Neural Computing and Applications (2021) 33:11297–11308
https://doi.org/10.1007/s00521-020-05652-6

ORIGINAL ARTICLE

A dual deep neural network with phrase structure and attention mechanism for sentiment analysis

An ablation experiment on Chinese short financial texts

Dongning Rao[1] · Sihong Huang[1] · Zhihua Jiang[2] · Ganesh Gopal Deverajan[3] · Rizwan Patan[4]

Received: 27 December 2019 / Accepted: 17 December 2020 / Published online: 11 January 2021

Abstract
Sentiment analysis of short texts is difficult for their simplicity and compactness. This goes a step further when it comes to the Chinese texts. Although deep learning achieved better accuracy in sentiment analysis, there is a lack of explain-ability. Thus, this paper evaluates the effectiveness of techniques for sentiment analysis of Chinese short financial texts with deep learning. For this, we built a Chinese short financial texts corpus (**CSFC**) and designed an ablation experiment. Beside the **CFSC**, we used a Chinese review collection and an English short-text repository in the experiment for comparison. There are five techniques involved. They are the Pinyin, the segmentation, the lexical analysis, the phrase structure and the attention mechanism. As results, we found that the phrase structure and the attention mechanism are two of the best. Therefore, the best model in the experiment is called a Phrase Structure and Attention-based Deep network model (*PhraSAD*). Moreover, to improve the classification accuracy on neutral data, we use a dual classifier strategy for 3-class problems. Experimental results showed that *PhraSAD* outperformed all other compared models on all experimental datasets.

Keywords Sentiment analysis · Chinese short text · Deep learning · Phrase structure · Attention · Dual classifier

✉ Rizwan Patan
rizwan@vrsiddhartha.ac.in

Dongning Rao
raodn@gdut.edu.cn

Sihong Huang
2111605084@mail2.gdut.edu.cn

Zhihua Jiang
tjiangzhh@jnu.edu.cn

Ganesh Gopal Deverajan
ganesh.gopal@galgotiasuniversity.edu.in

[1] School of Computer, Guangdong University of Technology, Guangzhou 510006, People's Republic of China

[2] Department of Computer Science, Jinan University, Guangzhou 510632, People's Republic of China

[3] Department of Computer Science Engineering, Chandigarh University, Mohali, Punjab 140413, India

[4] Department of Computer Science and Engineering, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada 520007, Andhra Pradesh, India

1 Introduction

In recent years, sentiment analysis is becoming more and more important in natural language processing tasks [3], which is especially hard for Chinese short texts. The state-of-the-art methods for sentiment analysis mostly adopt deep neural networks [14, 36]. However, these methods either depend heavily on manually labeling features or regional word-level representation of texts [32]. They have advantages for sentences or documents, but still have limitations for short texts [24]. For instance, the length of short texts is usually limited to 140 characters, referred as microblog texts. This makes text simple and compact. These characteristics increase the difficulty of sentiment analysis. For example, short texts often lack sentence elements. Moreover, short texts may contain multiple emotions. Apart from that, there is not only the homo-phonic but also the slang in Chinese.

However, short-text sentiment analysis is a wealth topic for it is profitable for financial texts [9, 21]. As a matter of fact, most retail investor does not present long speech and their emotion is expressed through short text often. On the other hand, if the emotion of the stock market is detected, it can lead to an arbitration [38]. Therefore, it is valuable to dig into the Chinese financial short-text's sentiment analysis. Along with the many success of deep learning, it is also popularly used in sentiment analysis. Sadly, it is not explainable. Particularly, some approaches have been proposed for short-text sentiment analysis. For example, Zeng [39] built latent topic models and memory networks for short-text classification, in which the topic models were pre-defined and very similar to original classes. As another example, Wang [32] proposed a cross-domain sentiment classification algorithm, while their algorithm uses Sentiment Related Index (SRI) to measure the association between different lexical elements across domains. Yet another example, Wu [35] built a slang sentiment dictionary to aid sentiment analysis, and it is laborious and time-consuming to collect a comprehensive list of slang words.

Based on the above, this work dedicates to figure out which techniques suit better for the sentiment analysis of Chinese financial short texts with deep learning. We designed an ablation experiment and tested among the five most popular features used in NLP. They are the Pinyin [25], the word segmentation [16], the lexical analysis [21], the phrase structures [20] and the attention mechanism [29].

The contributions of our paper are as follows:

- We collected a Chinese financial short-text corpus for sentiment analysis. It has more than 18,000 posts collected from stock forums.[1]
- We tested the Pinyin features, the segmentation, the phrase structure, the lexical analysis and the attention mechanism in an ablation experiment. This explains which technique(s) is more valuable in sentiment analysis with deep learning, for Chinese financial short texts.
- We found the most cost-effective model is a Phrase Structure and Attention-based Deep network model (*PhraSAD*). It combines phrase-level representation and self-attention information with deep neural networks for sentiment analysis. The phrases can provide a stronger emotional indication than single words. And the self-attention mechanism can be used to extract in-sentence correlation.
- We provided a dual classifier strategy to further improve the classification accuracy for neutral data.

The remainder of this paper is arranged as follows. In Sect. 2, we introduce related work. In Sect. 3, we present the overall architecture and algorithms of the *PhraSAD*. In Sect. 4, we evaluate the model performance with comparison. In the last section, we conclude this paper with future work.

2 Related work

In this section, the problem and our basic deep learning model are stated before major techniques used in Chinese natural language processing (NLP) are presented. These techniques are Pinyin, the segmentation, the lexical analysis, the phrase structure and the attention mechanism.

2.1 Sentiment analysis

Sentiment analysis is also known as polarity analysis, which aims to analyze, process, summarize and reason the data with emotion color in the text. In recent years, it has been developed rapidly in many applications, such as stock prediction and product reviews. For example, Ruan [26] manually labeled a Twitter dataset and used different neural networks for sentiment analysis and Batra [2] performed sentiment analysis on the tweets related to Apple products, which were extracted from financial StockTwits[2] from 2010 to 2017.

Traditionally, short-text sentiment analysis problems are categorized as 2-class (positive and negative), or 3-class (positive, neutral and negative). The following examples illustrate some characteristics of short texts. All examples are in the **CSFC** and annotated by three experts.

Example 1 Sample text: "我今天买了新股!" (I bought a new stock today!) This text does not provide sufficient information, such as the name or the value of the stock. However, the sentiment polarity of this text is positive, because the speaker would not buy it otherwise.

Example 2 Sample text: "今天不大跌就已经是胜利啦!" (It is a victory if (the stock) do not crash today!) This text lacks a subject which should be the price of some stock. The emotion of this text is negative. See the sentiment word "大跌," which means crash.

Example 3 Sample text: "涨是期待!跌是无奈!涨涨跌跌是一种常态!" (Expecting jumps while suffering slumps! Volatile and mix are in our daily life!) This text contains multiple emotions: "expecting" signals a positive emotion,

[1] The corpus will be available online after publication.

[2] A social networking website: http://stocktwits.com.

while “suffering” suggests a negative emotion. These two emotions are conflicting. In practice, this text is regarded as neutral.

However, basic approaches in this filed have many limitations. For example, the 3-class problem is difficult for the bag-of-word [33]-based methods. See the following example.

Example 4 Sample text: “利空出尽, 就是利好。” (It is good news that all bad news are revealed!) There are two sentiment words: “利空,” which means bad, and “利好,” which means good. A naive bag-of-word algorithm will regard this text as neutral, but this sentence is positive indeed.

2.2 Neural Language Model

The basic deep learning models for NLP, includes CNNs (Convolutional Neural Networks) and RNNs (Recurrent Neural Networks). CNNs are initially designed for computer vision while Yanmei [19] performed sentiment analysis on 1000 microblog comments using CNNs, and Ouyang [25] used CNNs to analyze sentiment polarity in movie reviews. On the other hand, RNNs are particularly beneficial for sequential data. Examples in this line include Silhavy [27], which proposed Hierarchical Bi-directional RNNs (HBRNNs) to perform emotion analysis on customer reviews, and Chen [4], which used the RNN-GRU model to analyze sentiment in the datasets of Yelp and IMDB, whose results indicated that RNN-GRU outperformed many baseline methods. Additionally, BiLSTM (Bi-directional Long Short-Term Memory) [11] is currently widely used in sequence-to-sequence learning, which took context into the consideration.

2.3 Pinyin

Pinyin is the romanization of Chinese characters and also a very popular input method for Chinese text. This leads to a near-homophones preferable social media culture, and some researchers consequently believe that leveraging on Pinyin feature could boost the performance of a Chinese NLP [25] task. We refer readers to previous studies for augmentations, see [25]. The following is an example.

Example 5 Sample text: “喋喋不休!” (Chatter.) The Pinyin for this sentence is: “Die Die Bu Xiu.” It is a homophone for “跌跌不休!”. (A sustained drop.) It is a typical sample of a negative text.

In previous studies, the Pinyin along with the Chinese characters were concatenated and used as inputs to NLP algorithms.

2.4 Segmentation

Chinese word segmentation is the task of splitting a Chinese text into valid words. It has a long history and many on-the-shelf tools.[3] However, recently, some researchers found that there is no more beneficial to learn word embedding after segmentation [16]. They supposed that “it is because word-based models are more vulnerable to data sparsity and the presence of out-of-vocabulary (OOV) words, and thus more prone to overfitting.” Here, we use a previous sample as an illustration.

Example 6 Sample text: “今天不大跌就已经是胜利啦!” (It is a victory if (the stock) do not crash today!) It can be segmented as “今天 不 大跌 就 已经 是 胜利 啦 !”.

Techniques took advantages of this pre-process used segmentation instead of the Chinese characters as inputs. The word segmentation means a technique of NLP, but in this paper, we used it to represent the result of this technique. This result can be another feature of NLP tools.

2.5 Syntax Analysis

Syntax analysis often results in a syntax tree structure, which can be provided as prior knowledge in the NLP. Tree-LSTM [28], DAG-LSTM [42] and DC-TreeNN [18] are three popular models using the syntax tree, while syntax analyzer tools like CoreNLP [22] are on-the-shelf. The tree-LSTM [28] used syntax trees to organize the hierarchical structure of LSTM units. Consequently, the content vectors obtained a semantic representation like that of original input text. The DAG-LSTM [42] also extended the chain-structured LSTM to directed acyclic graphs. Thus, it incorporated additional prior knowledge into RNN. The DC-TreeNN [18] introduced the dynamic compositional neural networks over tree structures to capture the meta-knowledge across the different compositional rules. There is many available syntax analysis software. For example, the CoreNLP [22] is a famous tool developed by the Stanford University. Lexical analysis is part of syntax analysis for today’s parsers. The results of lexical analysis can be used together with either the original text or segmentation of text and inputted to any software.

2.6 Phrase Structure

Phrases are one of the grammatical structures, which can obtain more dependency information from the context. This grammatical representation has got many achievements, e.g., Vilares [30] used parsers to perform opinion mining in Spanish reviews which accorded grammatical

[3] https://github.com/fxsjy/jieba.

structure with manually crafted regulations. Lazaridou [12] is another example, which proposed a model for unsupervised induction of sentiment aspect, and discourse information. They showed that performance can be improved by incorporating latent grammatical information. Dong [7] is also another example, which proposed a system which can learn in-sentence sentiment structures using grammatical structures. We use a figure to show how to use the above techniques on the same sentence. As the results of lexical analysis, and an approach of sentence modeling [40], the phrase structure could contribute to NLP tasks as a part of the input too.

2.7 Self-attention

The attention mechanism is used to evaluate the correlation or the alignment between sequence elements, it finds the words of importance for a given query word in a sentence in NLP. It was first proposed in the field of image classification [23] and then introduced into sequence-to-sequence learning [1] rapidly while encoder–decoder networks become more and more popular. In encoder–decoder networks, an attention score is calculated by mapping functions (e.g., inner product function or cosine function) on hidden state information of encoders and decoders. Therefore, it is also called self-attention. For example, the inner product function or *cosine* function. To capture the dependency of in-sentence words, the output of encoder is set to be the input of the decoder. The Transformer [29] further introduced self-attention into sequence-to-sequence neural networks learning. Transformer proposed a novel and simple network architecture and was based solely on attention mechanism, dispensing with recurrence and convolutions entirely. Later, the BERT (Bidirectional Encoder Representations from Transformers) [6] obtained new state-of-the-art results on eleven natural language processing tasks. For sentiment analysis, Lin [17] proposed self-attentive sentence embedding and applied them in English sentiment analysis. Moreover, Du [8] uses word attention on the existing review datasets and Li [15] further combined attention with sentiment-related features on another existing review datasets.

Different with the Pinyin, lexical and phrase structures, the attention is a technique instead of input features. However, in this paper we try to discuss them together. The reason behind this is that attention is basically an automated position-related features extraction method. When we discuss attention, in this paper, we are indeed discussing position-related features.

3 Model

3.1 Overall architecture

The model is composed of two parts and a hidden self-attention layer. The first part is feature extraction. It extracts features which focus on one aspect of a sentence.

- The Pinyin emphasizes on phonetics.
- The segmentation targets on the splitting words into logical units.
- The syntax analysis aims to provide lexical roles for every word.
- The phrase-level features can provide phrase structures of word units.

In a middle layer, self-attention can be used to extract the correlation of in-sentence words. The second part offers single- or dual-classifier options. We identify neutral data for 3-class problems by designing a dual classifier strategy. That is, a pos-classifier to decide whether an example is positive and a neg-classifier to decide whether this example is negative.[4] A text is regarded as neutral only if it is both positive and negative (i.e., multiple emotions), or neither positive nor negative (i.e., no emotion). See Fig. 1.

In feature extraction part, inspired by the multiple-branch sentiment analysis model [36], we built four CNN+BiLSTM branches to deal with different feature representations. For the Pinyin branch, we used the Pypinyin[5]to generate the Pinyin embedding. In the segmentation branch, we used the Jieba[6] to perform word segmentation, and then adopt the Word2vec[7] to produce word embedding. The lexical analysis and syntactic analysis were conducted with the Stanford CoreNLP [22]. The POS (Parts-of-Speech) of words and phrase structures were obtained from the resulting syntax trees. At last, we calculated phrase embedding based on parse trees. Its algorithm will be given in the next subsection.

In classifier construction part, there are two options for multi-class problems. The first option is to construct an ensemble. In this approach, we train base classifiers for only one class. Then, the results of base classifiers are aggregated by a voting system. While in our case, the ensemble does not really need three base classifiers in this situation. It can be composed of two dual classifiers: a pos-

[4] For example, every sentence in the CSFC corpus is annotated by three experts with a positive, negative or neutral label. Hence, we can use the CSFC to train the classifiers.

[5] https://github.com/mozillazg/python-pinyin.

[6] https://pypi.org/project/jieba/.

[7] http://code.google.com/archive/p/word2vec/.

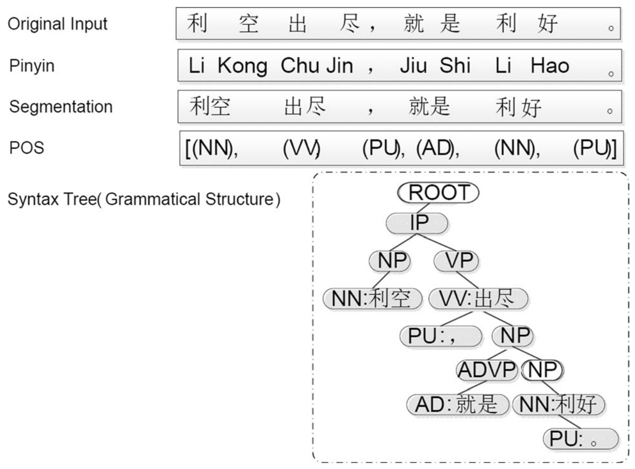
Original Input
利 空 出 尽， 就 是 利 好 。
Pinyin
Li Kong Chu Jin ， Jiu Shi Li Hao 。
Segmentation
利空 出尽 ， 就是 利好 。
POS
[(NN), (VV) (PU), (AD), (NN), (PU)]
Syntax Tree(Grammatical Structure)
ROOT
IP
NP
VP
NN:利空
VV:出尽
PU:，
NP
ADVP
NP
AD:就是
NN:利好
PU:。

Fig. 1 An example of features for a sample text

classifier and a neg-classifier. An example text is determined to be:

1. positive if the pos-classifier says “yes” and the neg-classifier says “no”;
2. negative if the pos-classifier says “no” and the neg-classifier says “yes”;
3. neutral, otherwise.

By contrast, the second option is to use a single *k*-class classifier.

All classifiers are trained with a CNN-BiLSTM network, as shown in Fig. 2. Each deep network includes CNN, Dropout, Batch Normalization and BiLSTM layers in order. Using CNN and BiLSTM jointly is because CNN can extract local information and BiLSTM can capture long-distance dependency. Apart from that, lexical features and phrase features can be combined into phrase-level representation. With concatenation, both word-level and phrase-level representations are packed together into a full-connected neural network, outputting the final label after a *softmax*.

The detailed formulas are as follows:

$$\overrightarrow{h_t^w} = LSTM(CNN(Y), \overrightarrow{h_{t-1}^w}) \tag{1}$$

$$\overleftarrow{h_t^w} = LSTM(CNN(Y), \overleftarrow{h_{t-1}^w}) \tag{2}$$

$$y_t^w = \overrightarrow{h_t^w} + \overleftarrow{h_t^w}, \tag{3}$$

where Y is the output of self-attention process, and h_t^w is the value of hidden state h, at step t. The $\overrightarrow{h_t^w}$ part of h_t^w considers information on the right side of h and the $\overleftarrow{h_t^w}$ part of h_t^w reflects information on the left side of h.

After that, we integrate the Pinyin, lexical and phrase features with the dot product, producing phrase-level representation. F^w represents word-level embedding and F^P

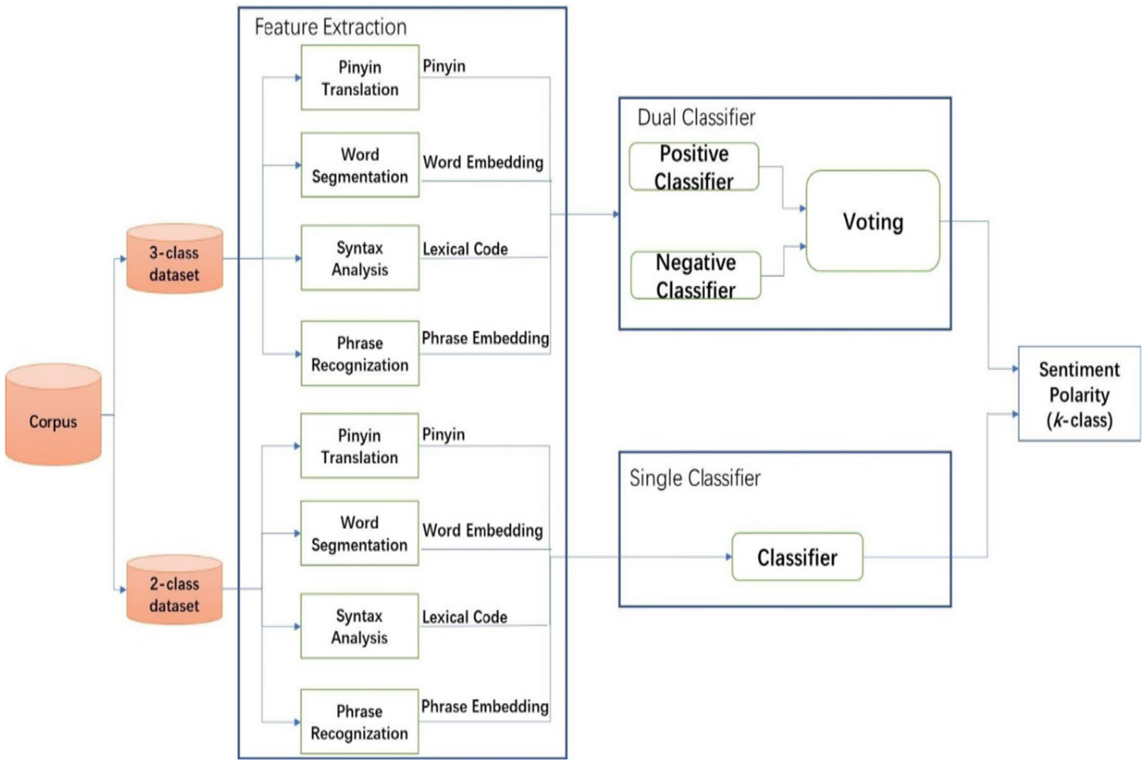

Fig. 2 Overall architecture of the ablation experiment

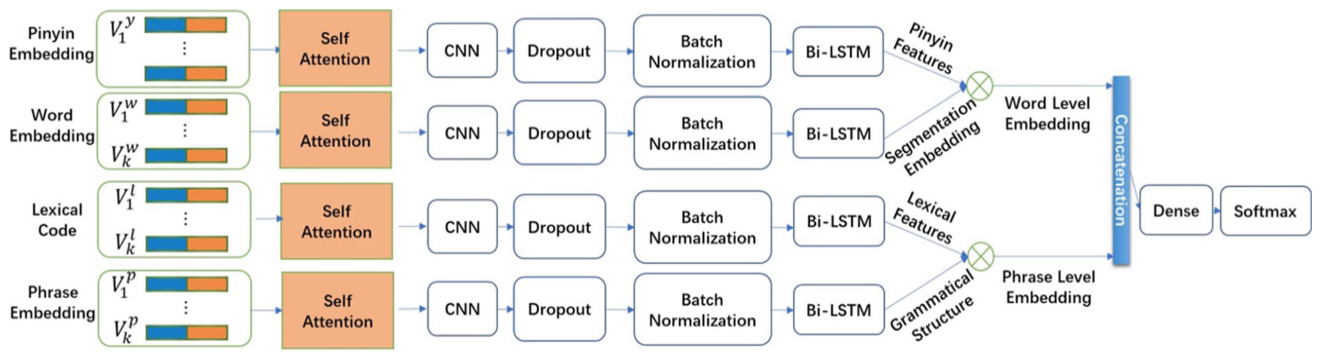

Fig. 3 The DNN (Dual or Single) classifier in the experiment

represents phrase-level embedding. The $y_t^y, y_t^w, y_t^l, y_t^p$ are inputted features for the Pinyin, word segmentation, lexical code, and phrase structure, correspondingly.

$$F^w = Dot\left(y_t^y, y_t^w\right) \tag{4}$$

$$F^P = Dot\left(y_t^l, y_t^p\right). \tag{5}$$

As a result, we get a word-level embedding and a phrase-level embedding. They are concatenated and fed into a fully connected layer.

$$F = Concat\left(F^w, F^P\right) \tag{6}$$

$$\tilde{F} = Dense(F). \tag{7}$$

At the end, through the *SoftMax*, the model outputs the sentiment polarity for a given text.

3.2 Phrase-level Representation

In this subsection, a new algorithm for calculating phrase embedding will be given. As a beginning, we use the Stanford CoreNLP[8] to perform syntactic analysis of texts in the corpus. Each syntax tree produced by CoreNLP is a binary tree, where each node has at most two children. It is easy to recognize phrase structures in such a parse tree. Figure 3 presents all results of feature extraction from a given example text. Particularly, there are two phrases separated by a comma in this text. The first phrase is a short sentence (i.e., IP) with complete subject and predicate, and the second phrase is a noun phrase (i.e., NP).

In our model, phrase embedding is calculated with a weighted linear function of word embedding. This is inspired by an existing approach which calculates the embedding of an English word with decomposition [21]. An English word can be decomposed into three parts: prefix, stem, and suffix. Therefore, the word embedding can be calculated based on these three components recursively. The phrase embedding is calculated based on the tree structure in a similar way. For the simplicity, we use abstract syntax trees by removing all the single-child nodes.

Algorithm 1 Calculating phrase embedding of root node *r*, given an abstract syntax tree

```
Input: Root node r
Output: Phrase embedding of r, namely r.e
1:  function CalPhraseEmbed(r)
2:      let r.label be the lexical code of r
3:      if r.label is a punctuation then
4:          return
5:      else
6:          if r.label is a word then
7:              return the word embedding associated with r
8:          else
9:              r.le = CalPhraseEmbed(r.lchild)
10:             r.re = CalPhraseEmbed(r.rchild)
11:             $r.e = r.le * w_1 + r.re * w_2 + b$
12:             return r.e
13:         end if
14:     end if
15: end function
```

8 https://stanfordnlp.github.io/CoreNLP/.

Algorithm 1 shows the pseudocode that calculates phrase embedding for nodes in an abstract syntax tree. It traverses the syntax tree in a depth-first way, starting from the root node and running recursively. Then, algorithm 1 calculates the embedding of every node by three cases. 1) In the punctuation-node case, there is nothing to do (line 3–4). 2) In the word-node case, the node is a leaf node. Therefore, the associated word embedding is returned directly (line 6–7). 3) In the phrase-node case, we first calculate phrase embedding for its two children recursively (line 9–10). Then, we get the weighted sum on phrase embedding of children to generate phrase embedding of the parent (line 11). Here, w1 and w2 are learned parameters to represent weights, and b is the bias. The phrase-level representation of a previous showed example is as follows.

Example 7 Sample text: “利空出尽, 就是利好。” (It is good news that all bad news is revealed!) The phrase-level representation: [[NN, Embedding of “利空”], [VV, Embedding of “出尽”], [PU, Embedding of “,”], [AD, Embedding of “就是”], [NN, Embedding of “利 好”], [PU, Embedding of “。”]].

4 Experiment

This section presents the details of our ablation experiment, including settings, comparison models and discussions. In deep learning, ablation experiments should consist of several trials, every trial removes some feature of the model.

4.1 Settings

The machine used in experiments has an Intel CoreTM i5-4590 CPU @ 3.30 GHz *4 processors and a single NVIDIA GeForce GTX 1070 Ti GPU with 8 GB VRAM and 16 GB RAM. We ran Keras 2.1.4 on Ubuntu 16.04 LTS and used TensorFlow 1.4.1 backend on the GPU. The source code of our model has been released on the GitHub.[9]

To test the effectiveness of different models, we use three corpus. Two Chinese short-text datasets (**CFSC** and **CTRD**) and one English short-text dataset (**TWITTER**) were used in our experiments. Here, the English dataset is used to show the extendibility of our method.

- CFSC. We collected 18,000 Chinese posts on Oriental Wealth Forum[10] from 2015 to 2017. Each record in this dataset was annotated by three experts. The final denotation was determined according to the majority voting principle. For that matter, all example texts in this paper come from this corpus.
- **CTRD**. This Chinese Review dataset[11] comes from a Chinese Takeaway platform. It contains only two classes: positive and negative. There are totally 4000 positive comments and 7987 negative comments.
- **TWITTER**. This is a very big English dataset which comes from the Twitter.[12] We randomly selected 10,000 examples out of fairness to be close to the size of other datasets. Moreover, for performance verification, we increased the size of the dataset to 30,000 samples. In experiments, these two datasets were, viz. **TWITTER(10 k)** and **TWITTER(30 k)**. However, because the difference between **TWITTER(10 k)** and **TWITTER(30 k)** is negligible, they are called TWITTER in all other parts of this paper but experiment result tables.

Table 1 gives the basic information about these datasets. Here, class means sentiment polarity. For each dataset, we took about 80% data as the training set and 20% data as the test set. The Avg.Len is the average length of sentences in a dataset. We choose the **CTRD** because it contains short Chinese text. Sentences in the **CTRD** is reviews, which might include sentiment but not necessary. However, as public Chinese short-text sentiment corpus is unavailable, the **CTRD** is our second-best choice.

4.2 Comparison models

We compared our model with seven related approaches. Among them, there were six deep learning-based models. They were CNN, LSTM/BiLSTM, CNN-BiLSTM, Attention, Regional CNN-LSTM (RCNN-LSTM) and Multiple Branches CNN-LSTM (MBCNN-LSTM).

1. THUCTC. This baseline was proposed by Li [14]. The paper focused on a full performance comparison between words and character-bigrams used as features in Chinese text processing tasks. We obtained its source code from the author.[13]
2. CNN. It was proposed by Kim [10]. The paper reported on a series of experiments with CNN trained on top of pre-trained word vectors for sentence-level classification tasks. Its source code is available on the GitHub.[14]
3. LSTM/BiLSTM. This method was developed by Cho [5]. The model learned phrase representations using

[9] https://github.com/fip-lab/Sentiment Analysis.

[10] http://guba.eastmoney.com/.

[11] https://github.com/SophonPlus/ChineseNlpCorpus.

[12] http://thinknook.com/wp-content/uploads/2012/09/Sentiment-Analysis-Dataset.zip.

[13] http://thuctc.thunlp.org/.

[14] https://github.com/yoonkim/CNN sentence.

Table 1 Summary of datasets

Datasets	Class	Train / Test	Avg. Len
CFSC	3	14,525 / 3632	45
CTRD	2	9589 / 2398	25
TWITTER	2	8000 / 2000	16

RNN encoder–decoder for statistical machine translation tasks and carried out the performance comparison between LSTM and BiLSTM.

4. CNN-BiLSTM. This baseline used the combination of CNN and BiLSTM to do the sentiment analysis. It can be viewed as the slimmest version of our model without phrase representation and self-attention.
5. Attention. Lin proposed it in 2017 [17]. It is a new model for extracting an interpretable sentence embedding by introducing self-attention.
6. Regional CNN-LSTM. This method combined the regional CNNs and LSTMs for sentiment analysis. The RCNN was introduced to extract features in the region.
7. Multiple Branches CNN-LSTM. This approach was introduced in 2017 [36]. The model built deep CNN-LSTMs with combined kernels from multiple branches for IMDb review sentiment analysis.

We chose them because of similar neural network architecture, i.e., deep networking. On the other hand, these methods represent different existing NLP techniques and their combinations. The THUCTC was a traditional classification method based on machine learning. Even though some related work was also involved in short-text sentiment analysis tasks, such as [32], they did not provide source codes for copyright limitations. Other deep learning approaches over syntax trees, such as the DC-TreeNN [18], are not open sourced as of the date of writing.

4.3 Results and discussion

Results are presented in Tables 2 and 3. The parameter setting of our model is as follows. We used 64 filters in each CNN. The filter size was [1, 2, 2]. The dropout rate was 0.5 and the learning rate was 0.01. The dimension of both word and phrase embedding was 100. And the dimension of hidden states in the LSTM was 64. We observed that the ensemble classifier outperformed the single classifier in our experiments. Therefore, we adopt the ensemble classifier for 3-class problems by default, as shown in Fig. 1.

Because the **TWITTER** is an English corpus, the Pinyin is not used on it. Using the **CTRD** leads to higher accuracy, for both previous studies and our method. It seems like the **CTRD** is an "easier" corpus. We believe this is because people are more emotional when it comes to stocks rather than goods, which caused the **CFSC** more complicated.

Overall Performance: As shown in Table 2, the *PhraSAD* outperformed all other models in all datasets. We adopted the classification accuracy as evaluation metric. Our model gained an absolute 16.93% improvement on **CFSC**, 1.27% on **CTRD** and 2.48% on **TWITTER**, compared with the most similar model, CNN-BiLSTM. Particularly, it improved the classification accuracy on **CFSC** by a substantial margin. Since **CFSC** is annotated manually, it is more credible than the other two datasets which are labeled by machine learning. It showed that, combining phrase-level representation and self-attention together seems to be a good choice for short-text analysis, for either Chinese or English short text.[15]

Another interesting observation is that the *PhraSAD* (dual classifier) outperformed the *PhraSAD* (single classifier) by 1.86% improvement on **CFSC**. The reason is that the dual classification alleviated the difficulty of identifying neutral data. Therefore, by default, we used the dual-classifier on **CFSC** and the single-classifier on **CTRD** and **TWITTER** in the following experiments.

Role of Pinyin for CFSC: the model with the additional Pinyin feature is worse than the model without a Pinyin feature. The Pinyin feature is input into the model as a separated vector. However, the result shows that the accuracy was decreased. Therefore, the Pinyin branch is not part of the final *PhraSAD* model. See Table 4. Further, there are other two possible reasons which can explain the inefficiency of the Pinyin feature. For one thing, homophone is popular in Chinese. This leads to an inexplicit aspect of Pinyin. For another, using Pinyin almost doubled the length of the inputs, which can be a burden for networks with a fixed layer number. In other words, the Pinyin feature dispersed representations, just like the segmentation in previous studies [16].

Contribution of Components: In order to analyze the effectiveness of each component of the *PhraSAD*, we executed the ablation test. We removed the self-attention, the lexical branch and the phrase branch, respectively, from the *PhraSAD*. The Pinyin branch is not used in this test because the accuracy will drop with it. Moreover, because segmentation is a special problem for Chinese text, we compared with a model without segmentation too.

[15] We fine-tuned the BERT-based, Chinese [6] model with our corpus (i.e., the CFSC). After 30 epochs, the accuracy is 74.5%. However, given more resources and times, the result could be better. This is not listed in the table because one motivation of this paper is to discover explanations.

Table 2 Overall evaluation results (accuracy)

Datasets models	CFSC	CTRD	TWITTER (10 k)	TWITTER (30 k)
THUCTC	62.11	81.03	-	-
CNN	61.62	84.05	76.9	77.45
LSTM/BiLSTM	65.57 / 66.96	86.91 / 87.82	76.4 / 76.75	76.53/77.78
CNN-BiLSTM	67.18	87.95	80.55	80.98
RCNN-LSTM	63.71	85.34	77.35	77.28
MBCNN-LSTM	65.31	87.43	78.45	78.97
Attention	62.64	84.03	75.65	76.73
PhraSAD (dual classifier)	**84.11**	–	–	–
PhraSAD (single classifier)	82.24	**89.28**	**83.03**	**83.49**

Table 3 Ablation testing results (accuracy) for contributions of different components with dual classifier

Datasets models	CFSC	CTRD	TWITTER (10 k)	TWITTER (30 k)
PhraSAD (dual classifier)	**84.11**	-	-	-
PhraSAD (dual classifier with additional Pinyin feature)	80.40	-	-	-

Table 4 Role of pinyin (accuracy)

Datasets models	CFSC	CTRD	TWITTER (10 k)	TWITTER (30 k)
PhraSAD no attention	82.33	87.78	81.9	82.17
PhraSAD no lexical	81.54	87.57	78.29	78.35
PhraSAD no phrase	80.42	87.32	78.85	79.21
PhraSAD no segmentation	79.70	87.98	–	–

Accordingly, the new models were denoted as *PhraSAD* no attention, *PhraSAD* no lexical, *PhraSAD* no phrase and *PhraSAD* no segmentation, respectively. It was impossible to discard the word branch in any case. Results are presented in Table 3.

It turns out that each component had a positive impact on the whole performance of the *PhraSAD*. The accuracy declined when we removed any branch. Using three branches jointly performed better than using any single branch or any pair of two branches. Interestingly, the accuracy dropped the most when we tested the *PhraSAD* no phrase model. It showed that the phrase branch contributed most to the proposed model. Concretely, comparing Tables 2 and 3, we know that for English **TWITTER** dataset the phrase structure improves the accuracy by at least (83.03–78.85=)4.18%, while for Chinese dataset **CTRD** the phrase structure improves the accuracy by (89.28–87.32=)1.96% and for **CFSC** this improvement turned out to be (84.11–80.42=)3.69%. Additionally, it is arguable whether segmentation is principal for Chinese texts. In our experiments, it is the most important component for the **CFSC** dataset as well as the least important component for the **CTRD** dataset.

Different Approaches for Phrase Embedding: We compared our weighted-sum method (i.e., Algorithm 1) with the other two phrase embedding methods. They take either a single sum or the average on the embedding of words contained in a phrase. As shown in Fig. 4, our method outperformed compared methods in all datasets. Even though using syntax trees is not novel, the purpose of using syntax trees in this paper is novel. This new phrase embedding generation approach based on syntax trees outperformed the existing simple-sum approaches by a substantial margin.

Position of Self-attention Layer: At last, we compared the effect of placing the self- attention layer on a different position in our deep network. As shown in Fig. 5, we found that in all datasets, the accuracy of Self-Attention+CNN+BiLSTM was higher than that of CNN+Self-Attention+BiLSTM. Because after CNN extracts local features, the data becomes blurred. Therefore, it is difficult to calculate the relevance in the next attention layer. This leads to a degradation of the accuracy.

There are previous studies which used word attention [8] or attention model [14] on review datasets. However, our approach is different with them in three major points. First, studies like Du [8] only use attention on word but *PhraSAD*

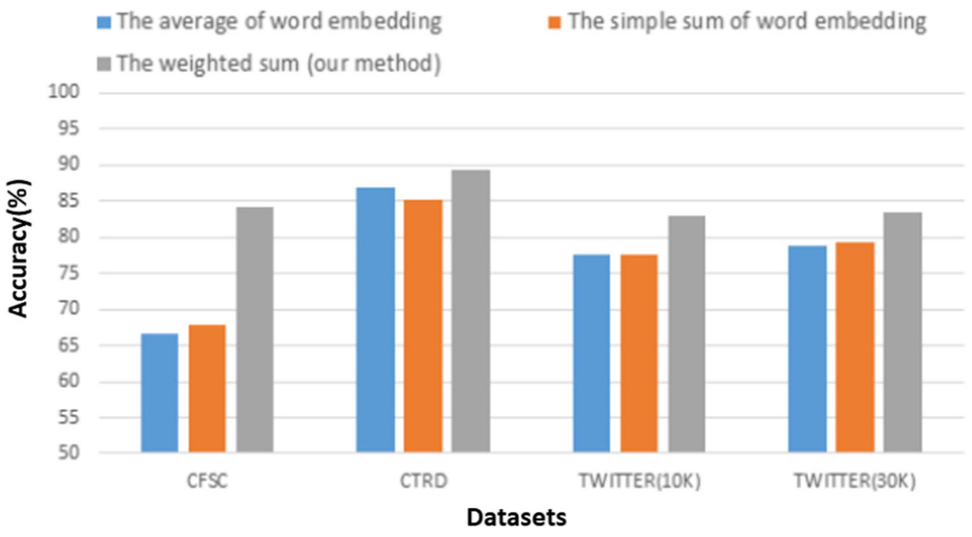

Fig. 4 Results of different phrase embedding generation methods

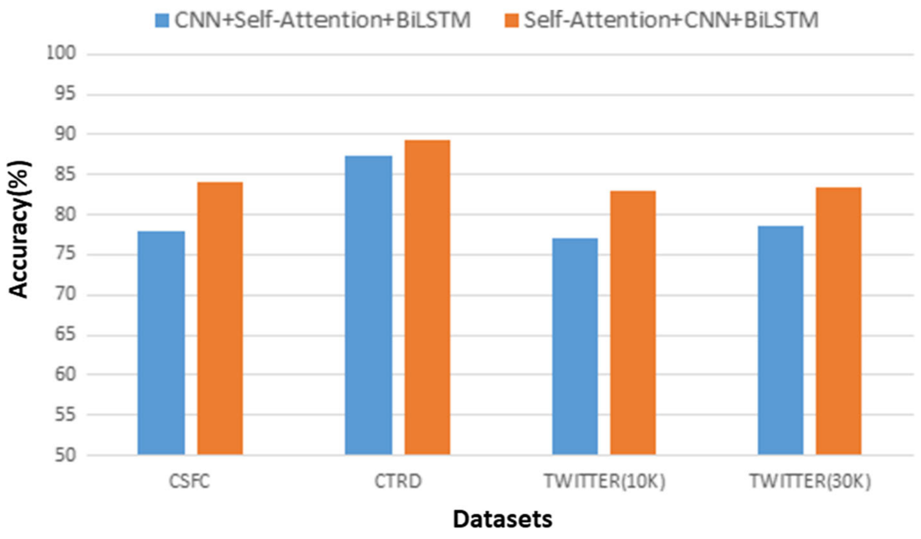

Fig. 5 Results of different self-attention layers

uses attention on all features. Second, the features considered by Li [14] are only semantic-related while the *PhraSAD* take the Pinyin, segmentation, lexical analysis and phrase structure into account. Therefore, we did an ablation experiment on the *PhraSAD*. Third, they focus on long review text for products [8] or movies [14] while we pay more attention on financial short texts. That is why we had to build a corpus (**CFSC**) while previous studies can use long existing corpus [8, 14]. Because some previous studies [8, 15] did not release their code, we cannot compare their network with the *PhraSAD* in this paper.

5 Conclusion

This paper focuses on explaining the effectiveness of features for deep learning in Chinese financial short-text sentiment analysis tasks. Each data item was labeled by financial experts thus it is highly credible. For evaluation, we build a new Chinese short-text corpus. It was used along with two other existing short-text datasets (**CTRD** and **TWITTER**) in our experiments. Our experiments showed that phrase structure with self-attention models is two of the best. A major contribution of this paper is the final model, which preformed best for all corpus in the experiments, especially for **CFSC**.

However, this should not be the end of the story. There are promising directions of our future work, which are listed in Table 5 with failure cases.

- For example, managing abbreviations and buzzwords. A case in point is a positive sentence “打死不割肉 (Never cut position),” which includes a buzzword “割肉(cut position).” This will be helpful to identify complete semantic information of short texts.

Table 5 Promising directions with failure cases

Future work	Failure case	Label	Prediction
Managing abbreviations and buzzwords	打死不割肉(Never cut position)	Positive	Negative
Use different weights for different types of phrases	"怎么涨起来了?楼主你不是说跌停吗?"(Why the stock rise while you said it would fall?)	Negative	Neutral
Consider advanced grammatical representation	"今天完完全全满仓了, 啥也不想, 啥也不看了　任你风吹雨打, 我自岿然不动 (Today I turned all my capital into securities, I don't think about anything, and I didn't watch anything. Let the wind and rain beat me, I won't move.)	Positive	Neutral
Facilitate Character-based features	"已经果断清仓, 明天再看低位做t"(I have sold all my stock, and I will buy at low position tomorrow, but that depends.)	Negative	Neutral

- For another example, it is more reasonable to use different weights for different types of phrases. "怎么涨起来了?楼主你不是说跌停吗?" is a stereotype positive sentence, which means "Why the stock rise while you said it would fall?". Although there are a rise and a fall, the fall is only a statement about past facts. To do this, we should use lexical code and phrase structures jointly to produce phrase embeddings.
- Yet another example is to consider more advanced grammatical representation. One candidate is tree bank provided by CoreNLP. Phrases usually are instantiated, while tree bank provides an abstract structure. This will alleviate the structured sparsity problem of short texts, see [37] for more details about the structured sparsity in NLP. An example is a positive sentence "今天完完全全满仓了，啥也不想，啥也不看了。任你风吹雨打，我自岿然不动" with 5 sparse phrases, which means "Today I turned all my capital into securities, I don't think about anything, and I didn't watch anything. Let the wind and rain beat me, I won't move."
- At last, character-based features [41] can be taken into the consideration via extending channels in our model. A negative sample is "已经果断清仓, 明天再看低位做t," which means "I have sold all my stock, and I will buy at low position tomorrow, but that depends." The character sequence "看低位做t," which means "I will buy at low position tomorrow, but that depends," is a neutral sentence with positive words.

Compliance with ethical standards

Conflict of interest There is no conflict of interest between the authors to publishing this article.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv: 1409.0473 (2014)
2. Batra, R., Daudpota, S.M.: Integrating StockTwits with sentiment analysis for better prediction of stock price movement. In: Computing, Mathematics and Engineering Technologies (iCoMET). pp. 1–5. IEEE (2018)
3. Chen J, Yan S, Wong KC (2018) Verbal aggression detection on twitter comments: Convolutional neural network for short-text sentiment analysis. Neural Comput Appl 32:1–10
4. Chen T, Xu R, He Y, Xia Y, Wang X (2016) Learning user and product distributed representations using a sequence model for sentiment analysis. IEEE Comput Intell Mag 11(3):34–44
5. Cho, K., Van Merrie¨nboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint:1406.1078 (2014)
6. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR abs/1810.04805 (2018), arXiv: 1810.04805
7. Dong L, Wei F, Liu S, Zhou M, Xu K (2015) A statistical parsing framework for sentiment classification. Comput Linguist 41 (2):293–336
8. Du Y, Zhao X, He M, Guo W (2019) A novel capsule based hybrid neural network for sentiment classification. IEEE Access 7:39321–39328
9. Kaplanski G, Levy H (2010) Sentiment and stock prices: The case of aviation disasters. J Financ Econ 95(2):174–201
10. Kim, Y.: Convolutional neural networks for sentence classification. In EMNLP pp.1746–1751 (2014)
11. La Su, Y., Liu, W.W., et al.: Research on the LSTM Mongolian and Chinese machine translation based on morpheme encoding. Neural Computing and Applications pp. 1–9 (2018)
12. Lazaridou, A., Titov, I., Sporleder, C.: A Bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 1630–1639 (2013)
13. Li, J., Sun, M., Zhang, X.: A comparison and semi-quantitative analysis of words and character-bigrams as features in Chinese text categorization. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. pp. 545–552 (2006)

14. Li, L., Goh, T.T., Jin, D.: How textual quality of online reviews affect classification performance: a case of deep learning sentiment analysis. Neural Computing and Applications pp. 1–29 (2018)
15. Li W, Liu P, Zhang Q, Liu W (2019) An improved approach for text sentiment classification based on a deep neural network via a sentiment attention mechanism. Future Internet 11(4):96
16. Li, X., Meng, Y., Sun, X., Han, Q., Yuan, A., Li, J.: Is word segmentation necessary for deep learning of Chinese representations? In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 3242–3252. Association for Computational Linguistics, Florence, Italy (Jul 2019), https://www.aclweb.org/anthology/P19-1314
17. Lin, Z., Feng, M., Santos, C.N.d., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. arXiv preprint arXiv: 1703.03130 (2017)
18. Liu, P., Qiu, X., Huang, X.: Dynamic compositional neural networks over tree structure. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017. pp. 4054–4060 (2017), https://doi.org/10.24963/ijcai.2017/566
19. Liu, Y., Chen, Y.: Research on Chinese micro-blog sentiment analysis based on deep learning. In: Computational Intelligence and Design (ISCID). vol. 1, pp. 358–361. IEEE (2015)
20. Long W, Tang Yr, Tian Yj (2018) Investor sentiment identification based on the universum SVM. Neural Comput Appl 30 (2):661–670
21. Luong MT, Frank MC, Johnson M (2013) Parsing entire discourses as very long strings: Capturing topic continuity in grounded language learning. Transactions of the Association of Computational Linguistics 1:315–326
22. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations. pp. 55–60 (2014), http://www.aclweb.org/anthology/P/P14/P14-5010
23. Mnih, V., Heess, N., Graves, A., et al.: Recurrent models of visual attention. In: Advances in neural information processing systems. pp. 2204–2212 (2014)
24. Nagarajan SM, Gandhi UD (2019) Classifying streaming of twitter data based on sentiment analysis using hybridization. Neural Comput Appl 31(5):1425–1433
25. Ouyang, X., Zhou, P., Li, C.H., Liu, L.: Sentiment analysis using convolutional neural network. In: Computer and Information Technology. pp. 2359–2364. IEEE (2015) Peng, H.: Linguistic-inspired Chinese sentiment analysis: from characters to radicals and phonetics. Ph.D. thesis (2019)
26. Ruan, X., Wilson, S., Mihalcea, R.: Finding optimists and pessimists on twitter. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). vol. 2, pp. 320–325 (2016)
27. Silhavy R, Senkerik R, Oplatkova ZK, Silhavy P, Prokopova Z (2016) Artificial Intelligence Perspectives in Intelligent Systems. Springer, Berlin
28. Tai, K.S., Socher, R., Manning, C.D.: 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 2015, July 26–31, 2015, Beijing, China, Volume 1: Long Papers. pp. 1556–1566 (2015), http://aclweb.org/anthology/P/P15/P15–1150.pdf
29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008 (2017)
30. Vilares D, Alonso MA, Gomez-Rodrıguez C (2015) A syntactic approach for opinion mining on Spanish reviews. Natural Language Engineering 21(1):139–163
31. Wang, J., Wang, Z., Zhang, D., Yan, J.: Combining knowledge with deep convolutional neural networks for short text classification. In: Proceedings of IJCAI. vol. 350 (2017)
32. Wang L, Niu J, Song H, Atiquzzaman M (2018) Sentirelated: A cross-domain sentiment classification algorithm for short texts through sentiment related index. J Netw Comput Appl 101:111–119
33. Wu L, Hoi SC, Yu N (2010) Semantics-preserving bag-of-words models and applications. IEEE Trans Image Process 19(7):1908–1920
34. Wu L, Morstatter F, Liu H (2018) Slangsd: building, expanding and using a sentiment dictionary of slang words for short-text sentiment classification. Language Resour Eval 52(3):839–852. https://doi.org/10.1007/s10579-018-9416-0
35. Yang Q, Rao Y, Xie H, Wang J, Wang FL, Chan WH, Cambria C (2019) Segment-level joint topic-sentiment model for online review analysis. IEEE Intell Syst 34(1):43–50
36. Yenter, A., Verma, A.: Deep CNN-LSTM with combined kernels from multiple branches for imdb review sentiment analysis. In: Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON). pp. 540–546. IEEE (2017)
37. Yogatama, D., Smith, N.A.: Linguistic structured sparsity in text categorization. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 786–796 (2014)
38. Yu Y, Duan W, Cao Q (2013) The impact of social and conventional media on firm equity value: A sentiment analysis approach. Decis Support Syst 55(4,SI):919–926
39. Zeng, J., Li, J., Song, Y., Gao, C., Lyu, M.R., King, I.: Topic memory networks for short text classification. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 3120–3131 (2018), https: //aclanthology.info/papers/D18–1351/d18–1351
40. Zhang H, Huang W, Liu L, Chow TW. Learning to match clothing from textual feature-based compatible relationships. IEEE Transactions on Industrial Informatics. 2019 Jun 24.
41. Zhang H, Li J, Ji Y, Yue H (2016) Understanding subtitles by character-level sequence-to-sequence learning. IEEE Trans Industr Inf 13(2):616–624
42. Zhu, X., Sobhani, P., Guo, H.: Dag-structured long short-term memory for semantic compositionality. In: NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12–17, 2016. pp. 917–926 (2016), http://aclweb.org/anthology/N/N16/N16-1106.pdf