rush, S. Chopra and J. Weston, A neural attention model for abstractive sentence summarization, *In Proceedings of EMNLP*, (2015), pp. 379–389.

[3] A.M. de Jesus Cardoso Cachopo, Improving methods for single-label text categorization. (2007).

[4] B. Pang and L. Lee, A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics* , (2004), pp. 271.

[5] B. Pang and L. Lee, Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceeding of the 43rd Annual Meeting on Association for Computational Linguistics* , (2005), pp. 115–124.

[6] D. Kingma and J. Ba, Adam, A method for stochastic optimization, In *Proceedings of ICLR* , (2015).

[7] D. Bahdanau, K. Cho and Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473, (2014).

[8] D. Bahdanau, K. Cho and Y. Bengio, Neural machine translation by jointly learning to align and translate, In *Proceedings of ICLR* , (2015).

[9] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R.R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580. (2012).

[10] J. Wiebe, T. Wilson and C. Cardie, Annotating Expressions of Opinions and Emotions in Language, *Language Resources and Evaluation* **39** (2-3) (2005), 165–210.

[11] J. Li, M.-T. Luong and D. Jurafsky, A hierarchical neural autoencoder for paragraphs and documents, In *Proceedings of ACL* , (2015), pp. 1106–1115.

[12] J. Turian, L. Ratinov and Y. Bengio, Word representations: a simple and general method for semi-supervised learning, In

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics , (2010), pp. 384–394.

[13] K.H.. Cho, B.V. Merriënboer, D. Bahdanau and Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, arXiv preprint arXiv:1409.1259,2014.

[14] K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition. In arXiv prepring arXiv:1506.01497.2015.

[15] M. Hu, B. Liu, Mining and Summarizing Customer Reviews. In Proceedings of ACM SIGKDD, 2004.

[16] M.-T. Luong, H. Pham and C.D. Manning, Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP* , (2015), pp. 1412–1421.

[17] R. Pasunuru, H. Guo and M. Bansal, Towards improving abstractive summarization via entailment generation. In *Proceeding of the workshop on New Frontiers in Summarization* , (2017), pp. 27–32.

[18] P. Zhou, Z. Qi, S. Zheng, et al., Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max pooling. arXiv preprint arXiv: 1611:06639, (2016).

[19] J. Pennington, R. Socher and C. Manning, Glove: Global Vectors for Word Representation[C]//*Conference on Empirical Methods in Natural Language Processing* , (2014), 1532–1543.

[20] R. Collobert, J. Weston, L. Bopttou, M. Karlen, K. Kavukcuglu and P. Kuksa, Natural Language Processing (Almost) from Scratch, *Journal of Machine Learning Research* , (2011), pp. 2493–2537.

[21] R. Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng and C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank. *In Proceedings of the conference on empirical methods in natural language processing (EMNLP)* , (2013), **1631** pp. 1642. Citeseer.

[22] S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural Computation* **9** (8) (1997), 1735–1780.

[23] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, (2015), pp. 448–456.

[24] S. Lai, L. Xu, K. Liu and J. Zhao, Recurrent convolutional neural networks for text classification. In AAAI, (2015), pp. 2267–2273.

[25] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado and J. Dean, Distributed representations of word and phrases and their compositionality, In *Proceedings of NIPS,* (2013), pp. 3111–3119.

[26] W. Bian, S. Li, Z. Yang, G. Chen and Z. Lin, A Compare-Aggregate Model with Dynamic-Clip Attention for Answer Selection, (2017), 1987–1990.

[27] W. Yin, H. Schütze, B. Xiang and B. Zhou, ABCNN: Attention-based convolutional neural network for modeling sentence pairs. arXiv preprint arXiv:1512.05193, (2015).

[28] X. Li and D. Roth, Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume* **1** (2002), pp. 1–7.

[29] Y. Wang, M. Huang, L. Zhao and X. Zhu, Attention-based LSTM for Aspect-level Sentiment Classification, In *Proceedings of the 2016 Conference on Empirical Methods in NLP* , (2016), pp. 606–615.

[30] Y. Shen, Q. Zhang, J. Zhang, et al., Improving Medical Short Text Classification with Semantic Expansion Using Word-Cluster Embedding, *Information Science and Application* (2018), pp. 401–411.

[31] Y. Kim, Convolutional neural networks for sentence classification, In *Proceedings of EMNLP* , (2014), pp. 1746–1751.

[32] J. Bromley, I. Guyon, Y. Lecun, et al., Signature Verification Using a Siamese Time Delay Neural Network. 7th NIPS Conference. (1993).

[33] L. Bertinetto, J. Valmadre, Henriques, F. João, et al., Fully-Convolutional Siamese Networks for Object Tracking. (2016).

[34] V.R. Rama, M. Haloi, G. Wang, et al., Gated Siamese Convolutional Neural Network Architecture for Human Re-identification, (2016).

[35] G. Chen, Y. Shao, C. Tang, et al., Deep transformation learning for face recognition in the unconstrained scene, *Machine Vision and Applications* , (2018).

[36] Y. Benajiba, J. Sun, Y. Zhang, et al., Siamese Networks for Semantic Pattern Similarity, (2018).

[37] E.L. Pontes, S. Huet, C.A. Linhares, et al., Predicting the Semantic Textual Similarity with Siamese CNN and LSTM, (2018).