Applied Intelligence (2018) 48:3797–3806
https://doi.org/10.1007/s10489-018-1176-4

Recurrent networks with attention and convolutional networks for sentence representation and classification

Tengfei Liu[1] · Shuangyuan Yu[1] · Baomin Xu[1] · Hongfeng Yin[2]

Published online: 27 April 2018

Abstract
In this paper, we propose a bi-attention, a multi-layer attention and an attention mechanism and convolution neural network based text representation and classification model (ACNN). The bi-attention have two attention mechanism to learn two context vectors, forward RNN with attention to learn forward context vector $\overrightarrow{\mathbf{c}}$ and backward RNN with attention to learn backward context vector $\overleftarrow{\mathbf{c}}$, and then concatenation $\overrightarrow{\mathbf{c}}$ and $\overleftarrow{\mathbf{c}}$ to get context vector $\mathbf{c}$. The multi-layer attention is the stack of the bi-attention. In the ACNN, the context vector $\mathbf{c}$ is obtained by the bi-attention, then the convolution operation is performed on the context vector $\mathbf{c}$, and the max-pooling operation is used to reduce the dimension. After max-pooling operation the text is converted to low-dimensional sentence vector $\mathbf{m}$. Finally, the Softmax classifier be used for text classification. We test our model on 8 benchmarks text classification datasets, and our model achieved a better or the same performance compare with the state-of-the-art methods.

Keywords Natural language processing · Deep neural networks · Attention mechanism · Representation learning · Text classification

1 Introduction

Text classification, with the goal is to assign labels to text, is one of the classic tasks in NLP [39]. And Text classification has a wide range of applications, such as spam detection [30], sentiment classification [22, 29] and topic labeling [36] and so on.

But, a better text representation is the key to get a better performance for natural language processing tasks such as text classification. The traditional text representation is the one-hot representation, which not only lost the context information of a text, but also is sparse and faced with the curse of dimensionality.

Recently, word-level representation, distributed distributions, based on neural network models [3, 5, 11, 23–25], have become increasingly popular. The distribution representation based on the neural network model is called word vector, word embedding or distribution representation. The neural network word embedded technique models the context and the relationship between the context and the target word by neural network technology, which can map the words to low-dimensional vector space [18].

For text representation, with the performance improvement of hardware and the increase in the amount of data, the deep learning methods are more and more popular, such as convolutional neural networks [13, 14], recurrent neural networks [18] and attention mechanism [39] to learn text representations for text classification, and have a better performance. All these have proved that the text classification method based on the neural networks is quite effective [6, 12, 14, 21, 34, 38].

In this paper, we propose a text representation and classification model (ACNN) that combines attention mechanism and convolution network. In this model, we use the bi-attention to learn two context vectors. The bi-attention can provide the context vector $\mathbf{c}$ for the followed convolution layer, so that the convolution layer can be targeted for feature extraction; and the convolution layer can

✉ Tengfei Liu
dugufei@bjtu.edu.cn

Shuangyuan Yu
shyyu@bjtu.edu.cn

Baomin Xu
bmxu@bjtu.edu.cn

1 School of Computer and Information Technology, Beijing Jiaotong University, Beijing, 100044, China

2 Department of Computer Science, Beijing Jiaotong University Haibin College, Huanghua, 061199, China

extract the feature of the context vector **c** and convert the text into a low dimension feature vector **m**. Finally, the Softmax classifier can be used for text classification. We test our model on 8 benchmarks text classification datasets, and our model achieved a better or the same performance compared with the state-of-the-art methods.

2 Related work

Text classification is one of the basic tasks of natural language processing. There are many researchers exploring various ways to improve the performance of the classification. Hill, Cho, and Korhonen 2016 proposed learning distributed representations of sentences from unlabelled data [8]. Conneau et al. 2017 indicate the suitability of natural language inference for transfer learning to other NLP tasks [6]. Le and Mikolov 2014 proposed to learn a Paragraph Vector for classification [19]. Illinois-LH system [17], Tree-LSTM [33] and AdaSent [41] are all the state-of-the-art methods for text classification.

Arora, Liang, and Ma proposed an unsupervised sentence embedding methods [1]. Lin et al. 2017 proposed a structure for sentence embedding, too [21]. Lai et al. 2015 proposed a recurrent convolutional neural networks for text classification [18]. Zhang, Zhao, and LeCun 2015 use a character-level convolutional networks for text classification [40]. Kim 2014 apply a word-level convolutional neural networks for sentence classification [14]. Socher et al. 2013 proposed a deep recursive model [31]. Dai and Le 2015 use semi-supervised to improve sequence learning [7]. Kalchbrenner, Grefenstette, and Blunsom 2014 apply a convolutional neural network for modelling sentences [13]. Kiros et al. 2015 proposed SkipThouht [16] and Wang and Manning 2012; 2013 use the SVM and F-Dropout etc [35, 36]. All the state-of-the-art methods have proved that the text classification method based on the neural networks is quite effective.

The model proposed in this paper combined attention mechanisms and convolutional neural networks to learn a sentence vector for classification. The attention mechanism was first proposed by Mnih et al. in the computer vision [26]. Bahdanau, Cho, and Bengio 2014 applied it to neural machine translation [2]. Yang et al. 2016 proposed to use the hierarchical attention model for document classification [39], and achieve a well performance.

3 Background

In natural language processing, most of the data are in the form of sequence data. However, the original neural network is not suitable for processing data in sequence form. Compared with the original neural network, the recurrent neural network(RNN) allows the information to be persist, which makes it have achieved success in speech recognition, language model, machine translation and other tasks. However, the RNN is poorly performing for long-dependent problems.

Long short-term memory(LSTM) proposed by [9], using the gated mechanism to solve the problem of poor performance of RNN on long-dependent problems. Gated Recurrent Unit(GRU) proposed by [4] changes the gated mechanism of the LSTM, making the training time of the model optimized. Bahdanau et al. [2] proposed to apply the attention mechanism to the end-to-end neural machine translation, and get a better performance.

In this paper, we use the attention mechanism and the convolution neural network for text classification, and get a better performance. In the following subsection, we will give a introduction to the bidirectional RNN, attention mechanism, bi-attention and multi-layer attention.

3.1 BiRNN

Bidirectional RNN (BiRNN), Bidirectional LSTM (BiLSTM) and Bidirectional GRU (BiGRU) are what we need for experiment, here, we only have a brief recall of BiRNN.

Let $(\mathbf{x}_1, \cdots, \mathbf{x}_T)$ as the input sequence, The hidden state $\mathbf{h}_t$ of the RNN at the step t calculated as (1).

$$\mathbf{h}_t = f\left(\mathbf{U}\mathbf{x}_t + \mathbf{W}\mathbf{h}_{t-1} + \mathbf{b}\right) \tag{1}$$

where, $f(\cdot)$ is an non-linear activation function, such as: sigmoid, tanh, etc., and the trainable parameters U, W and b are the same for each step of the RNN.

However for the BiRNN, it contains a forward RNN $\overrightarrow{f}$ and a backward RNN $\overleftarrow{f}$, as shown in (2) and (3) respectively.

$$\overrightarrow{\mathbf{h}}_t = \overrightarrow{f}\left(\overrightarrow{\mathbf{U}}\mathbf{x}_t + \overrightarrow{\mathbf{W}}\overrightarrow{\mathbf{h}}_{t-1} + \overrightarrow{\mathbf{b}}\right) \tag{2}$$

$$\overleftarrow{\mathbf{h}}_t = \overleftarrow{f}\left(\overleftarrow{\mathbf{U}}\mathbf{x}_t + \overleftarrow{\mathbf{W}}\overleftarrow{\mathbf{h}}_{t+1} + \overleftarrow{\mathbf{b}}\right) \tag{3}$$

where, $\overrightarrow{f}(\cdot)$ and $\overleftarrow{f}(\cdot)$ are the corresponding non-linear activation function, $\overrightarrow{\mathbf{U}}$, $\overrightarrow{\mathbf{W}}$, $\overrightarrow{\mathbf{b}}$, $\overleftarrow{\mathbf{U}}$, $\overleftarrow{\mathbf{W}}$ and $\overleftarrow{\mathbf{b}}$ are the trainable parameters.

The forward RNN $\overrightarrow{f}$ reads the input sequence from $\mathbf{x}_1$ to $\mathbf{x}_T$ and calculates a sequence of forward hidden states $\left(\overrightarrow{\mathbf{h}}_1, \cdots, \overrightarrow{\mathbf{h}}_T\right)$. And the backward RNN $\overleftarrow{f}$ reads the sequence from $\mathbf{x}_T$ to $\mathbf{x}_1$ and get a backward hidden states sequence $\left(\overleftarrow{\mathbf{h}}_1, \cdots, \overleftarrow{\mathbf{h}}_T\right)$.

Finally, we get the hidden state $\mathbf{h}_t$ of the BiRNN at step t by concatenating the forward hidden state $\overrightarrow{\mathbf{h}}_t$ and the backward hidden state $\overleftarrow{\mathbf{h}}_t$, as shown in (4).

$$\mathbf{h}_t = \left[\overrightarrow{\mathbf{h}}_t^\top ; \overleftarrow{\mathbf{h}}_t^\top\right]^\top \tag{4}$$

3.2 Attention mechanism

Mnih et al. [26] proposed to apply the attention mechanism to the computer vision. In NLP, Bahdanau et al. [2] first applied the attention mechanism to the neural machine translation system, and achieved a well performance.

Figure 1 show an architecture of a RNN with the attention mechanism. Where $\mathbf{x}_1, \cdots, \mathbf{x}_T$ is the input sequence, $\mathbf{c}_t$ is the output(context vector) of the attention model part corresponding to time t, calculated as follows:

$$\mathbf{c}_t = \sum_{k=1}^{T} \alpha_t^k \mathbf{h}_k \tag{5}$$

It can be seen from (5) that $\mathbf{c}_t$ is the weighted sum of the hidden state $(\mathbf{h}_1, \cdots, \mathbf{h}_T)$ of RNN, where α_t^k is the weight of the hidden state $\mathbf{h}_k$, and α_t^k is calculated as follows

$$\alpha_t^k = \frac{\exp\left(\hat{\alpha}_t^k\right)}{\sum_{j=1}^{T_x} \exp\left(\hat{\alpha}_t^j\right)} \tag{6}$$

Where $\hat{\alpha}_t^k$ is jointly learned with other part of the model.

From (6) we know that $\left(\alpha_t^1, \cdots, \alpha_t^T\right)$ is obtained by softmax normalization the trainable weight $\left(\hat{\alpha}_t^1, \cdots, \hat{\alpha}_t^T\right)$.

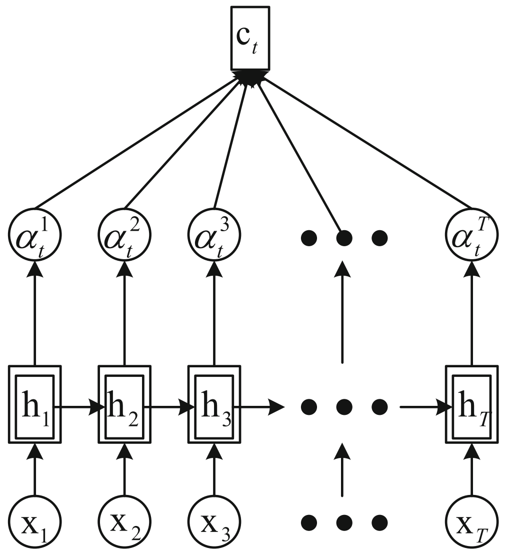

Fig. 1 The architecture of a RNN with the attention mechanism

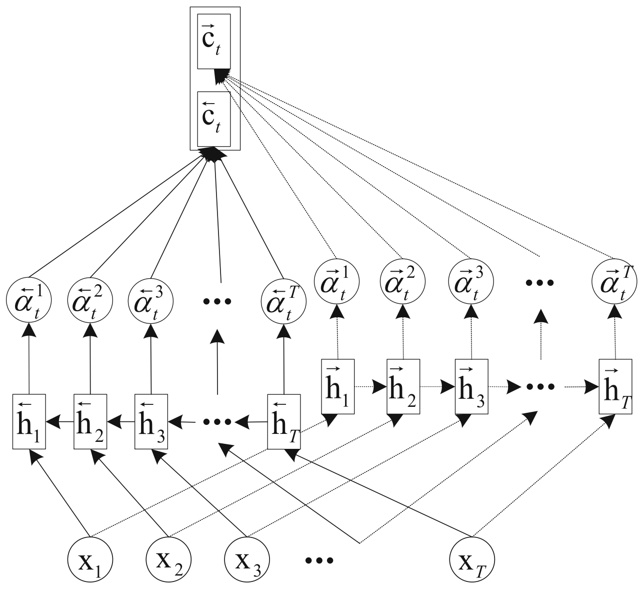

Fig. 2 The architecture of Bi-Attention

3.3 Bi-attention

The bi-attention has two attention mechanism to learn two context vectors, forward RNN with attention to learn forward context vector $\overrightarrow{\mathbf{c}}$ and backward RNN with attention to learn backward context vector $\overleftarrow{\mathbf{c}}$, and then concatenation $\overrightarrow{\mathbf{c}}$ and $\overleftarrow{\mathbf{c}}$ to get context vector $\mathbf{c}$.

Figure 2 shown is the architecture of bi-attention. For the forward RNN, the forward weight $\left(\overrightarrow{\alpha}_t^1, \cdots, \overrightarrow{\alpha}_t^T\right)$ learned for forward hidden state $\left(\overrightarrow{\mathbf{h}}_1, \cdots, \overrightarrow{\mathbf{h}}_T\right)$ to get the t-th forward context vector $\overrightarrow{\mathbf{c}}_t$. For the backward RNN, the backward weight $\left(\overleftarrow{\alpha}_t^1, \cdots, \overleftarrow{\alpha}_t^T\right)$ learned for backward hidden state $\left(\overleftarrow{\mathbf{h}}_1, \cdots, \overleftarrow{\mathbf{h}}_T\right)$ to get the t-th backward context vector $\overleftarrow{\mathbf{c}}_t$. Then concatenation the $\overrightarrow{\mathbf{c}}_t$ and $\overleftarrow{\mathbf{c}}_t$ to get the t-th context vector $\mathbf{c}_t$.

3.4 Multi-layer attention

The multi-layer attention is the stack of the bi-attention.

Figure 3 shown is the architecture of two-layer attention. The inputs of the forward RNN in the second layer is the forward context vector $\overrightarrow{\mathbf{c}}_t^1$ was calculated by the forward weight $\left({}^1\overrightarrow{\alpha}_t^1, \cdots, {}^1\overrightarrow{\alpha}_t^T\right)$ and the forward hidden state $\left(\overrightarrow{\mathbf{h}}_1^1, \cdots, \overrightarrow{\mathbf{h}}_T^1\right)$ of the first layer, the inputs of the backward RNN in the second layer is the backward context vector $\overleftarrow{\mathbf{c}}_t^1$ was calculated by the backward weight $\left({}^1\overleftarrow{\alpha}_t^1, \cdots, {}^1\overleftarrow{\alpha}_t^T\right)$ and the backward hidden state $\left(\overleftarrow{\mathbf{h}}_1^1, \cdots, \overleftarrow{\mathbf{h}}_T^1\right)$ of the first layer. The second layer calculated the forward context vector $\overrightarrow{\mathbf{c}}_t$ by the forward weight $\left({}^2\overrightarrow{\alpha}_t^1, \cdots, {}^2\overrightarrow{\alpha}_t^T\right)$ and the forward

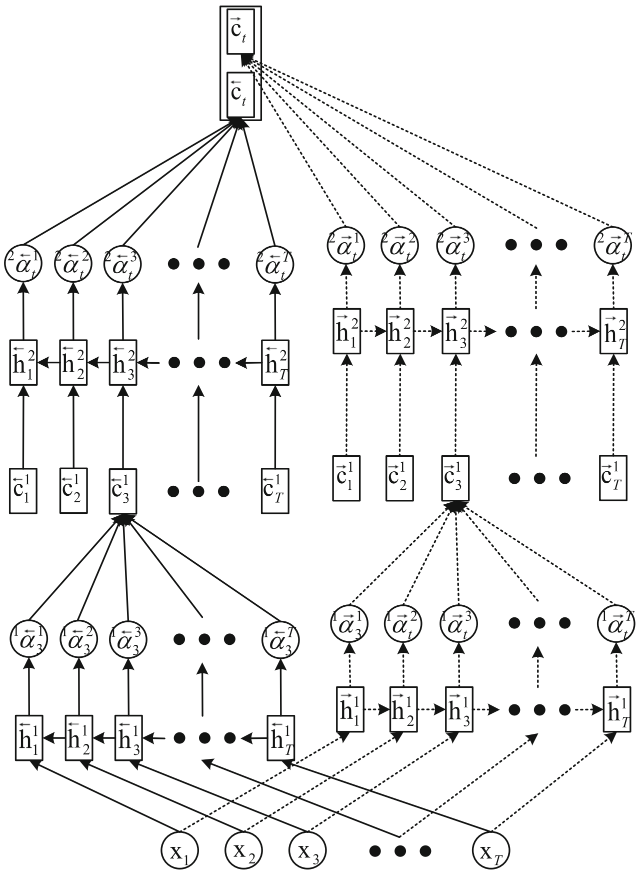
Fig. 3 The architecture of multi-layer attention

hidden state $\left(\overrightarrow{\mathbf{h}}_1^2, \cdots, \overrightarrow{\mathbf{h}}_T^2\right)$ of the second layer, and calculated the backward context vector $\overleftarrow{\mathbf{c}}_t$ by the backward weight $\left({}^2\overleftarrow{\alpha}_t^1, \cdots, {}^2\overleftarrow{\alpha}_t^T\right)$ and the backward hidden state $\left(\overleftarrow{\mathbf{h}}_1^2, \cdots, \overleftarrow{\mathbf{h}}_T^2\right)$ of the second layer, and then concatenated $\overrightarrow{\mathbf{c}}_t$ and $\overleftarrow{\mathbf{c}}_t$ to get the context vector $\mathbf{c}_t$.

4 Model

In this section, we will describe the text representation and classification model(ACNN:we proposed in this paper) which combined with attention mechanism and convolution neural network. The model architecture shown in Fig. 4. We will introduce our model from the following three subsections.

4.1 BiRNN with attention mechanism

As shown in the left part of Fig. 4, we use a BiRNN with the attention mechanism (Bi-Attention in Section 3.3) in the attention part of our model.

The input $\mathbf{x}_t$ at the t-th step of the BiRNN is the word vector corresponding to the t-th word in the input sequence. Then we got the forward hidden state $\overrightarrow{\mathbf{h}}_t$ and the backward hidden state $\overleftarrow{\mathbf{h}}_t$ for the BiRNN (as (2) and (3)). We calculate the forward context vector $\overrightarrow{\mathbf{c}}_t$ with $\overrightarrow{\mathbf{h}}_t$ and $\overrightarrow{\alpha}_t$, and the backward context vector $\overleftarrow{\mathbf{c}}_t$ were calculated with $\overleftarrow{\mathbf{h}}_t$ and $\overleftarrow{\alpha}_t$ (as (5)). Where the weights $\overrightarrow{\alpha}_t$ and $\overleftarrow{\alpha}_t$ are learned as (6).

Then we concatenated the the forward context vector $\overrightarrow{\mathbf{c}}_t$ and the backward context vector $\overleftarrow{\mathbf{c}}_t$ to got the context vector $\mathbf{c}_t$ (7).

$$\mathbf{c}_t = \left[\overrightarrow{\mathbf{c}}_t^\top; \overleftarrow{\mathbf{c}}_t^\top\right]^\top \tag{7}$$

Finally, we concatenate the context vector of each step ,ie.

$$\mathbf{c} = [\mathbf{c}_1, \cdots, \mathbf{c}_T], \tag{8}$$

as the input of the convolution operation in the middle part of Fig. 4.

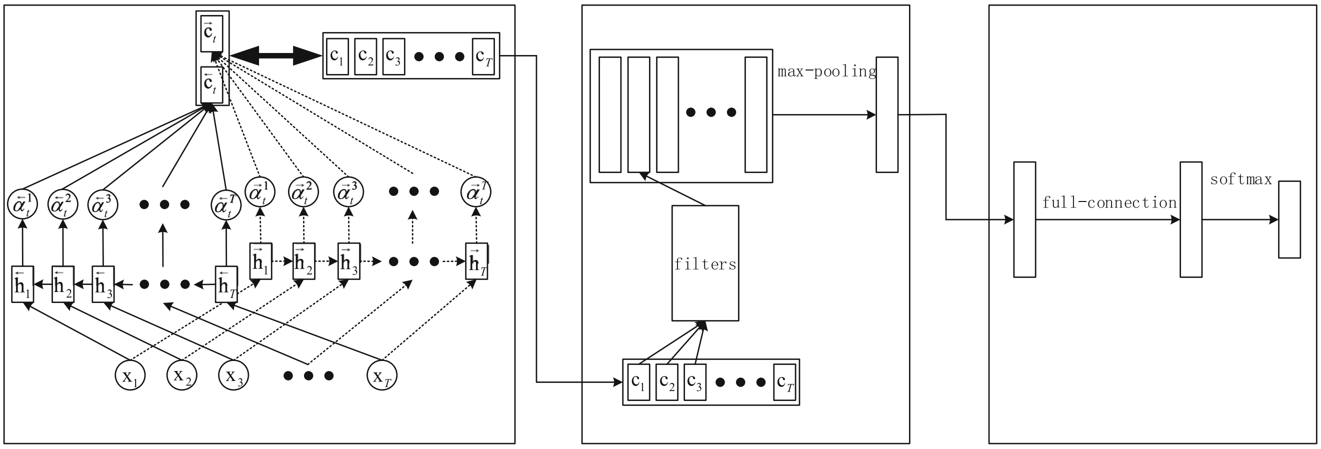

Fig. 4 The architecture of the ACNN

4.2 Convolution and max-pooling

As the middle part of Fig. 4 shown, we apply a max-pooling operation after the convolution operation to get the sentence vector **m**.

In the convolution operation, we use the context vector **c** obtained by the attention model as the input to the convolution operation. The convolution operation is shown as (9)

$$\mathbf{v}_i = f\left(\mathbf{V} \cdot \mathbf{c}_{i:i+h} + \mathbf{b}\right) \tag{9}$$

Where, $f(\cdot)$ is the non-liner activation function, h is the size of the filter window, and the weights **V** and the biases **b** are the trainable parameters. We concatenate the feature $\mathbf{v}_i$ of each step convolution operation to obtain **v**:

$$\mathbf{v} = [\mathbf{v}_1, \cdots, \mathbf{v}_T] \tag{10}$$

Then, use the max-pooling(see (11)) [5] to get the sentence vector **m**, and send **m** to the classifier shown in the right part of Fig. 4.

$$\mathbf{m} = \max(\mathbf{v}) \tag{11}$$

4.3 Softmax classifier

Shown in the right part of Fig. 4, there is a fully connected layer after the max-pooling layer, followed by is a softmax classifier to predict the text label.

In this part, the sentence vector **m**, which is the output of the max-pooling layer, are used as the input of classifier. Intuitively, after the full connection transform, there is a softmax function(see (12)) transform to predict the probability p_k of the text belongs to the category k, and then by the argmax to obtain the predict of the text.

$$p_k = \frac{\exp(y_k)}{\sum_{j=1}^{n} \exp\left(y_j\right)} \tag{12}$$

Where y_k is the output of the transition between the full connection layers and the softmax layer, and p_k is the output of the softmax layer.

5 Experiments

This section mainly introduces the datasets, experimental setup and the model variations.

5.1 Datasets

We test the model proposed in this paper on 8 benchmarks text classification datasets. A summary of the datasets shown in Table 1

Table 1 A summary of the datasets

Datasets	C	l	N	|V|	T
MR	2	20	10,662	18,765	CV
Subj	2	23	10,000	21,323	CV
SST	5	18	11,855	17,836	2,210
SST2	2	18	9,613	16.185	1,821
IMDB	2	231	50,000	392K	25,000
TREC	6	10	5,952	9,125	500
CR	2	18	3739	5340	CV
MPQA	2	3	10504	6246	CV

Note, C: the number of the target classes. L: average length of the sentences in the dataset. N: dataset size. $|V|$: Vocabulary size of the dataset. T: test set size(CV means use the 10-fold cross-validation)

- MR: MR (Movie reviews) was first used in [28] with one line per review. The dataset has 5331 positive and 5331 negative processed reviews. There are two target classes(positive or negative) of this dataset. So this is a binary classification task. We use the 10-fold cross-validation on this dataset.
- Subj: Sub (Subjectivity) with 5,000 subjective and 5,000 objective processed sentences, was first used in [27]. There are two target classes(subjective or objective) of this dataset. So this is a binary classification task, too. And we use the 10-fold cross-validation on this dataset, too.
- SST: SST (Stanford Sentiment Treebank) was split for train/dev/test provided by [31], with 5 labels (very positivity, positive, neutral, negative and very negative). There 11855 samples in this dataset, 8544 for train set, 2210 for test set and others for dev set.
- SST2: SST2 is derived from SST. We removed the neutral reviews, and merge the very positive reviews and the positive reviews as positive reviews, the negative reviews and the very negative reviews as negative reviews. So there are 9,613 samples,7,792 for train set and 1,821 for test set. And this is a binary classification task, too.
- IMDB: IMDB contains 50,000 reviews split evenly into 25k train and 25k test sets, provided by [22]. This is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets. There is additional unlabeled data for use as well. This also is a binary classification task.
- TREC: TREC is a question classification dataset with 6 labels, first used in [20]. There 5952 samples in this dataset. There are 1000; 2000; 3000; 4000 and 5500 samples labeled train set respectively, and the test set have 500 samples.

Table 2 Results of our models and the state-of-the-art models

Model	MR	Subj	SST	SST2	TREC	CR	MPQA
ACNN(BiRNN)	81.6	93.3	46.6	86.0	94.0	86.1	90.3
ACNN(BiLSTM)	**83.4**	**94.2**	**49.2**	**87.8**	**95.8**	**87.5**	90.5
ACNN(BiGRU)	82.6	94.1	47.9	87.4	95.3	86.9	**90.8**
FastSent	70.8	88.7	–	–	76.8	78.4	80.6
FastSent+AE	71.8	88.8	–	–	80.4	76.7	81.5
SkipThought	76.5	93.6	–	82.0	**92.2**	80.1	87.1
SkipThought-LN	**79.4**	**93.7**	**82.9**	–	88.4	**83.1**	**89.3**
Dependency Tree-LSTM	–	–	48.4	85.7	–		
Tree-LSTM(randomly initialized vectors)	–	–	43.9	82.0	–	–	–
Tree-LSTM(Glove vectors, fixed)	–	–	49.7	87.5	–		
Tree-LSTM(Glove vectors, tuned)	–	–	**51.0**	**88.0**	–	–	–
ParagraphVec (DBOW)	60.2	76.3	–	–	59.4	66.9	70.7
SDAE+embs.	**74.6**	**90.8**	–	–	78.4	78.0	**86.9**
Unigram-TFIDF	73.7	90.3	–	–	**85.0**	**79.2**	82.4
CNN-static	81.0	93.0	45.5	86.8	92.8	84.7	**89.6**
CNN-non-static	**81.5**	93.4	**48.0**	87.2	**93.6**	84.3	89.5
CNN-multichannel	81.1	93.2	47.4	**88.1**	92.2	**85.0**	89.4
RNTN	–	–	45.7	85.4	–	–	–
DCNN	–	–	48.5	86.8	93.0	–	–
Paragraph-Vec	74.8	90.5	48.7	87.8	91.8	78.1	74.2
MNB	79.0	**93.6**	–	–	–	80.0	86.3
word2vec BOW	77.7	90.9	–	79.7	83.6	79.8	88.3
fastText BOW	76.5	**91.6**	–	78.8	81.8	78.9	87.4
GloVe BOW	**78.7**	**91.6**	–	79.8	83.6	78.5	87.6
GloVe Positional Encoding	78.3	91.1	–	80.6	83.3	77.4	87.1
BiLSTM-Max (untrained)	77.5	89.6	–	**80.7**	**85.8**	**81.3**	**88.7**
CaptionRep (bow)	61.9	77.4	–	–	72.2	69.3	70.8
DictRep BOW+embs.	**76.7**	**90.7**	–	–	81.0	**78.7**	**87.2**
NMT En-to-Fr	64.7	84.9	–	–	**82.8**	70.1	81.5
BiLSTM-Max (on SNLI)	79.9	92.1	–	83.3	**88.7**	84.6	89.8
BiLSTM-Max (on AllNLI)	**81.1**	**92.4**	–	**84.6**	88.2	**86.3**	**90.2**
Naive Bayes - SVM	79.4	93.2	–	**83.1**	–	81.8	86.3
RNN	77.2	82.3	–	–	90.2	82.3	90.1
BRNN	82.3	94.2	–	–	91.0	82.6	90.3
AdaSent	**83.1**	**95.5**	–	–	**92.4**	**86.3**	**93.3**
G-Dropout	79.0	93.4	–	–	–	**82.1**	86.1
F-Dropout	**79.1**	**93.6**	–	–	–	81.9	**86.3**

Note: the results of the state-of-the-art methods are reported by the corresponding papers. The baseline of our model is BiRNN based model

– CR: Annotated customer reviews of 14 products obtained from Amazon [10]. The task is to classify each customer review into positive and negative categories.
– MPQA: Phrase level opinion polarity detection subtask of the MPQA data set [37].

5.2 Experimental setup

– word vectors: The word vectors we used were pre trained by Mikolov et al. [23] on 100 billion words of GoogleNews. The word vectors was trained with CBOW (continuous bag-of-words) model, and the dimensionality of the word vectors is 300. For the words not present in the word vectors, initialized randomly. To make sure that the dimensionality of the words are same as the hidden units number of the BiRNN, this is a full connection layer we use in this study.

$$\mathbf{u} = f(\mathbf{W} \cdot \mathbf{w} + \mathbf{b}) \tag{13}$$

where, $f(\cdot)$ is the nun-liner activation function. $\mathbf{w}$ is the word vectors, $\mathbf{W}$ and $\mathbf{b}$ are the trainable parameters. And the dimensionality of $\mathbf{u}$ is same as the hidden units number.

– Weights and biases initialization: All weights in the training are randomly initialized with a normal distribution which mean is 0 and standard variance of 0.1. All bias are initialized with 0.1.
– Hyperparameters: In the experiment, we used the Adam [15] optimization method to train our model with a learning rate is set to 0.001. For the fully connected layer, the L2 regularization method is used to prevent over-fitting, with the rate is set to 0.0001. The rate of dropout [32] is set to 0.5. Batch size for train is 64, and hidden units of the BiRNN is set to 128. For convolutions layer, filter windows of 3, 4 and 5 with 100 for each.

5.3 Model variations

There are three variation models in our experiments. The difference between the three models are that the recurrent neural networks structure of the attention model part. In other words, is that the structure used to get the hidden state **h** are different.

– ACNN(BiRNN): The recurrent neural networks structure based on original RNN. The hidden state **h** calculate as (14)

$$\mathbf{h}_t = \mathrm{BiRNN}(\mathbf{x}_t), \ t \in [1, \cdots, T] \tag{14}$$

For simplicity, BiRNN is used to represent the calculation of bidirectional recurrent neural networks.
– ACNN(BiLSTM): The recurrent neural networks structure based on LSTM. The hidden state **h** calculate as following:

$$\mathbf{h}_t = \mathrm{BiLSTM}(\mathbf{x}_t), \ t \in [1, \cdots, T] \tag{15}$$

where, LSTM cell used to replace of the original RNN cell.
– ACNN(BiGRU): The recurrent neural networks structure based on GRU. The hidden state **h** calculate as following:

$$\mathbf{h}_t = \mathrm{BiGRU}(\mathbf{x}_t), \ t \in [1, \cdots, T] \tag{16}$$

where, GRU cell used to replace of the original RNN cell.

6 Results and discussion

In this section, we will give the analysis of the experimental results. The classification accuracy of our ACNN compared with the state-of-the-art models is shown in Tables 2 and 3.

Table 3 Performance of models on the IMDB sentiment classification task

Models	Accuracy
ACNN(BiRNN)	88.4
ACNN(BiLSTM)	<u>**91.1**</u>
ACNN(BiGRU)	<u>**91.1**</u>
MNB	86.6
G-Dropout	91.2
F-Dropout	91.1
NBSVM	91.2
Paragraph-Vec	**92.5**
LSTM with tuning and dropout	86.5
LSTM initialized with word embeddings	86.5
LM-LSTM	90.0
SA-LSTM	**92.7**
SA-LSTM with linear gain	90.8
SA-LSTM with joint training	85.3
Full+Unlabeled+BoW	88.9
WRRBM + BoW (bnc)	89.2
NBSVM-bi	91.2
seq2-bown-CNN	**92.3**

Underlined are the best results for ACNNs, and in bold are best results of the state-of-the-art methods reported by the corresponding papers.

– Compare with the state-of-the-art models: As shown in Table 2 the BiRNN based model got the same performance as most of the state-of-the-art models on the 7 text classification tasks. The BiLSTM based model and the BiGRU based model get almost the same accuracy as the best result of the state-of-the-art models. In particular, on the TREC dataset and CR dataset, our methods achieves the best accuracy. We reduce the error rate by 6.3% with BiRNN based ACNN, by 26.7% with BiGRU based ACNN and by 34.4% with BiLSTM based ACNN for the TREC dataset, and by 8.8% with BiLSTM based ACNN for the CR dataset. For the MR dataset, the accuracy of our model BiLSTM based ACNN are the same as the best of the state-of-the-art methods AdaSent. For the Subj dataset, the SST dataset and the MPQA dataset, ACNN got the second higher accuracy, only lower than AdaSent. And for the SST2 dataset, ACNN got the third higher accuracy, only lower than Tree-LSTM (Glove vectors, tuned) and CNN-multichannel. The results prove the effectiveness of the proposed method.
– Compare BiRNN, BiLSTM and BiGRU based model: The BiLSTM based ACNN and the BiGRU based ACNN got a better performance than the BiRNN based model. For all of the datasets, the max length of

sentences are all than 30 words. As discuss in the section Background that the RNN is poorly performing for long-dependent problems, and the LSTM use the gated mechanism to improve RNN. GRU only changes the gated mechanism of the LSTM. So that BiGRU based ACNN and BiLSTM based ACNN can get a better performance than BiRNN based ACNN. GRU is the improvement of the LSTM gating mechanism, reducing the number of gates, so the training time is accelerated. However, there are less weights, performance will be not better than LSTM. So that BiGRU based ACNN got less performance than BiLSTM based ACNN.

- Performance on the IMDB dataset: Shown in Table 3, the accuracy of our model is 91.1%. It's better than most of the state-of-the-art models. But the Paragraph-Vec, SA-LSTM and seq2-bown-CNN are better than our model. There are multiple sentences in a sample in the IMDB dataset. In the experiment, we treat a sample as a sentence. So the average length of the sentences in the IMDB dataset is 231(shown in Table 1). The RNN is not good at long-dependent problem, even though the LSTM is the improvement of RNN, but for a too long sentence, it's not good, too. The Paragraph-Vec [19] learned a paragraph vector, which may be more suitable for multi-sentence data processing. Maybe we can use a hierarchical structure(like [39] done) to handle multiple sentence data.
- Convergence speed: Seen in Fig. 5, with the training is going on, BiLSTM based ACNN and BiGRU based ACNN will be able to get more than 90% accuracy soon, almost 1000 batches of training, but BiRNN based ACNN requires 5,000 batches of training to reach 90% accuracy. So the convergence speed of the BiLSTM based ACNN and the BiGRU based ACNNs are quicker than BiRNN based ACNN. As discuss in the Background, LSTM and GRU are better at long dependence problems than RNN. And for all of the datasets, the max length of sentence are all than 30 words. So for the long sentence, the BiLSTM based ACNN and the BiGRU based ACNN are easier to get a local optimal than BiRNN based ACNN while the LSTM and the GRU improved the performance of the RNN for long sentence.
- Compare with BRCNNs (with no attention mechanism): As shown in Table 4, when we stack the LSTM layers, the accuracy is improved. However, the ACNN with 1 attention layer got the same performance as the BRCNN with 2 or 3 LSTM layers. The accuracy of the validation set is very volatile when we stack LSTM layers (2 or 3 LSTM layers, shown in Fig. 6), but not appear when we stack the attention layers(shown in Fig. 7). So when we join the attention mechanism to our model, it can be a stable convergence to a local optimal.

Table 4 Performance of models on the TREC dataset

Models	Accuracy
ACNN(1 BiLSTM with attention layer)	95.8
ACNN(2 BiLSTM with attention layers)	95.8
ACNN(3 BiLSTM with attention layers)	95.3
BRCNN(1 BiLSTM layer)	95.3
BRCNN(2 BiLSTM layers)	95.8
BRCNN(3 BiLSTM layers)	95.8

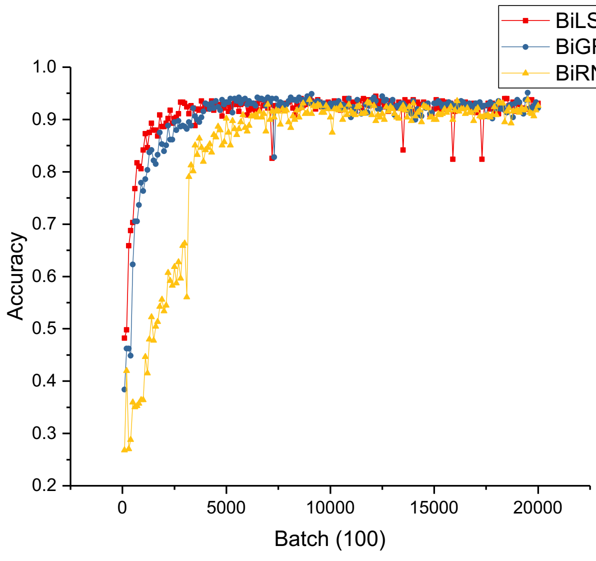

Fig. 5 The accuracy of different ACNNs

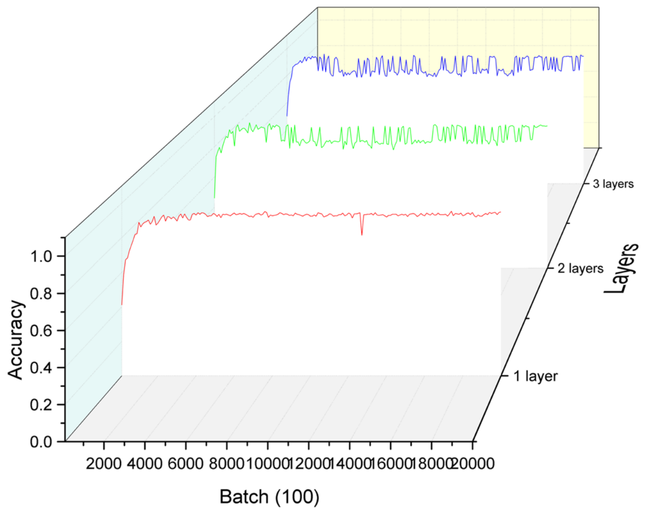

Fig. 6 The accuracy of BRCNN based on multi-layer BiLSTM with attention mechanism

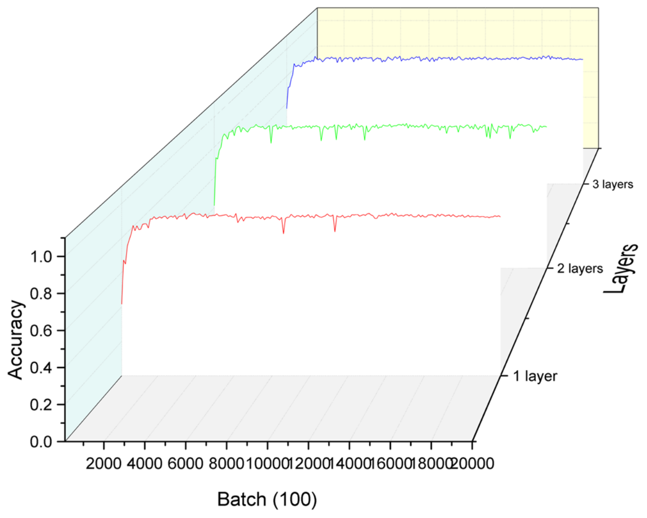

Fig. 7 The accuracy of ACNN based on BiLSTM with multi layers

That is, when we apply the attention mechanism, we get a better sentence vector.

– Multi-layer attention: We also trained the ACNNs which have multi-layer attention (seen in Fig. 3), the accuracy seen in Table 4. When we stack the attention layers the accuracy does not increase. This means that a layer of attention is enough.

In this section, we have a detailed analysis of the experimental results. The experiment showed that ACNNs can learning a better sentence vector for classification. The RNN of the attention mechanism models the word order information of a sentence. The attention mechanism give different weight of the information and calculate the context vectors. The convolutional operation get the locale information and the max-pooling operation get the max feature information. So it is better for sentence representation and classification.

7 Conclusion

This paper propose a bi-attention, a multi-layer attention, and describes a text representation and classification model based on attention mechanism and convolution neural network. The bi-attention has two attention mechanisms to learn two context vectors, forward RNN with attention to learn forward context vector and backward RNN with attention to learn backward context vector, and then concatenation them to get context vector. The multi-layer attention is the stack of the bi-attention. For ACNN, it combines bi-attention and convolutional neural network for sentence representation and classification. All the testing on the 8 benchmarks text classification datasets, our model achieved a better or the same performance compare with the state-of-the-art methods, which shows that our approach is feasible. In other word, it's shown that ACNNs can learn a better sentence vector for classification.

8 Future work

In this work we only focused on English text representation and classification, further we will do more experiments on other language text. We can also speed up the training of the model using only the max-pooling operation instead of the convolution operation and max-pooling operation.

Acknowledgements This research was supported by the National Natural Science Foundation of China [NSFC61572005], the Fundamental Research Funds for the Central Universities [2016JBM080], and Key Projects of Science and Technology Research of Hebei Province Higher Education [ZD2-017304].

References

1. Arora S, Liang Y, Ma T (2017) A simple but tough-to-beat baseline for sentence embeddings
2. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv:14090473
3. Bengio Y, Ducharme R, Vincent P, Jauvin C (2003) A neural probabilistic language model. J Mach Learn Res 3:1137–1155
4. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:14061078
5. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. J Mach Learn Res 12:2493–2537
6. Conneau A, Kiela D, Schwenk H, Barrault L, Bordes A (2017) Supervised learning of universal sentence representations from natural language inference data. arXiv:170502364
7. Dai AM, Le QV (2015) Semi-supervised sequence learning. In: Advances in neural information processing systems, pp 3079–3087
8. Hill F, Cho K, Korhonen A (2016) Learning distributed representations of sentences from unlabelled data. arXiv:160203483
9. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
10. Hu M, Liu B (2004) Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 168–177
11. Huang E, Socher R, Manning C, Ng A (2012) Improving word representations via global context and multiple word prototypes. In: ACL. ACL, pp 873–882
12. Johnson R, Zhang T (2014) Effective use of word order for text categorization with convolutional neural networks. arXiv:14121058
13. Kalchbrenner N, Grefenstette E, Blunsom P (2014) A convolutional neural network for modelling sentences. arXiv:14042188
14. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv:14085882

15. Kingma D, Ba J (2014) Adam: a method for stochastic optimization. arXiv:14126980
16. Kiros R, Zhu Y, Salakhutdinov RR, Zemel R, Urtasun R, Torralba A, Fidler S (2015) Skip-thought vectors. In: Advances in neural information processing systems, pp 3294–3302
17. Lai A, Hockenmaier J (2014) Illinois-lh: a denotational and distributional approach to semantics. In: SemEval@ COLING, pp 329–334
18. Lai S, Xu L, Liu K, Zhao J (2015) Recurrent convolutional neural networks for text classification. In: AAAI, vol 333, pp 2267–2273
19. Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: Proceedings of the 31st international conference on machine learning (ICML-14), pp 1188–1196
20. Li X, Roth D (2002) Learning question classifiers. In: COLING. ACL, pp 1–7
21. Lin Z, Feng M, Santos CNd, Yu M, Xiang B, Zhou B, Bengio Y (2017) A structured self-attentive sentence embedding. arXiv:170303130
22. Maas A, Daly R, Pham P, Huang D, Ng A, Potts C (2011) Learning word vectors for sentiment analysis. In: ACL. ACL, pp 142–150
23. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119
24. Mikolov T, Chen K, Corrado G, Dean J (2013a) Efficient estimation of word representations in vector space. arXiv:13013781
25. Mnih A, Hinton G (2007) Three new graphical models for statistical language modelling. In: Proceedings of the 24th international conference on machine learning. ACM, pp 641–648
26. Mnih V, Heess N, Graves A et al (2014) Recurrent models of visual attention. In: Advances in neural information processing systems, pp 2204–2212
27. Pang B, Lee L (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: ACL. ACL, p 271
28. Pang B, Lee L (2005) Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In: ACL. ACL, pp 115–124
29. Pang B, Lee L et al (2008) Opinion mining and sentiment analysis. Foundations and Trends®, in Information Retrieval 2(1–2):1–135
30. Sahami M, Dumais S, Heckerman D, Horvitz E (1998) A Bayesian approach to filtering junk e-mail. In: Learning for text categorization: papers from the 1998 workshop, vol 62, pp 98–105
31. Socher R, Perelygin A, Wu JY, Chuang J, Manning CD, Ng AY, Potts C et al (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the conference on empirical methods in natural language processing (EMNLP), vol 1631, pp 1642
32. Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
33. Tai KS, Socher R, Manning CD (2015) Improved semantic representations from tree-structured long short-term memory networks. arXiv:150300075
34. Tang D, Qin B, Liu T (2015) Document modeling with gated recurrent neural network for sentiment classification. In: EMNLP, pp 1422–1432
35. Wang S, Manning C (2013) Fast dropout training. In: ICML, pp 118–126
36. Wang S, Manning CD (2012) Baselines and bigrams: simple, good sentiment and topic classification. In: Proceedings of the 50th Annual meeting of the association for computational linguistics: short papers, vol 2. Association for Computational Linguistics, pp 90–94
37. Wiebe J, Wilson T, Cardie C (2005) Annotating expressions of opinions and emotions in language. Lang Resour Eval 39(2):165–210
38. Wieting J, Bansal M, Gimpel K, Livescu K (2015) Towards universal paraphrastic sentence embeddings. arXiv:151108198
39. Yang Z, Yang D, Dyer C, He X, Smola AJ, Hovy EH (2016) Hierarchical attention networks for document classification. In: HLT-NAACL, pp 1480–1489
40. Zhang X, Zhao J, LeCun Y (2015) Character-level convolutional networks for text classification. In: Advances in neural information processing systems, pp 649–657
41. Zhao H, Lu Z, Poupart P (2015) Self-adaptive hierarchical sentence model. In: IJCAI, pp 4069–4076