Neurocomputing 337 (2019) 325–338

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Bidirectional LSTM with attention mechanism and convolutional layer for text classification

Gang Liu, Jiabao Guo*

School of Computer Science, Hubei University of Technology, Wuhan 430072, China

ARTICLE INFO

Article history:
Received 14 March 2018
Revised 7 January 2019
Accepted 26 January 2019
Available online 1 February 2019

Communicated by Shenglan Liu

Keywords:
Long short-term memory
Attention mechanism
Natural language processing
Text classification

ABSTRACT

Neural network models have been widely used in the field of natural language processing (NLP). Recurrent neural networks (RNNs), which have the ability to process sequences of arbitrary length, are common methods for sequence modeling tasks. Long short-term memory (LSTM) is one kind of RNNs and has achieved remarkable performance in text classification. However, due to the high dimensionality and sparsity of text data, and to the complex semantics of the natural language, text classification presents difficult challenges. In order to solve the above problems, a novel and unified architecture which contains a bidirectional LSTM (BiLSTM), attention mechanism and the convolutional layer is proposed in this paper. The proposed architecture is called attention-based bidirectional long short-term memory with convolution layer (AC-BiLSTM). In AC-BiLSTM, the convolutional layer extracts the higher-level phrase representations from the word embedding vectors and BiLSTM is used to access both the preceding and succeeding context representations. Attention mechanism is employed to give different focus to the information outputted from the hidden layers of BiLSTM. Finally, the softmax classifier is used to classify the processed context information. AC-BiLSTM is able to capture both the local feature of phrases as well as global sentence semantics. Experimental verifications are conducted on six sentiment classification datasets and a question classification dataset, including detailed analysis for AC-BiLSTM. The results clearly show that AC-BiLSTM outperforms other state-of-the-art text classification methods in terms of the classification accuracy.

1. Introduction

Text classification is the task of automatically classifying a set of documents into categories from a predefined set and is an important task in many areas of nature language processing (NLP). It has been applied to recommender systems [1], spam filtering system [2] and other areas where it is necessary to understand the sentiment of the users. Sentiment analysis, is a branch of text classification, which is the field of study that analyzes people's opinions, sentiments, appraisals, attitudes, and emotions toward entities and their attributes expressed in written text [3]. At present, there are three main types of the methods for text classification. These methods are: (1) the statistics-based classification methods, such as Bayesian classifier [4]; (2) the connected network learning classification methods, such as neural networks [5]; (3) the rule-making methods, such as decision tree classification [6].

Text classification mainly includes topic classification, question classification and sentiment analysis. Hence, it is hard to find a universal approach, which will consistently performs for all classes of text classification problems. In addition, most of the existing research on traditional text classification centers on one special type of phrase or sentences. These methods only rely on the target sentences or the target words to solve the text classification problem without considering the relationship between each word. In fact, the classification of the text should be determined based on all the contexts. Especially, the traditional sentiment analysis focuses on identifying the polarity of a text (e.g. positive, negative, neutral) based on the language clues extracted from the textual contents of sentences [7–9]. Generally, the sentiment text can often be expressed in a more subtle or arbitrary manner, making it difficult to be identified by simply looking each sentence or word in isolation. Despite having several striking features and successful applications in various fields, the traditional text classification approaches have been shown to have certain weaknesses.

Deep learning technology (DL) [10] has achieved remarkable results in many fields, such as computer vision [11], speech recognition [12] and text classification [13] in recent years. For text classification, most of the studies with the deep learning methods can be divided into two parts: (1) learning word vector representations

* Corresponding author.
E-mail addresses: lg0061408@126.com (G. Liu), Garbo_Guo@163.com (J. Guo).

https://doi.org/10.1016/j.neucom.2019.01.078
0925-2312/

through neural language models [14]; (2) performing composition over the learned word vectors for classification [15]. There are two kinds of deep learning models in text classification: convolutional neural networks (CNNs) [16] and recurrent neural networks (RNNs) [17]. In recent years, many text classification methods based on CNNs or RNNs have been proposed [18–22]. CNNs are able to learn the local response from the temporal or spatial data but lack the ability to learn sequential correlations. In contrast to CNNs, RNNs are specilized for sequential modelling but unable to extract features in a parallel way. In fact, text classification can be considered as the sequential modelling task. Due to the characteristics of RNNs, RNNs are used more frequently in text classification. However, for long data sequences, traditional RNNs cause exploding and vanishing state against its gradient. Long short term memory (LSTM) [23] is a kind of RNNs architecture with long short term memory units as hidden units and effectively solves vanishing gradient and gradient explosion problems. Moreover, it can capture long-term dependencies. In terms of the great power of LSTM to extract the high-level text information, it plays a pivotal role in NLP. Bidirectional long short term memory (BiLSTM) [24] is a further development of LSTM and BiLSTM combines the forward hidden layer and the backward hidden layer, which can access both the preceding and succeeding contexts. Compared to BiLSTM, LSTM only exploits the historical context. Hence, BiLSTM can solve the sequential modelling task better than LSTM. Currently, LSTM and BiLSTM have been applied to text classification and made some achievements [25–28].

For text classification, the vector representation of the text is generally the high-dimensional vector. The high-dimensional vector as the input of LSTM will cause a sharp increase in the network parameters and make the network difficult to optimize. The convolution operation can extract the features while reducing dimensionality of data. Therefore, the convolution operation can be used to extract the features of the text vector and reduce the dimensions of the vector. Although BiLSTM can obtain the contextual information of the text, it is not possible to focus on the important information in the obtained contextual information. Focusing on the important information will improve the accuracy of the classification. Attention mechanism can highlight the important information from the contextual information by setting different weights. The combination of BiLSTM and attention mechanism can further improve the classification accuracy.

To further continue the research in this direction, this paper proposes a novel deep learning architecture for text classification. This new architecture is enhanced BiLSTM using attention mechanism (AM) [29] and the convolutional layer, referred to as attention-based BiLSTM with the convolutional layer (AC-BiLSTM). The basic idea of the proposed architecture is based on the following consideration. The one-dimensional convolutional filters in the convolutional layer perform in extracting n-gram features at different positions of a sentence and reduce the dimensions of the input data. BiLSTM is used to extract the contextual information from the features outputted by the convolutional layer. Attention mechanism has also been successfully applied to text classification [30]. In AC-BiLSTM, attention mechanism is respectively employed to give different focus to the information extracted from the forward hidden layer and the backward hidden layer in BiLSTM. Attention mechanism strengthens the distribution of weights to the variable-length sequences. There are two attention mechanism layers in AC-BiLSTM. The features extracted from the attention mechanism layers are banded together and will be classified by the softmax classifier. In order to verify the performance of the proposed approach, seven comprehensive labeled datasets of experiments (including 6 sentiment analysis datasets and 1 text classification dataset) are conducted. Compared with other state-of-the-art text classification methods, our approach performs better, or at least comparably, in terms of the classification accuracy and the robustness.

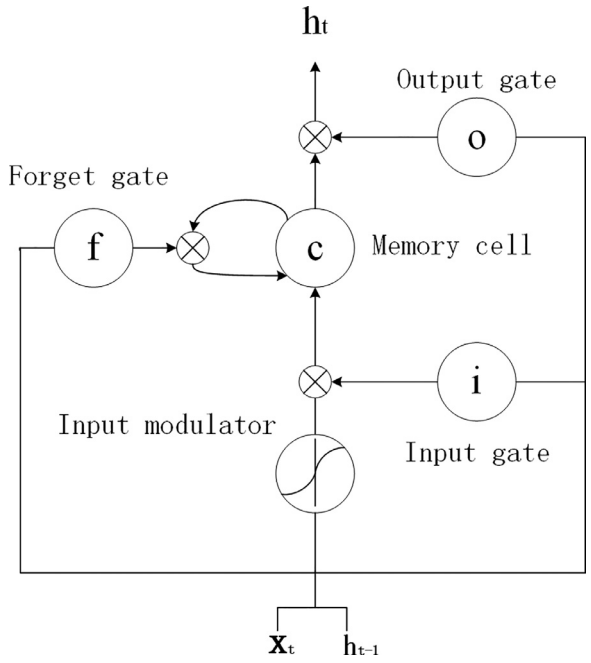


Fig. 1. Illustration of the LSTM unit. The weight matrices are represented by lines with arrows.

The remainder of this paper is organized as follows. Section 2 introduces LSTM, BiLSTM and gives a short literature review on text classification. The proposed approach is presented in detail in Section 3. Experimental results and discussions are reported in Section 4. Finally, some conclusions and possible paths for future research are provided in Section 5.

2. Long short-term memory and related work

2.1. Long short-term memory

RNNs are a kind of feedforward neural networks which have a recurrent hidden state and the hidden state is activated by the previous states at a certain time. Therefore, RNNs can model the contextual information dynamically and can handle the variable-length sequences. LSTM is a kind of RNNs architecture and has become the mainstream structure of RNNs at present. LSTM addresses the problem of vanishing gradient by replacing the self-connected hidden units with memory blocks. The memory block uses purpose-built memory cell to store information, and it is better at finding and exploiting long range context. The memory units enable the network to be aware of when to learn new information and when to forget old information. A LSTM unit consists of the four components and it is as illustrated in Fig. 1. The i is an input gate and it controls the size of the new memory content added to the memory. The f is a forget gate and it determines the amount of the memory that needs to be forgotten. The o is an output gate and it modulates the amount of the output memory content. The c is the cell activation vector and it consists of two components, namely partially forgotten previous memory c_{t-1} and modulated new memory $\tilde{c}_t$. t nominates the tth moment.

The mathematical form of LSTM shown in Fig. 1 is given. The hidden state h_t given input x_t is computed as follows:

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

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2}$$

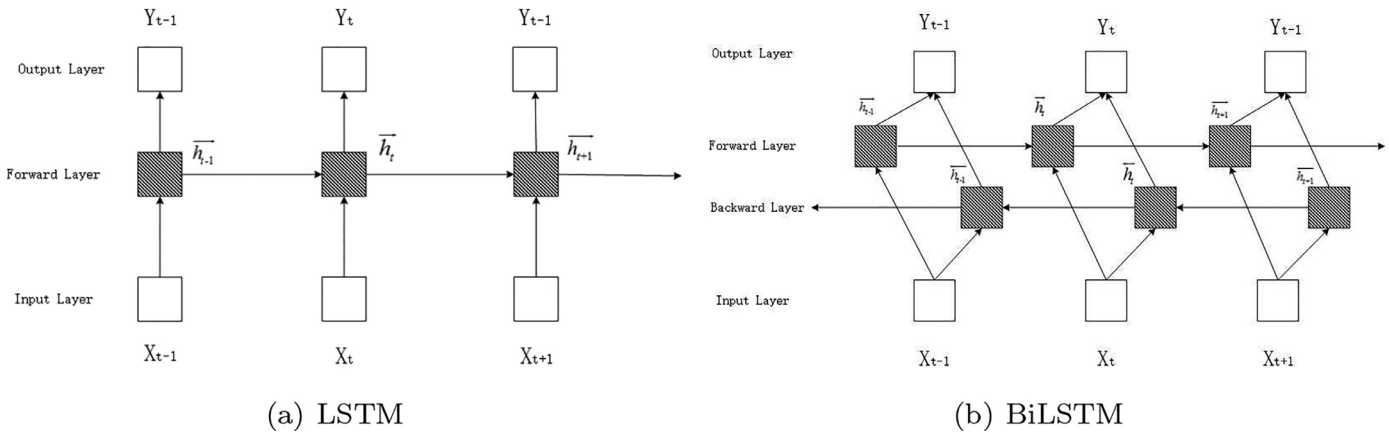


Fig. 2. Illustration of a LSTM model (a) and a BiLSTM model (b).

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{3}$$

$$\tilde{c_t} = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{4}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tilde{c_t} \tag{5}$$

$$h_t = o_t \otimes tanh(c_t), \tag{6}$$

where i_t, f_t, o_t and c_t represent the value of i, f, o and c at the moment t, respectively. The W denotes the self-updating weights of the hidden layer and the term b denotes the bias vector. The $\sigma(.)$ and $tanh(.)$ are sigmoid and hyperbolic tangent function respectively. All the gate values and hidden layer outputs lie within the range of [0, 1]. The operator $\otimes$ denotes element-wise multiplication.

The graphical illustration of the standard LSTM network can be found in the (a) of Fig. 2. The standard LSTM network can only exploit the historical context. However, the lack of future context may lead to incomplete understanding of the meaning of the problem. Therefore, BiLSTM is proposed to access both the preceding and succeeding contexts by combining a forward hidden layer and a backward hidden layer as depicted in the (b) of Fig. 2. The forward and backward pass over the unfolded network over time are carried out in a similar way to regular network forward and backward passes, except that BiLSTM need to unfold the forward hidden states and the backward hidden states for all time steps. The BiLSTM networks are trained using backpropagation through time (BPTT) [24].

2.2. Related work

In deep learning, LSTM is mainly used to process the sequence data. The breadth of applications for LSTM has expanded rapidly in recent years. In order to further improve the performance of LSTM to handle the variable-length sequential information for the requirements of various tasks, many researchers have proposed many methods to improve LSTM. Currently, LSTM and its variants have been employed to produce the promising results on a variety of tasks.

The combination of LSTM and other network structures is an important research direction. Kolawole [31] employed the Child-Sum Tree-LSTM for solving the challenging problem of textual entailment. Their approach is simple and able to generalize well without excessive parameter optimization. The literature [32] demonstrated that LSTM networks predict the subcellular location of proteins given only the protein sequence with high accuracy outperforming current state-of-the-art algorithms. They further improved the performance by introducing convolutional filters and experiment with an attention mechanism which lets the LSTM focus on specific parts of the protein. Wang et al. [33] proposed a regional CNN-LSTM model consisting of two parts: regional CNN and LSTM to predict the valence-arousal (VA) ratings of texts. The proposed regional CNN uses an individual sentence as a region, dividing an input text into several regions such that the useful affective information in each region can be extracted and weighted according to their contribution to the VA prediction. Such regional information is sequentially integrated across regions using LSTM for VA prediction. By combining the regional CNN and LSTM, both local (regional) information within sentences and long-distance dependency across sentences can be considered in the prediction process. Experimental results showed that the proposed method outperforms lexicon-based, regression-based, and NN-based methods proposed in previous studies. Lu et al. [34] proposed a novel model based on LSTM called P-LSTM for sentiment classification. In P-LSTM, three-words phrase embedding is used instead of single word embedding as is often done. P-LSTM introduces the phrase factor mechanism which combines the feature vectors of the phrase embedding layer and the LSTM hidden layer to extract more exact information from the text. The experimental results showed that the P-LSTM achieves excellent performance on the sentiment classification tasks. Chen et al. [27] proposed a divide-and-conquer approach which first classifies sentences into different types, then performs sentiment analysis separately on sentences from each type. Their approach, BiLSTM-CRF, is used to classify opinionated sentences into three types according to the number of targets appeared in a sentence. Each group of sentences is then fed into a one-dimensional convolutional neural network separately for sentiment classification. The literature [35] proposed a deep learning-based approach for temporal 3D pose recognition problems based on a combination of a CNN and a LSTM recurrent network. The paper presents a two-stage training strategy which firstly focuses on CNN training and secondly, adjusts the full method (CNN+LSTM). Le et al. [36] introduced a multi-view recurrent neural network (MV-RNN) approach for 3D mesh segmentation. The architecture combines CNN and a two-layer LSTM to yield coherent segmentation of 3D shapes. The imaged-based CNN are useful for effectively generating the edge probability feature map while the LSTM correlates these edge maps across different views and output a well-defined per-view edge image.

Currently, attention mechanism has become an effective method to select the significant information to obtain the

superior results. Many studies have been conducted on the architecture of attention mechanism and many novel attention mechanisms are proposed. Xu et al. [37] proposed a stochastic "hard" attention mechanism and a deterministic "soft" attention mechanism. The deterministic attention model is an approximation to the marginal likelihood over the attention locations and it is the most widely used attention mechanism. Luong et al. [38] examined two simple and effective classes of attentional mechanism: a global approach which always attends to all source words and a local one that only looks at a subset of source words at a time. These classes differ in terms of whether the attention is placed on all source positions or on only a few source positions. The idea of a global attentional model is to consider all the hidden states of the encoder when deriving the context vector. The local attention mechanism selectively focuses on a small window of context and is differentiable. Lin et al. [39] proposed a self-attention mechanism. The proposed self-attention mechanism allows extracting different aspects of the sentence into multiple vector representations. Vaswani et al. [40] proposed scaled dot-product attention and multi-head attention. Scaled dot-product attention computes the dot products of the input data, divide each by the scaling factor, and apply a softmax function to obtain the weights on the values. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality. Shen et al. [41] proposed bidirectional block self-attention (Bi-BloSA) for fast and memory-efficient context fusion. The basic idea is to split a sequence into several length-equal blocks (with padding if necessary), and apply an intra-block self-attention networks (SAN) to each block independently. The outputs for all the blocks are then processed by an inter-block SAN. The intra-block SAN captures the local dependency within each block, while the inter-block SAN captures the long-range/global dependency. Hence, every SAN only needs to process a short sequence.

The combination of LSTM and attention mechanism can obtain better results. Especially, for sequence problems, attention mechanism has been used successfully in a variety of tasks including text classification, reading comprehension and so on. Yang et al. [42] proposed a hierarchical attention network for document classification. The model has two distinctive characteristics: (i) it has a hierarchical structure that mirrors the hierarchical structure of documents; (ii) it has two levels of attention mechanisms applied at the word and sentence-level, enabling it to attend differentially to more and less important content when constructing the document representation. Experiments conducted on six large scale text classification tasks demonstrate that the proposed architecture outperform previous methods by a substantial margin. Cui et al. [43] presented a simple but novel model called attention-over-attention reader for better solving cloze-style reading comprehension task. The proposed model aims to place another attention mechanism over the document-level attention and induces "attended attention" for final answer predictions. Experimental results show that the proposed methods significantly outperform various state-of-the-art systems by a large margin in public datasets. Li et al. [44] developed a novel model, employing context-dependent word-level attention for more accurate statement representations and question-guided sentencelevel attention for better context modeling. Employing these attention mechanisms, the model accurately understands when it can output an answer or when it requires generating a supplementary question for additional input depending on different contexts. Paulus et al. [45] introduced a neural network model with a novel intra-attention that attends over the input and continuously generated output separately, and a new training method that combines standard supervised word prediction and reinforcement learning (RL). The model obtains a 41.16 ROUGE-1 score on the CNN/Daily Mail dataset, an improvement over previous state-of-the-art models. Human evaluation also shows that our model produces higher quality summaries. Huang [46] introduced a new neural structure called FusionNet, which extends existing attention approaches from three perspectives. First, it puts forward a novel concept of "history of word" to characterize attention information from the lowest word-level embedding up to the highest semantic-level representation. Second, it introduces an improved attention scoring function that better utilizes the "history of word" concept. Third, it proposes a fully-aware multi-level attention mechanism to capture the complete information in one text (such as a question) and exploit it in its counterpart (such as context or passage) layer by layer. Seo et al. [47] introduced the Bi-Directional Attention Flow (BIDAF) network, a multi-stage hierarchical process that represents the context at different levels of granularity and uses bi-directional attention flow mechanism to obtain a query-aware context representation without early summarization. Experimental evaluations show that the model achieves the state-of-the-art results in Stanford Question Answering Dataset (SQuAD) and CNN/DailyMail cloze test. Daniluk et al. [48] proposed a neural language model with a key-value attention mechanism that outputs separate representations for the key and value of a differentiable memory, as well as for encoding the next-word distribution. This model outperforms existing memory-augmented neural language models on two corpora. The literature [49] proposed a simple neural architecture for natural language inference. The approach uses attention to decompose the problem into subproblems that can be solved separately, thus making it trivially parallelizable. On the Stanford Natural Language Inference (SNLI) dataset, it obtains state-of-the-art results with almost an order of magnitude fewer parameters than previous work and without relying on any word-order information. Min et al. [50] presented an attention-based bidirectional LSTM approach to improve the target-dependent sentiment classification. The method learns the alignment between the target entities and the most distinguishing features. The experimental results showed that the model achieves state-of-the-art results.

Some studies have used other methods to improve LSTM and proposed many new variants of LSTM. Wei et al. [51] proposed a transfer learning framework based on a convolutional neural network and a long short-term memory model, called ConvL, to automatically identify whether a post expresses confusion, determine the urgency and classify the polarity of the sentiment. Luo et al. [52] proposed the models based on LSTM for classifying relations from clinical notes. They compared the segment LSTM model with the sentence LSTM model, and demonstrated the benefits of exploring the difference between concept text and context text, and between different contextual parts in the sentence. They also evaluated the impact of word embedding on the performance of LSTM models and showed that medical domain word embedding help improve the relation classification. Hu et al. [53] established a keyword vocabulary and proposed an LSTM-based model that is sensitive to the words in the vocabulary. Experimental results demonstrated that their model outperforms the baseline LSTM in terms of accuracy and is effective with significant performance enhancement over several non-recurrent neural network latent semantic models. Huang et al. [54] discovered that encoding syntactic knowledge (part-of-speech tag) in neural networks can enhance sentence/phrase representation. Specifically, they proposed to learn tag-specific composition functions and tag embeddings in recursive neural networks, and proposed to utilize POS tags to control the gates of tree-structured LSTM networks. The literature [55] proposed a quadratic connections of the LSTM model in terms of RvNNs (abbreviated as qLSTM-RvNN) in order to attack the problem of representing compositional semantics.

Empirical results showed that it outperforms the state-of-the-art RNN, RvNN, and LSTM networks in two semantic compositionality tasks by increasing the classification accuracies and sentence correlation while significantly decreasing computational complexities. Tang et al. [56] introduced a neural network model to learn vector-based document representation in a unified, bottom-up fashion. The model first learns sentence representation with convolutional neural network or LSTM. Afterwards, semantics of sentences and their relations are adaptively encoded in document representation with gated recurrent neural network. Wang et al. [57] regarded the microblog conversation as sequence, and leveraged BiLSTM models to incorporate preceding tweets for context-aware sentiment classification. Their proposed method could not only alleviate the sparsity problem in the feature space, but also capture the long distance sentiment dependency in the microblog conversations. Extensive experiments on a benchmark dataset showed that the BiLSTM models with context information could outperform other strong baseline algorithms.

It can be seen that much research has been done in the basic structure of LSTM to enhance its performance and LSTM has also achieved outstanding results in text classification. These methods are the basis of AC-BiLSTM and therefore they are elaborated.

3. Attention-based BiLSTM with convolutional layer

LSTM is good at handling the variable-length sequences; however, LSTM can not utilize the contextual information from the future tokens and it lacks the ability to extract the local contextual information. Furthermore, not all parts of the document are equally relevant but LSTM can not recognize the different relevance between each part of the document. These problems affect the text classification accuracy of LSTM. In order to improve the performance of LSTM in text classification, this paper attempts to design the novel architecture which helps to address the drawbacks mentioned above by integrating BiLSTM, attention mechanism and the convolutional layer. The proposed architecture is named attention-based BiLSTM with convolutional layer (AC-BiLSTM). In AC-BiLSTM, the convolutional layer extracts n-gram features from the text for sentence modeling. And then BiLSTM accesses both the preceding and succeeding contextual features by combining a forward hidden layer and a backward hidden layer. The attention mechanism (AM) for the single word representation pays more attention to the words related to the sentiment of the text and it can help to understand the sentence semantics. Two attention mechanism layers in AC-BiLSTM process the preceding and succeeding contextual features, respectively. The features processed by the AM layers are concatenated together and then are fed into the softmax classifier. The architecture of the AC-BiLSTM model is shown in Fig. 3.

The entire learning algorithm of AC-BiLSTM is summarized as Algorithm 1, where $\bigoplus$ denotes the future context representation and the history context representation are concatenated together.

The key points of our approach are described in detail as follows.

3.1. Word embedding

Traditional word representations, such as one-hot vectors, face the two main problems: losing word order and oversize of dimensionality. Compared to one-hot representations of word embedding, distributed representations of word embedding is more suitable and more powerful. This paper focuses on text-level classification in this work. Assume that a text has M words, wr_m with $m \in [1, M]$ represents the vector of the mth words in the text. Given a text with words wr_m, AC-BiLSTM embeds the words to vectors through an embedding matrix W_e. The x_m is the vector representation of wr_m, which is formulated by Eq. (7).

$$x_m = W_e wr_m \tag{7}$$

The off-the-shelf word embedding matrices that are already on line can be easily employed. In this paper our approach uses the word2vec method proposed by Mikolve et al. [58] for word embedding. The skip-gram model is used in the word2vec method for the task. The model is trained by using the skip-gram method by max-mizing the average log probability of all the words. The skip-gram model trains semantic embeddings by predicting the target word in accordance with its context and the skip-gram model can also capture semantic relations between words. In this paper, the dimensionality of each word vector is 300.

Algorithm 1 Pseudo-code for AC-BiLSTM.

1: Construct word embedding table using pre-trained word vectors with Eq. (7);
2: Employ the convolutional layer to obtain the feature sequences $Lc = [Lc_1, Lc_2, \ldots, Lc_{100}]$, using Eq. (8);
3: Employ BiLSTM to obtain the preceding contextual features $\overrightarrow{h_f}$ and the succeeding contextual features $\overleftarrow{h_b}$ from the feature sequences, using Eqs. (9) and (10);
4: Employ two attention layers to obtain the future context representation Fc and the historical context representation Hc from the preceding and succeeding contextual features, using Eqs. (13) and (14);
5: Combine the future and historical context representations to obtain the comprehensive context representations $S = [Fc, Hc]$;
6: Feed the comprehensive context representations into the softmax classifier to get the class labels;
7: Update parameters of the model using the loss function Eq. (15) with the Adam method.

3.2. One dimension convolutional layer

In AC-BiLSTM, the single convolutional layer is used to capture the sequence information and reduce the dimensions of the input data. The convolution operation in the convolutional layer is conducted in one dimension. In the convolutional layer, 100 filters with windows size of 3 move on the textual representation to extract the features. As the filter moves on, many sequences, which capture the syntactic and semantic features, are generated. The illustration of the convolutional layer is shown as Fig. 4. Blocks of the same pattern in the feature sequences layer and the filter windows layer corresponds to features for the same window. The dashed lines connect the feature of a window with the source feature sequences. For the convolutional layer, the dimension of the input data is $300 * M$ and the dimension of the output data is $100 * M$. M is the number of words in the text. Hence, the convolutional layer is an effective way for dimensionality reduction.

The convolutional layer, which is between k filters $W_c \in R^{md \times k}$ (R is real number system and the term d is the dimension of word embedding.) and a word embedding vector $x_{i:i+m-1}$ which represents a window of m words starting from the ith word, is used to obtain the features for the window of words in the corresponding feature sequences. Multiple filters with differently initialized weights are used to improve learning capability of the model. The nth feature sequence Lc_n is generated from a window of words $x_{i:i+m-1}$ by

$$Lc_n = g(\mathbf{W_c}^\top x_{i:i+m-1} + b) \in R^k, \tag{8}$$

where b is a bias vector and $g(.)$ represents the nonlinear activation function of the convolutional operation, rectified linear

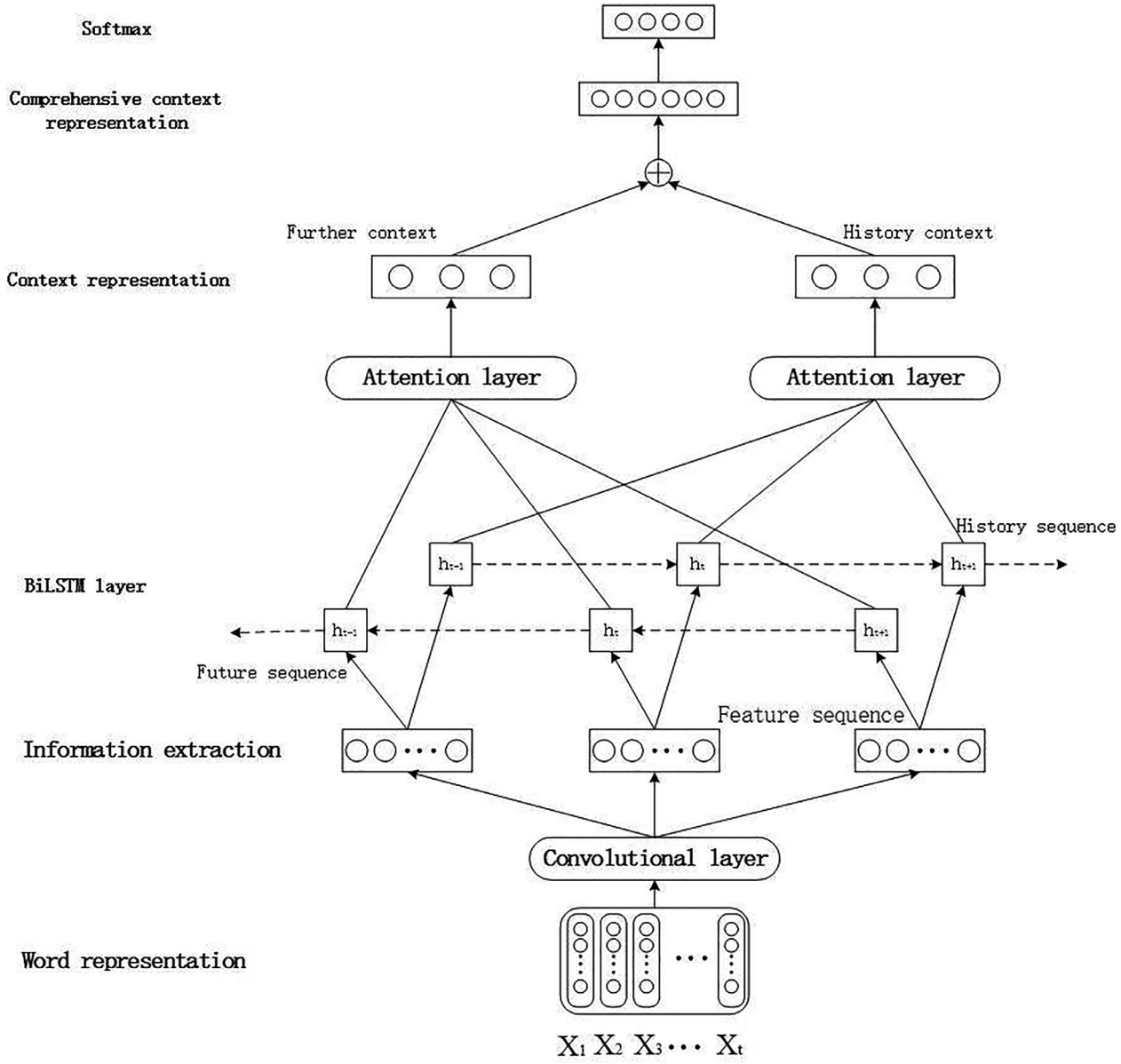


Fig. 3. The architecture of the AC-BiLSTM.

units (ReLU). In AC-BiLSTM, ReLU is used as the nonlinear activation function because it can improve the learning dynamics of the networks and significantly reduce the number of iterations required for convergence in deep networks. Because the number of filters is 100, the feature sequences Lc of words $x_{i:i+m-1}$ are $Lc = [Lc_1, Lc_2, \ldots, Lc_{100}]$.

3.3. BiLSTM and attention mechanism

Intuitively, text classification is the processing of sequential information. However, the feature sequences obtained in a parallel way from the convolutional layer do not contain sequence information. BiLSTM is specilized for sequential modelling and can further extract the contextual information from the feature sequences obtained by the convolutional layer. The effect of BiLSTM is to build the text-level word vector representation. Because all the words have different contributions to the sentiment of the context, assigning different weights to words is a common way of solving the problem. Attention mechanism is to assign different weights to words to enhance understanding of the sentiment of the entire text. Hence, BiLSTM and attention mechanism can improve classification efficiency.

BiLSTM obtains the annotations of words by summarizing information from both directions (forward and backward) for words, and hence the annotations incorporate the contextual information. BiLSTM contains the forward LSTM (represented as $\overrightarrow{LSTM}$) which reads the feature sequences from Lc_1 to Lc_{100} and the backward LSTM (represented as $\overleftarrow{LSTM}$) which reads from Lc_{100} to Lc_1. Formally, the outputs of BiLSTM are stated as follows:

$$\overrightarrow{h_f} = \overrightarrow{LSTM}(Lc_n), n \in [1, 100] \tag{9}$$

$$\overleftarrow{h_b} = \overleftarrow{LSTM}(Lc_n), n \in [100, 1] \tag{10}$$

An annotation for a given feature sequence Lc_n is obtained by the forward hidden state $\overrightarrow{h_f}$ and the backward hidden state $\overleftarrow{h_b}$. These states summarize the information of the entire text centered around Lc_n and implement the word encoding.

Attention mechanism can focus on the features of the keywords to reduce the impact of non-keywords on the text sentiment and

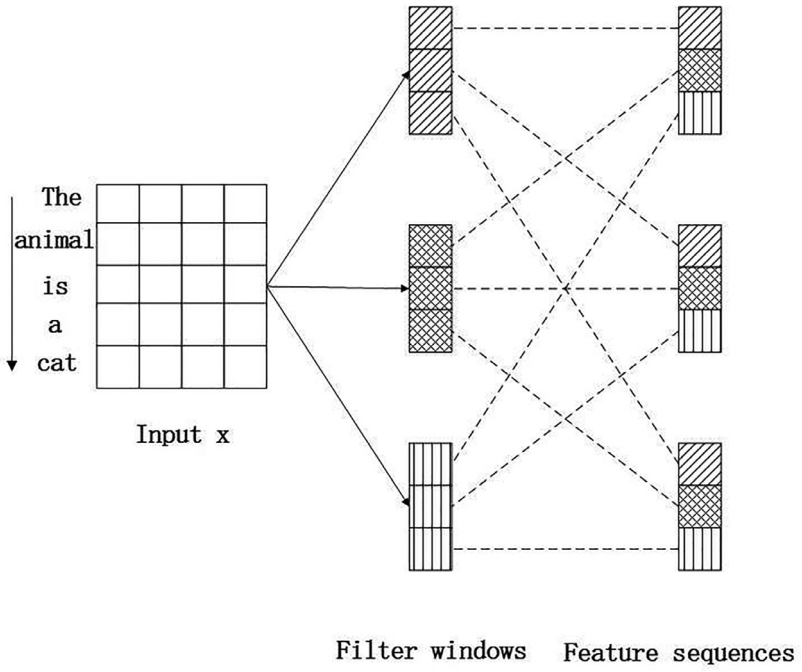


Fig. 4. The architecture for the convolution operation.

it is considered as a fully-connected layer and a softmax function. The work process of attention mechanism in AC-BiLSTM is following detailed.

The word annotation $\overrightarrow{h_f}$ is first fed to get $\overrightarrow{u_f}$ by one layer perceptron as a hidden representation of $\overrightarrow{h_f}$. The $\overrightarrow{u_f}$ is formulated as follows:

$$\overrightarrow{u_f} = tanh(w\overrightarrow{h_f} + b), \tag{11}$$

where w and b are represented as the weight and bias in the neuron, $tanh(.)$ is hyperbolic tangent function. The model uses the similarity between $\overrightarrow{u_f}$ and a word level context vector $\overrightarrow{v_f}$ to measure the importance of each word. And then it uses the softmax function to get the normalized weight $\overrightarrow{a_f}$ of each word. $\overrightarrow{a_f}$ is formulated as follows:

$$\overrightarrow{a_f} = \frac{exp(\overrightarrow{u_f} * \overrightarrow{v_f})}{\sum_{i=1}^{M}(exp(\overrightarrow{u_f} * \overrightarrow{v_f}))}, \tag{12}$$

where M is the number of words in the text and $exp(.)$ is the exponential function. The word level context vector $\overrightarrow{v_f}$ can be seen as a high-level representation of the informative words over the words and is randomly initialized and jointly learned during the training process.

After that, a weighted sum of the forward read word annotations based on the weight $\overrightarrow{a_f}$ is computed as the forward context representation Fc. The Fc is the part of the output of the attention layer, and it can be expressed as:

$$Fc = \sum(\overrightarrow{a_f} * \overrightarrow{h_f}) \tag{13}$$

Similar to $\overrightarrow{a_f}$, $\overleftarrow{a_b}$ can be calculated using the backward hidden state $\overleftarrow{h_b}$. Like Fc, the backward context representation Hc is also the part of the output of the attention layer, and it can be expressed as:

$$Hc = \sum(\overleftarrow{a_b} * \overleftarrow{h_b}) \tag{14}$$

AC-BiLSTM obtains an annotation for a given feature sequence Lc_n by concatenating the forward context representation Fc and backward context representation Hc. Finally, the comprehensive context representations $S = [Fc, Hc]$ are obtained. The comprehensive context representations are considered as the features for text classification. In AC-BiLSTM, the dropout layer and the softmax layer are used to generate the conditional probabilities over the class space to achieve classification. The purpose of the dropout layer is to avoid overfitting.

Currently, the cross entropy is a commonly used loss function to evaluate the classification performance of the models. It is often better than the classification error rate or the mean square error. In our approach, Adam optimizer [59] is chosen to optimize the loss function of the network. The model parameters are fine-tuned by Adam optimizer which has been shown as an effective and efficient backpropagation algorithm. The cross entropy as the loss function can reduce the risk of a gradient disappearance during the process of stochastic gradient descent. The loss function can be denoted as follows in Eq. (15)

$$L_{total} = -\frac{1}{num}\sum_{Sp}[y \ln o + (1-y)\ln(1-o)] \tag{15}$$

where num is the number of training samples, Sp represents the training sample, y is the label of the sample, o is the output of AC-BiLSTM.

The main contributions and originality of AC-BiLSTM are as follows:

(1) the convolutional layer extracts the low-level semantic features from the raw text and is used for dimensionality reduction. For text classification, the vector representation of the entire document is generally the high-dimensional vector. The parameters of BiLSTM will increase significantly when BiLSTM is used to capture the semantics of the entire document. However, too many network parameters will increase the difficulty of network optimization. Directly reducing the dimensionality of the text vector will lose a lot of information and reduce the accuracy of classification. The convolutional layer is supposed to be good at extracting robust and abstract features of the input. In addition, the convolutional layer can also reduce the dimension of the input data. Therefore, one dimension convolutional layer can extract the feature information of

the text vector while reducing the dimensionality of the text vector;

(2) BiLSTM extracts the contextual information from the low-level semantic features. RNNs is a biased model, where later words are more dominant than earlier words [60]. Therefore, the information extracted by LSTM cannot effectively represents the actual semantics of the text. Since BiLSTM can access both the preceding and succeeding contextual features, the features extracted by BiLSTM can more realistically represent the actual semantics of the text. Compared with extracting the contextual information directly from the text, extracting the contextual information from the low-level semantic features can improve the efficiency of extracting information of BiLSTM;

(3) The forward hidden layer and backward hidden layer in BiLSTM use their respective attention mechanism layers. Since BiLSTM can access both the preceding and succeeding contexts, the information obtained by BiLSTM can be considered as two different representations of the text. The same information may use different representations in the information obtained by BiLSTM. Therefore, using attention mechanism for each representation of the text can better focus on the respective important information and avoid mutual interference of the important information in the different representations. Moreover, the attention mechanism layers in AC-BiLSTM make the understanding of text semantics more accurate.

Hence, our approach effectively improves the classification accuracy.

4. Experiments

Experiments are conducted to evaluate the performance of the proposed approach for text classification on various benchmarking datasets. In this section, the experimental setup and baseline methods followed by the discussion of results are described.

4.1. Experimental setup

(a) Datasets

Our model is evaluated on text classification task (including sentiment and question classification) using the following datasets. Summary statistics of these datasets are as follow:

MR: Movie review sentence polarity dataset v1.0 [14]. It contains 5331 positive snippets and 5331 negative snippets extracted from Rotten Tomatoes web site pages where reviews marked with “fresh” are labeled as positive, and reviews marked with “rotten” are labeled as negative.

IMDB: A benchmark dataset for sentiment classification [61]. It is a large movie review dataset with full-length reviews. The task is to determine if the movie reviews are positive or negative. Both the training and test set have 25 K reviews.

RT-2k: The standard 2000 full-length movie review dataset [7]. Classification involves detecting positive/negative reviews.

SST-1: Stanford Sentiment Treebank an extension of MR but with train/dev/test splits provided and fine-grained labels (very positive, positive, neutral, negative, very negative), re-labeled by Socher [62].

SST-2: Binary labeled version of Stanford sentiment treebank, in which neutral reviews are removed, very positive and positive reviews are labeled as positive, negative and very negative reviews are labeled as negative [62].

Subj: The subjectivity dataset consists of subjective reviews and objective plot summaries [8]. The task of subjectivity dataset is to classify the text as being subjective or objective.

TREC: TREC question dataset task involves classifying a question into 6 question types [52]. TREC divides all questions into 6 categories, including location, human, entity, abbreviation, description and numeric. The training dataset contains 5452 labelled questions while the testing dataset contains 500 questions.

Table 1
Summary statistics for the datasets after tokenization.

Data	c	l	N	$\|V\|$	Test
MR	2	20	10,662	18,765	CV
IMDB	2	231	50,000	392,000	25000
SST-1	5	18	11,855	17,836	2210
SST-2	2	19	9613	16,185	1821
Subj	2	23	10,000	21,323	CV
RT-2K	2	787	2000	51,000	CV
TREC	6	10	5952	9592	CV

We test our model on various benchmarks. Summary statistics of the datasets are in Table 1. The c is the number of the target classes, l means the average sentence length, N is the dataset size and $|V|$ is the vocabulary size. “Test” is the test set size and “CV” means that there is no standard train/test split and thus 10-fold CV is used.

(b) Parameter settings

Our experiments use accuracy as the evaluation metric to measure the overall classification performance. During training AC-BiLSTM for feature extraction in the text, the input sequence x_m is set to the mth word embedding (a distributed representation for a word [63]) in a input sentence. Publicly available word vectors trained from Google News are used as pre-trained word embeddings. The size of these embeddings is 300. The memory dimension of BiLSTM is set to be 150 and the number of filters of length 3 is set to be 100 in the convolutional layer. The training batch size for all datasets is set as 50. The dropout rate is 0.7. A back-propagation algorithm with Adam stochastic optimization method is used to train the network through time with the learning rate of 0.001. After each training epoch, the network is tested on validation data. The log-likelihood of validation data is computed for convergence detection.

4.2. Baseline methods

This paper benchmarks the following baseline methods for text classification, they are effective methods and have achieved some good results in text classification:

SVM: Support Vector Machine [64].

MNB: Multinomial naive Bayes with uni-bigrams [65].

NBSVM: SVM variant using naive Bayes log-count ratios as feature values proposed by Wang and Manning [65].

RAE: Semi-supervised recursive auto-encoders with pre-trained word vectors from Wikipedia proposed by Socher et al. [66].

MV-RNN: Recursive neural network using a vector and a matrix on every node in a parse tree for semantic compositionality proposed by Socher et al. [67].

RNTN: Recursive deep neural network for semantic compositionality over a sentiment treebank using tensor-based feature function proposed by Socher et al. [62].

Paragraph-Vec: An unsupervised algorithm learning distributed feature representations from sentences and documents proposed by Le Mikolov [68].

DCNN: Dynamic convolutional neural network with dynamic k-max pooling operation proposed by Kalchbrenner et al. [69].

CNN-static: 1d-CNN with pre-trained word embedding vector from word2vec proposed by Kim [18].

CNN-non-static: 1d-CNN with pre-trained word embedding and fine-tuning optimizing strategy proposed by Kim [18].

CNN-multichannel: 1d-CNN with two sets of pre-trained word embeddings proposed by Kim [18].

Table 2
Experimental results of sentiment classification accuracy. % is omitted and "-" indicates no data and this dataset is not used by the method.

Model	MR	IMDB	SST-1	SST-2	RT-2k	Subj
SVM	–	89.2	40.7	79.4	87.4	91.7
MNB	79.0	86.6	–	–	85.9	93.6
NBSVM	79.4	91.2	–	–	89.5	93.2
RAE	77.7	–	43.2	82.4	–	–
MV-RNN	79.0	–	44.4	82.9	–	–
RNTN	–	–	45.7	85.4	–	–
Paragraph-Vec	–	–	48.7	87.8	–	–
DCNN	–	–	48.5	86.8	–	–
CNN-static	81.0	–	45.5	86.8	–	93.0
CNN-non-static	81.5	–	48.0	87.2	–	93.4
CNN-multichannel	81.1	–	47.4	88.1	–	93.2
DRNN	–	–	49.8	86.6	–	–
Multi-task LSTM	–	–	49.6	87.9	–	–
Tree-LSTM	–	–	**50.6**	86.9	–	–
P-LSTM	–	91.5	–	–	89.3	93.8
C-LSTM	–	–	49.2	87.8	–	–
LSTM	80.1	87.0	48.0	86.4	86.7	91.3
BiLSTM	80.3	87.9	48.4	88.0	87.2	92.3
AC-BiLSTM	**83.2**	**91.8**	48.9	**88.3**	**93.0**	**94.0**

DRNN: Deep recursive neural networks with stacked multiple recursive layers proposed by Irsoy and and Cardie [70].

Multi-task LSTM: A multi-task learning framework using LSTM to jointly learn across multiple related tasks proposed by Liu et al. [71].

Tree LSTM: A generalization of LSTM to tree structured network topologies proposed by Tai et al. [13].

P-LSTM: A model introduces the phrase factor mechanism which combines the feature vectors of the phrase embedding layer and the LSTM hidden layer to extract more exact information from the text proposed by Lu et al. [34].

C-LSTM: A model combining with the strengths of CNN and RNN for sentence representation and text classification proposed by Zhou et al. [72].

LSTM: Long short term memory.

BiLSTM: Bidirectional long short term memory.

4.3. Results

4.3.1. Overall comparison

In this section, our evaluation results are shown on the sentiment classification and question type classification tasks. Moreover, some approach analysis are given.

(a) Sentiment classification

The comparison results for long reviews (RT-2k and IMDB) and short reviews (MR, SST-1, SST-2 and Subj) are presented in Table 2. The experimental results are evaluated by the classification accuracy. The best results are shown in **boldface**. The top 3 approaches are conventional machine learning approaches with hand-crafted features. Other 16 approaches, including our approach, are deep neural network (DNN) approaches, which can automatically extract features from the input data for classifier training without feature engineering. The results of the top 16 approaches are taken from Kim [18,27,34,72].

From Table 2, AC-BiLSTM achieves better results than other methods on the majority of the benchmark datasets. Among the 19 approaches mentioned above, our approach outperforms other baselines on all datasets except SST-1. The results of AC-BiLSTM are 83.2%, 91.8%, 88.3%, 93.0% and 94.0% for MR, IMDB, SST-2, RT-2k and Subj datasets. AC-BiLSTM gives the relative improvements of 2.09%, 0.33%, 0.23%, 3.91% and 0.21% compared to CNN-non-static on MR dataset, P-LSTM on IMDB dataset, CNN-multichannel on SST-2 dataset, NBSVM on RT-2k dataset and P-LSTM on Subj dataset, respectively. It is observed that comparing with four CNN-based methods (DCNN, CNN-static, CNN-non-static and CNN-multichannel), AC-BiLSTM gives better results on the four datasets. Compared to six LSTM-based methods (Multi-task LSTM, Tree-LSTM, P-LSTM, C-LSTM, LSTM and BiLSTM), AC-BiLSTM gives superior performance on the five datasets. AC-BiLSTM outperforms three hand-crafted features based methods (SVM, MNB and NBSVM) and other methods (RAE, MV-RNN, RNTN, Paragraph-Vec and DRNN) on all datasets. For the dataset SST-1, where the data is divided into 5 classes, Tree-LSTM is the only method to arrive at above 50%. But our approach do not differ significantly from the result of Tree-LSTM. It demonstrates that AC-BiLSTM, as an end-to-end model, the results are still promising and comparable with those models that heavily rely on linguistic annotations and knowledge. This indicates that AC-BiLSTM will be more feasible for various scenarios. Simultaneously, it can be seen that the performance of DNN-based methods is better than that of the conventional machine learning approaches.

Table 3
The 6-way question type classification accuracy on TREC. % is omitted and "ACC" means the classification accuracy.

Model	ACC	Reported in
SVM	95.0	Silva et al. [74]
Paragraph-Vec	91.8	Zhao et al. [73]
Ada-CNN	92.4	Zhao et al. [73]
CNN-non-static	93.6	Kim Kim [18]
CNN-multichannel	92.2	Kim [18]
DCNN	93.0	Kalchbrenner [69]
C-LSTM	94.6	Zhou [72]
LSTM	95.3	Our implementation
BiLSTM	95.5	Our implementation
AC-BiLSTM	**97.0**	Our implementation

(b) Question type classification

The prediction accuracy on TREC question classification is reported in Table 3. The SVM classifier uses unigrams, bigrams, wh-word, head word, POS tags, parser, hypernyms, WordNet synsets as engineered features and 60 hand-coded rules. Ada-CNN [73] is a self-adaptive hierarchical sentence model with gating networks and it is added to the baseline models. Other baseline models have been introduced in the Section 4.2.

From Table 3, it can be seen that our approach achieves better results than other baselines on the TREC dataset and the result of AC-BiLSTM is 97.0%. Our approach gives the relative improvements of 1.57% and 3.63% compared to BiLSTM and CNN-non-static on TERC dataset, respectively. Comparing with the CNN-based methods and the LSTM-based methods, AC-BiLSTM gives superior performance on the question type classification dataset. For the TREC dataset, the results of LSTM-based methods are better than these of the CNN-based methods. It shows that the LSTM-based methods are more suitable than the CNN-based methods for this task. As shown above, AC-BiLSTM captures intentions of TREC questions well.

Combined with the results in sentiment classification and question classification, our results consistently outperform the most of the published baseline models. In view of the above discussion it can be concluded that the overall performance of AC-BiLSTM is better than that of the state-of-the-art methods in terms of the classification accuracy.

4.3.2. Effect of each component of AC-BiLSTM

AC-BiLSTM contains three components, namely, the convolutional layer, BiLSTM, the attention mechanism layers. For AC-BiLSTM, it should be proven that all components are useful for the final results. In this section, a set of experiments are to investigate the effect of each component on the performance of AC-BiLSTM.

Table 4
Effect of each component on the performance of AC-BiLSTM. % is omitted, "ACC" means the classification accuracy and "-" indicates no data.

Dataset	A-BiLSTM		AC-LSTM		C-BiLSTM		AC-BiLSTM	
	ACC	Δ	ACC	Δ	ACC	Δ	ACC	Δ
MR	82.7	0.60	81.5	2.09	81.5	2.09	**83.2**	–
IMDB	91.0	0.88	90.8	1.10	90.0	2.00	**91.8**	–
SST-1	48.8	0.20	48.2	1.45	48.5	0.82	**48.9**	–
SST-2	88.0	0.34	87.6	0.80	87.9	0.46	**88.3**	–
RT-2k	90.9	2.31	91.0	2.20	89.7	3.68	**93.0**	–
Subj	93.8	0.21	93.4	0.64	92.5	1.62	**94.0**	–
TREC	96.7	0.31	96.3	0.73	96.8	0.21	**97.0**	–

AC-BiLSTM without the convolutional layer, AC-BiLSTM replacing BiLSTM with LSTM, AC-BiLSTM without the attention mechanism layers and AC-BiLSTM are compared in this section. The relative improvement ratio Δ and the classification accuracy are used as the evaluation metric. The relative improvement ratio Δ calculates as follows:

$$\Delta = (ACC_{AC-BiLSTM} - ACC_{var}) \div ACC_{var} \tag{16}$$

where $ACC_{AC-BiLSTM}$ is the classification accuracy of our approach and ACC_{var} is the classification accuracy of each AC-BiLSTM variant. A-BiLSTM is the classification accuracy of AC-BiLSTM without the convolutional layer. AC-LSTM is the classification accuracy of AC-BiLSTM replacing BiLSTM with LSTM. C-BiLSTM is the classification accuracy of AC-BiLSTM without the attention mechanism layers. The results are presented in Table 4.

From Table 4, it can be seen that the attention mechanism layers and BiLSTM have a powerful influence on the performance of AC-BiLSTM. Among the all architectures mentioned above, AC-BiLSTM obtains the best results. Compared with AC-LSTM and C-BiLSTM, AC-BiLSTM brings the relative improvements of 0.21% to 2.09%. It is observed that the performance of AC-BiLSTM decreases considerably when the attention mechanism layers and BiLSTM are removed. In AC-BiLSTM, the attention mechanism layers can identify the effect of each word for the text and BiLSTM can obtain both preceding and succeeding information. These components effectively improve the classification accuracy of AC-BiLSTM. Compared with A-BiLSTM, AC-BiLSTM brings the relative improvements of 0.21% to 0.88%. It means that the influence of the convolutional layer on our approach is lower than that of other components. But the convolutional layer still helps to improve the classification accuracy. For AC-BiLSTM, the importance of the attention mechanism layers or BiLSTM is higher than the importance of the convolutional layer. It proves that all components are useful for the final results in AC-BiLSTM.

The input of the attention mechanism layers also has an important influence on the classification results. In AC-BiLSTM, the preceding and succeeding contexts are fed to different attention mechanism layers. In this section, a set of experiments are to investigate the effect of the different input of the attention mechanism layers on the performance of AC-BiLSTM. AC-BiLSTM with the single attention mechanism layer and AC-BiLSTM are compared. In AC-BiLSTM with the single attention mechanism layer, the preceding and succeeding contextual features are concatenated together to form the output of BiLSTM. And then the output of BiLSTM is as the input of the attention mechanism layer. The relative improvement ratio Δ and the classification accuracy are used as the evaluation metric. The results are presented in Table 5. A1C-BiLSTM is the classification accuracy of AC-BiLSTM with the single attention mechanism layer.

From Table 5, it can be seen that using the two attention mechanism layers to process the forward and backward information separately is better than using the single attention mechanism layer to process the concatenation of the forward and backward information. AC-BiLSTM gives the relative improvements of 1.09% and 0.94% compared to A1C-BiLSTM on the RT-2k dataset and the TREC dataset, respectively. Except for the IMDB dataset, AC-BiLSTM achieves the better results on other datasets. It proves that using the two attention mechanism layers to process the forward and backward information separately can further improve the performance in AC-BiLSTM.

Table 5
Effect of the different input of the attention mechanism layer on the performance of AC-BiLSTM. % is omitted, "ACC" means the classification accuracy and "-" indicates no data.

Dataset	A1C-BiLSTM		AC-BiLSTM	
	ACC	Δ	ACC	Δ
MR	82.5	0.85	**83.2**	–
IMDB	**91.8**	0	**91.8**	–
SST-1	48.7	0.41	**48.9**	–
SST-2	88.2	0.11	**88.3**	–
RT-2k	92.0	1.09	**93.0**	–
Subj	93.4	0.64	**94.0**	–
TREC	96.1	0.94	**97.0**	–

4.3.3. Tuning of hyperparameters in one dimension convolutional layer

The convolutional layer usually uses the fixed-size convolution filters. It means that there is a fixed-size window sliding from the beginning to the end of a text to produce feature maps, which is equivalent to extracting fixed-size n-gram features. Therefore, it is especially important to choose the appropriate fixed-size convolution window size. In order to verify the impact of the window size of the convolution filters applied in one dimension convolutional layer, a set of experiments are to investigate the effect of the window size. The window size m is as follows: $m = 2, 3, 5, 7$. Except to the window size, all other parameters are kept unchanged. The results are presented in Fig. 5.

As shown in Fig. 5, the accuracy is not significantly influenced by the window size on the most datasets. For IMDB dataset, the window size $m = 2$ gives the relative improvements of 0.98% compared to $m = 3$. For other datasets, the window size $m = 3$ has excellent or comparable performance compared to other window sizes. The results show that when the window size is 3, the classification accuracy is better.

The stride size of the convolutional sliding windows also affects the features extracted by the convolutional layer. In this section, a set of experiments are to investigate the effect of the stride size of the convolutional sliding windows. The stride size s is as follows: $m = 1, 2, 3, 4$. Except to the stride size, all other parameters are kept unchanged. The results are presented in Fig. 6.

From Fig. 6, it can be seen that the classification accuracy of our method is significantly reduced for all datasets when the stride size increases. For the long sentence datasets (IMDB and RT-2k), the performance of our method is less affected by the stride size. But for the short sentence datasets (MR and SST-2), the performance of the approach has declined dramatically. The reason is that increasing the stride size means that the convolutional layer will lose more semantic information. In the short sentences, each word contains relatively more semantic information, and in the long sentences, each word contains relatively less semantic information. Therefore, when the stride size increases, the short sentences will lose more semantic information than the long sentences. In short, reducing the stride size can achieve better results.

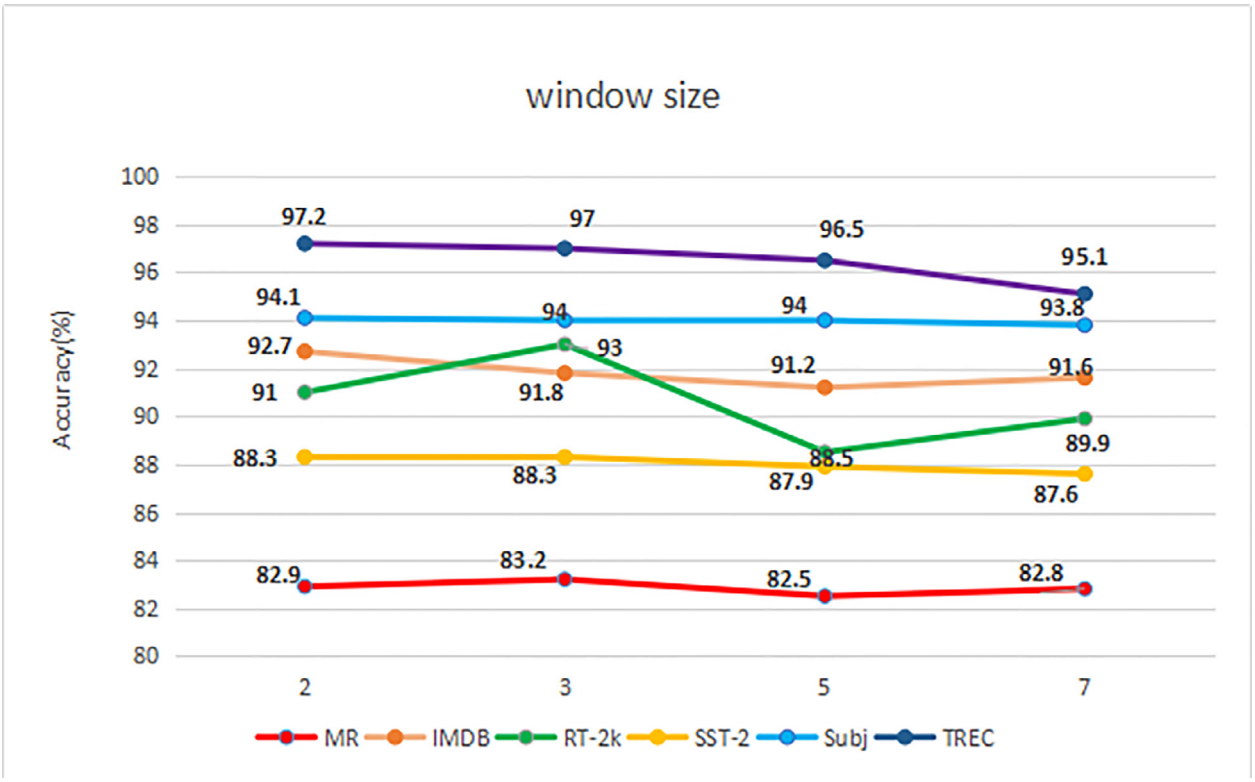


Fig. 5. Accuracy with different window sizes.

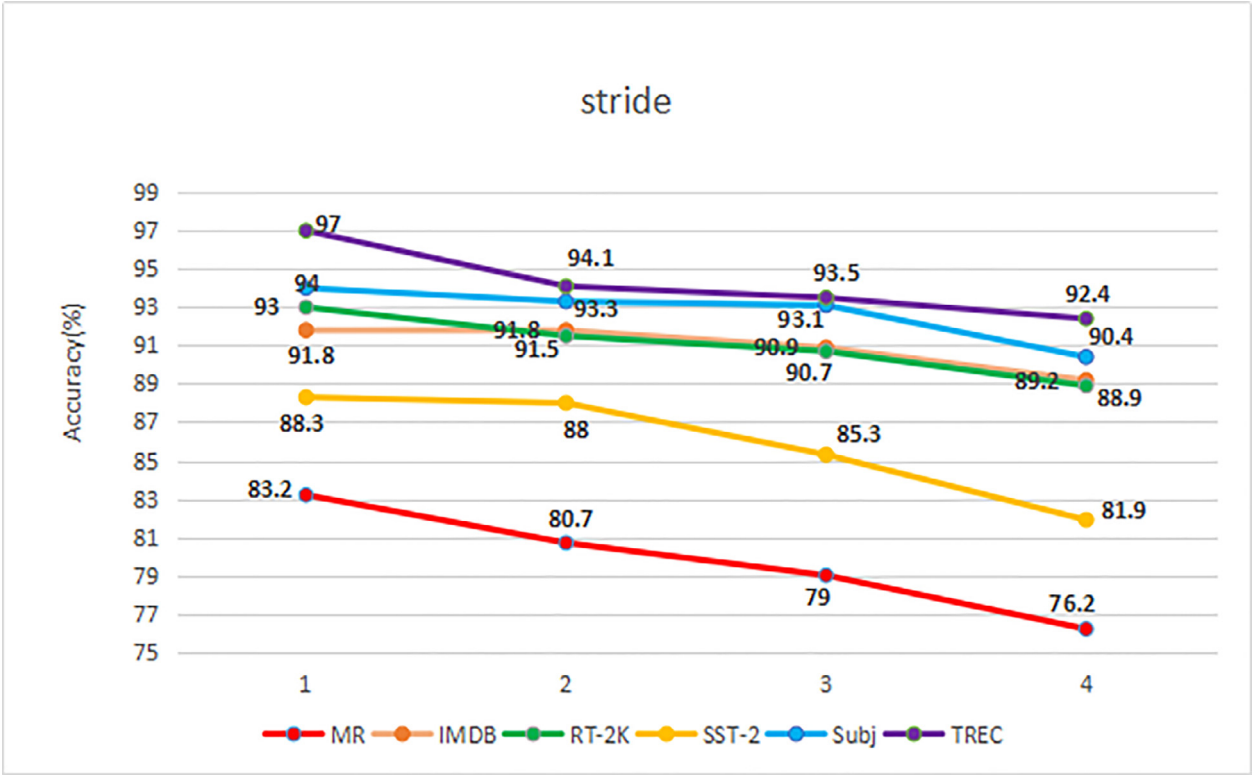


Fig. 6. Accuracy with different stride sizes.

Table 6
Experimental results of different word embedding vector generation methods on binary classification datasets. % is omitted.

Dataset	Embedding variant	Recall	Precision	F1-score
MR	Pre-trained	83.48	82.86	83.17
	Random	78.80	77.20	78.00
IMDB	Pre-trained	92.09	88.91	90.47
	Random	86.45	89.17	87.78
RT-2k	Pre-trained	89.90	95.70	92.71
	Random	90.91	85.71	88.24
SST-2	Pre-trained	87.62	87.26	87.44
	Random	87.77	85.25	86.49
Subj	Pre-trained	93.76	94.14	93.95
	Random	91.65	92.62	92.13

4.3.4. Comparison on word embedding vector variants

In word2vec, there are four ways to generate word embedding vector. The commonly used methods to generate word embedding vector are mainly the pre-training method and the random method. Pre-trained word embedding vector means the model with the pre-trained vectors from word2vec. Random word embedding vector means the model where all words are randomly initialized and then modified during training. Generally, the different methods to generate word embedding vector have different effects on the classification performance. In this section, a set of experiments are to investigate the effect of the different methods to generate word embedding vector on the performance of AC-BiLSTM. In order to disentangle the effect of the above word embedding variations versus other random factors, all parameters are kept unchanged. In this section, recall, precision and F1-score are used to measure binary classification performance. Recall is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Precision is the fraction of relevant instances among the retrieved instances. For a classifier dedicated to binary classification, F1-score is an important indicator and it is the combination of the recall and precision. F1-score calculates as follows:

$$F1-score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{17}$$

The higher the three indicators are, the better the binary classification performance and robustness should be. Table 6 shows the performance comparison results on the binary classification datasets.

From Table 6, AC-BiLSTM using pre-trained word embedding vector achieves better results than AC-BiLSTM using random word embedding vector on all datasets. For F1-score, AC-BiLSTM using pre-trained word embedding vector can get higher results than AC-BiLSTM using random word embedding vector. For the MR and Subj datasets, pre-trained word embedding vector is superior to random word embedding vector in all indicators. For the RT-2k and SST-2 datasets, pre-trained word embedding vector is superior to random word embedding vector in precision and F1-score while random word embedding vector performs better in recall. For IMDB, random word embedding vector only performs better in precision while pre-trained word embedding vector performs better in other indicators. Compared to random word embedding vector, pre-trained word embedding vector has obvious advantages.

The F1-score comparison results of the five binary classification datasets are depicted in Fig. 7. From Fig. 7, it can be seen that pre-trained word embedding vector can obtain better classification performance compared to random word embedding vector. For all datasets, the F1-scores of pre-trained word embedding vector are higher than these of random word embedding vector. The results suggest that the pre-trained word vectors are good universal feature extractors and can be utilized across datasets.

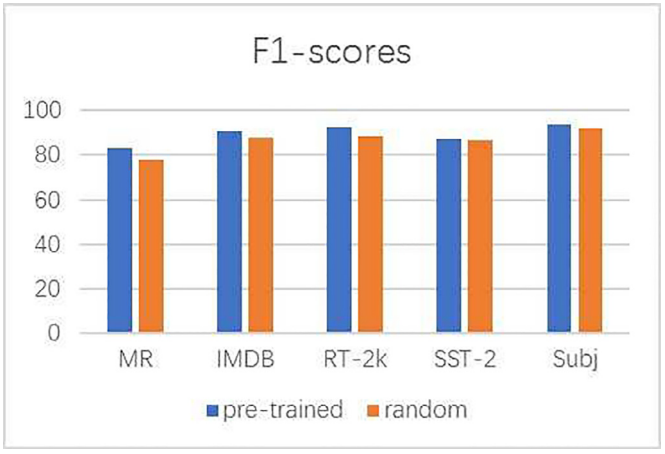


Fig. 7. Statistics and comparison of word embedding vector variants.

4.4. Discussions

For AC-BiLSTM, the purpose of the convolutional layer is to preprocess the input text data. Owing to the capability of capturing local correlations of spatial or temporal structures, the convolutional layer performs excellently in extracting n-gram features at different positions of the text through the convolutional filters from the word vectors. In addition, the convolutional layer reduces the parameters of the network. Compared to LSTM, BiLSTM can accesses both the preceding and succeeding contextual information. Hence, BiLSTM can more effectively learn the context of each word in the text. Attention mechanism is mainly to identify the influence of each word on the sentence. It assigns the attention weights to each word and can capture the important components of the sentence semantics. The combination of these methods makes the understanding of sentence semantics more accurate and improves the classification ability of AC-BiLSTM. Experiments show that the convolutional layer, BiLSTM and attention mechanism have an important influence on the performance of AC-BiLSTM. It is worthwhile to note that BiLSTM and attention mechanism have greater effects than the convolutional layer on the classification accuracy. For the convolutional layer, the convolution window size and the stride size also affect the performance of AC-BiLSTM. Experiments also show that when the window size is 3 and the stride size is 1, AC-BiLSTM can achieve the best results.

For text classification, the methods to generate word embedding vector can affect the classification accuracy. Compared to pre-trained embedding word vector, random embedding word vector requires to train more parameters and it causes relatively lower classification accuracy in limited iterations. The experiments show that pre-trained embedding word vector can achieves better results than random word embedding vector. Hence, the method to generate pre-trained embedding word vector is more suitable for AC-BiLSTM.

All experiment results indicate the combination of the convolutional layer, BiLSTM and attention mechanism remarkably improves text classification accuracy. For the most of the benchmark datasets, AC-BiLSTM can obtain better results than other baseline models. It shows that AC-BiLSTM has the better classification ability and our proposed AC-BiLSTM performs better than some state-of-the-art DNNs.

5. Conclusions

For text classification, feature extraction and the design of classifier are very important. LSTM has shown better performance on

many real-world and benchmark text classification problems. However, it is still difficult to understand the semantics and the classification accuracy still needs to be improved. In order to solve these problems, this paper presents an improved LSTM method, namely AC-BiLSTM, in which the convolutional layer, BiLSTM and attention mechanism are used to enhance semantic understanding and improve the classification accuracy. Experiments are conducted on seven benchmark datasets to evaluate the performance of our presented approach. The experimental results indicate that AC-BiLSTM can understand semantics more accurately and enhance the performance of LSTM in terms of the quality of the final results.

Comparisons with some state-of-the-art baseline methods, it demonstrates that the new method is more effective and efficient in terms of the classification quality in most cases.

Future work focuses on the research of attention mechanism and the design of network architecture. In addition, the new methods are also applied to the field of machine reading comprehension. Future works mainly includes the following parts: (1) using other attention mechanisms to further improve our approach; (2) investigating the effect of attention mechanism on the performance of our approach; (3) designing the new attention mechanism and the network architecture; (4) applying our approach to the practical applications; (5) applying the designed attention mechanism and the network architecture to machine reading comprehension.

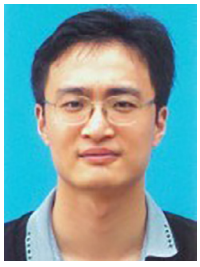
Acknowledgement

The work described in this paper was support by National Natural Science Foundation of China Foundation No. 61300127. Any conclusions or recommendations stated here are those of the authors and do not necessarily reflect official positions of NSFC.

References

[1] A. Watanabe, R. Sasano, H. Takamura, M. Okumura, Generating personalized snippets for web page recommender systems, Trans. Jpn. Soc. Artif. Intell. 31 (5) (2016).
[2] T.A. Almeida, T.P. Silva, I. Santos, J.M. Gomez Hidalgo, Text normalization and semantic indexing to enhance instant massaging and SMS spam filtering, Knowl. Based Syst. 108 (September 15, 2016) (2016) 25–32.
[3] B. Liu, Sentiment Analysis: Mining Opinions, Sentiments, and Emotions, Cambridge University Press, Cambridge, England, 2015.
[4] L.H. Lee, D. Isa, W.O. Choo, W.Y. Chue, High relevance keyword extraction facility for Bayesian text classification on different domains of varying characteristic, Expert Syst. Appl. 39 (1) (2012) 1147–1155.
[5] J. Lei, T. Jin, Hierarchical text classification based on bp neural network, J. Comput. Inf. Syst. 5 (2) (2009) 581–590.
[6] V.N. Phu, V.T.N. Tran, V.T.N. Chau, N.D. Dat, K.L.D. Duy, A decision tree using id3 algorithm for english semantic analysis, Int. J. Speech Technol. 20 (3) (2017) 593–613.
[7] P.D. Turney, Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL), Association for Computational Linguistics (ACL), Philadelphia, Pennsylvania, United states, 2002, pp. 417–424.
[8] B. Pang, 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 (ACL), Association for Computational Linguistics (ACL), Barcelona, Spain, 2004, pp. 271–278.
[9] B. Liu, Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies, Morgan and Claypool Publishers, San Rafael, CA, United states, 2012.
[10] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Netw. 61 (January 01, 2015) (2015) 85–117.
[11] V. Campos, B. Jou, X. Giro-i Nieto, From pixels to sentiment: Fine-tuning cnns for visual sentiment prediction, Image Vis. Comput. 65 (September 2017) (2017) 15–22.
[12] L. Brocki, K. Marasek, Deep belief neural networks and bidirectional long-short term memory hybrid for speech recognition, Arch. Acoust. 40 (2) (2015) 191–195.
[13] K.S. Tai, R. Socher, C.D. Manning, Improved semantic representations from tree-structured long short-term memory networks, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing ACL-IJCNLP 2015, Association for Computational Linguistics (ACL), Beijing, China, 2015, pp. 1556–1566.
[14] B. Pang, L. Lee, Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales, in: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics (ACL), Ann Arbor, MI, United states, 2005, pp. 115–124.
[15] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (August 2011) (2011) 2493–2537.
[16] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012 (NIPS'2012), Lake Tahoe, NV, United states, 2012, pp. 1097–1105.
[17] K.-i. Funahashi, Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, Neural Netw. 6 (6) (1993) 801–806.
[18] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (ACL), Doha, Qatar, 2014, pp. 1746–1751.
[19] S. Liao, J. Wang, R. Yu, K. Sato, Z. Cheng, CNN for situations understanding based on sentiment analysis of twitter data, in: Proceedings of the 8th International Conference on Advances in Information Technology, Elsevier B.V., Macau, China, 2016, pp. 376–381.
[20] W. Cao, A. Song, J. Hu, Stacked residual recurrent neural network with word weight for text classification, IAENG Int. J. Comput. Sci. 44 (3) (2017) 277–284.
[21] Y. Zhang, M.J. Er, R. Venkatesan, N. Wang, M. Pratama, Sentiment classification using comprehensive attention recurrent models, in: Proceedings of the International Joint Conference on Neural Networks, IEEE, Vancouver, BC, Canada, 2016, pp. 1562–1569.
[22] L. Wang, Z. Wang, S. Liu, An effective multivariate time series classification approach using echo state network and adaptive differential evolution algorithm, Expert Syst. Appl. 43 (January 1, 2016) (2016) 237–249.
[23] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[24] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional Lstm and other neural network architectures, Neural Netw. 18 (5-6) (2005) 602–610.
[25] W. Liu, P. Liu, Y. Yang, Y. Gao, J. Yi, An attention-based syntax-tree and tree-LSTM model for sentence summarization, Int. J. Performab. Eng. 13 (5) (2017) 775–782.
[26] J. Nowak, A. Taspinar, R. Scherer, LSTM recurrent neural networks for short text and sentiment classification, in: Proceedings of the 16th International Conference on Artificial Intelligence and Soft Computing, Springer Verlag, Zakopane, Poland, 2017, pp. 553–562.
[27] T. Chen, R. Xu, Y. He, X. Wang, Improving sentiment analysis via sentence type classification using biLSTM-Crf and CNN, Expert Syst. Appl. 72 (April 15, 2017) (2017) 221–230.
[28] X. Niu, Y. Hou, P. Wang, Bi-directional LSTM with quantum attention mechanism for sentence modeling, in: Proceedings of the 24th International Conference on Neural Information Processing, Springer Verlag, Guangzhou, China, 2017, pp. 178–188.
[29] M.-T. Luong, H. Pham, C.D. Manning, Effective approaches to attention-based neural machine translation, in: Proceedings of Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1412–1421.
[30] Z. Zhang, Y. Zou, C. Gan, Textual sentiment analysis via three different attention convolutional neural networks and cross-modality consistent regression, Neurocomputing 275 (31 January 2018) (2018) 1407–1415.
[31] A. Kolawole John, L. Di Caro, L. Robaldo, G. Boella, Textual inference with tree-structured LSTM, in: Proceedings of the 28th Benelux Conference on Artificial Intelligence, Revised Selected Papers, Springer Verlag, Amsterdam, Netherlands, 2016, pp. 17–31.
[32] S.K. Sonderby, C.K. Sonderby, H. Nielsen, O. Winther, Convolutional LSTM networks for subcellular localization of proteins, in: Proceedings of the 2nd International Conference on Algorithms for Computational Biology, Springer Verlag, Mexico City, Mexico, 2015, pp. 68–80.
[33] J. Wang, L.-C. Yu, K.R. Lai, X. Zhang, Dimensional sentiment analysis using a regional CNN-LSTM model, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Short Papers, Association for Computational Linguistics (ACL), Berlin, Germany, 2016, pp. 225–230.
[34] C. Lu, H. Huang, P. Jian, D. Wang, Y.-D. Guo, A p-LSTM neural network for sentiment classification, in: Proceedings of the 21st Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer Verlag, Jeju, Korea, Republic of, 2017, pp. 524–533.
[35] J.C. Nunez, R. Cabido, J.J. Pantrigo, A.S. Montemayor, J.F. Velez, Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition, Pattern Recognit. 76 (April 2018) (2018) 80–94.
[36] T. Le, G. Bui, Y. Duan, A multi-view recurrent neural network for 3d mesh segmentation, Comput. Graph. (Pergamon) 66 (August 2017) (2017) 103–112.
[37] K. Xu, J.L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R.S. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, International Machine Learning Society (IMLS), Lile, France, 2015, pp. 2048–2057.

[38] M.-T. Luong, H. Pham, C.D. Manning, Effective approaches to attention-based neural machine translation, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Association for Computational Linguistics (ACL), Lisbon, Portugal, 2015, pp. 1412–1421.
[39] Z. Lin, M. Feng, C.N. dos Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured self-attentive sentence embedding, CoRR abs/1703.03130 (2017).
[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the 31st Annual Conference on Neural Information Processing Systems, NIPS 2017, Neural information processing systems foundation, Long Beach, CA, United states, 2017, pp. 5999–6009.
[41] T. Shen, T. Zhou, G. Long, J. Jiang, C. Zhang, Bi-directional block self-attention for fast and memory-efficient sequence modeling, CoRR abs/1804.00857 (2018).
[42] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical attention networks for document classification, in: Proceedings of the 15th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016, Association for Computational Linguistics (ACL), San Diego, CA, United states, 2016, pp. 1480–1489.
[43] Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu, G. Hu, Attention-over-attention neural networks for reading comprehension, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Association for Computational Linguistics (ACL), Vancouver, BC, Canada, 2017, pp. 593–602.
[44] H. Li, M.R. Min, Y. Ge, A. Kadav, A context-aware attention network for interactive question answering, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017, Association for Computing Machinery, Halifax, NS, Canada, 2017, pp. 927–935.
[45] R. Paulus, C. Xiong, R. Socher, A deep reinforced model for abstractive summarization, CoRR abs/1705.04304 (2017).
[46] H. Huang, C. Zhu, Y. Shen, W. Chen, Fusionnet: Fusing via fully-aware attention with application to machine comprehension, CoRR abs/1711.07341 (2017).
[47] M.J. Seo, A. Kembhavi, A. Farhadi, H. Hajishirzi, Bidirectional attention flow for machine comprehension, CoRR abs/1611.01603 (2016).
[48] M. Daniluk, T. Rocktäschel, J. Welbl, S. Riedel, Frustratingly short attention spans in neural language modeling, CoRR abs/1702.04521 (2017).
[49] A.P. Parikh, O. Täckström, D. Das, J. Uszkoreit, A decomposable attention model for natural language inference, CoRR abs/1606.01933 (2016).
[50] M. Yang, W. Tu, J. Wang, F. Xu, X. Chen, Attention-based LSTM for target-dependent sentiment classification, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI press, San Francisco, CA, United states, 2017, pp. 5013–5014.
[51] X. Wei, H. Lin, L. Yang, Y. Yu, A convolution-LSTM-based deep neural network for cross-domain MOOC forum post classification, Inf. (Switz.) 8 (3) (2017).
[52] Y. Luo, Recurrent neural networks for classifying relations in clinical notes, J. Biomed. Inf. 72 (August 2017) (2017) 85–95.
[53] F. Hu, L. Li, Z.-L. Zhang, J.-Y. Wang, X.-F. Xu, Emphasizing essential words for sentiment classification based on recurrent neural networks, J. Comput. Sci. Technol. 32 (4) (2017) 785–795.
[54] M. Huang, Q. Qian, X. Zhu, Encoding syntactic knowledge in neural networks for sentiment classification, ACM Trans. Inf. Syst. 35 (3) (2017).
[55] D. Wu, M. Chi, Long short-term memory with quadratic connections in recursive neural networks for representing compositional semantics, IEEE Access 5 (2017) (2017) 16077–16083.
[56] D. Tang, B. Qin, T. Liu, Document modeling with gated recurrent neural network for sentiment classification, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (ACL), Lisbon, Portugal, 2015, pp. 1422–1432.
[57] Y. Wang, S. Feng, D. Wang, Y. Zhang, G. Yu, Context-aware chinese microblog sentiment classification with bidirectional LSTM, in: Proceedings of the 18th Asia-Pacific Web Conference on Web Technologies and Applications, Springer Verlag, Suzhou, China, 2016, pp. 594–606.
[58] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, CoRR abs/1301.3781 (2013).
[59] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the 3rd International Conference for Learning Representations, Springer Verlag, San Diego, CA, United states, 2015, pp. 1–15.
[60] S. Lai, L. Xu, K. Liu, J. Zhao, Recurrent convolutional neural networks for text classification, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence, AAAI 2015 and the 27th Innovative Applications of Artificial Intelligence Conference, IAAI 2015, AI Access Foundation, Austin, TX, United states, 2015, pp. 2267–2273.
[61] A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics (ACL), Portland, OR, United states, 2011, pp. 142–150.
[62] R. Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (ACL), Seattle, WA, United states, 2013, pp. 1631–1642.
[63] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (6) (2003) 1137–1155.
[64] Y. Liu, J.-W. Bi, Z.-P. Fan, A method for multi-class sentiment classification based on an improved one-vs-one (ovo) strategy and the support vector machine (svm) algorithm, Inf. Sci. 394-395 (July 1, 2017) (2017) 38–52.
[65] S. Wang, C.D. Manning, Baselines and bigrams: Simple, good sentiment and topic classification, in: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics (ACL), Jeju Island, Korea, Republic of, 2012, pp. 90–94.
[66] R. Socher, J. Pennington, E.H. Huang, A.Y. Ng, C.D. Manning, Semi-supervised recursive autoencoders for predicting sentiment distributions, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (ACL), Edinburgh, United kingdom, 2011, pp. 151–161.
[67] R. Socher, B. Huval, C.D. Manning, A.Y. Ng, Semantic compositionality through recursive matrix-vector spaces, in: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Association for Computational Linguistics (ACL), Jeju Island, Korea, Republic of, 2012, pp. 1201–1211.
[68] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31st International Conference on Machine Learning, International Machine Learning Society (IMLS), Beijing, China, 2014, pp. 2931–2939.
[69] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics (ACL), Baltimore, MD, United states, 2014, pp. 655–665.
[70] O. Irsoy, C. Cardie, Deep recursive neural networks for compositionality in language, in: Proceedings of the 28th Annual Conference on Neural Information Processing Systems, Neural information processing systems foundation, Montreal, QC, Canada, 2014, pp. 2096–2104.
[71] P. Liu, X. Qiu, H. Xuanjing, Recurrent neural network for text classification with multi-task learning, in: Proceedings of the 25th International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence, New York, NY, United states, 2016, pp. 2873–2879.
[72] C. Zhou, C. Sun, Z. Liu, F.C.M. Lau, A C-LSTM neural network for text classification, Comput. Sci. 1 (4) (2015) 39–44.
[73] H. Zhao, Z. Lu, P. Poupart, Self-adaptive hierarchical sentence model, in: Proceedings of the 24th International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence, Buenos Aires, Argentina, 2015, pp. 4069–4076.
[74] J. Silva, L. Coheur, A.C. Mendes, A. Wichert, From symbolic to sub-symbolic information in question classification, Artif. Intell. Rev. 35 (2) (2011) 137–154.



Gang Liu received the Ph.D. degree in computer software and theory from State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China, in 2012. He is currently an associate professor with the School of Computer Science, Hubei University of Technology, Wuhan, China. He has published more than 20 international journal/conference papers. His current research interests include evolutionary computation, deep learning technology, image processing and natural language processing.

Jiabao Guo is currently a postgraduate student in the School of Computer Science, Hubei University of Technology, Wuhan, China. Her current research interests include evolutionary computation, deep learning technology and Natural Language Processing.