Sentiment Analysis of Chinese Microblog Based on Stacked Bidirectional LSTM

Junhao Zhou, Yue Lu, Hong-Ning Dai, *Senior Member, IEEE*, Hao Wang, *Member, IEEE* and Hong Xiao

***Abstract*—Sentiment analysis on Chinese microblogs has received extensive attention recently. Most previous studies focus on identifying sentiment orientation by encoding as many word-properties as possible while they fail to consider contextual features (e.g., the long-range dependencies of words), which are however essentially important in the sentiment analysis. In this paper, we propose a Chinese sentiment analysis method by incorporating Word2Vec model and Stacked Bidirectional long short-term memory (Stacked Bi-LSTM) model. We first employ Word2Vec model to capture semantic features of words and transfer words into high dimensional word vectors. We evaluate the performance of two typical Word2Vec models: Continuous Bag-of-Words (CBOW) and Skip-gram. We then use Stacked Bi-LSTM model to conduct the feature extraction of sequential word vectors. We next apply a binary softmax classifier to predict the sentiment orientation by using semantic and contextual features. Moreover, we also conduct extensive experiments on real dataset collected from Weibo (i.e., one of the most popular Chinese microblogs). The experimental results show that our proposed approach achieves better performance than other machine learning models.**

***Index Terms*—Long short-term memory (LSTM), Stacked bi-directional LSTM, Sentiment analysis, Continuous Bag-of-Words, Chinese MicroBlog, Contextual features**

I. Introduction

Social media such as microblogs are becoming the most important sources of real-time news and reflect the public opinions or sentiments on special events. As an alternative to Twitter in China, Sina Weibo[1] is a major microblogging service in China. In the past few years, Weibo (aka Chinese microblog) has gained tremendous popularity. According to the first season financial report of Sina Weibo in 2017, there are more than 340 million monthly active users in Weibo. Moreover, the increasing Internet popularity and the appearance of "We-Media" greatly inspire the enthusiasm of the Internet users to freely express their opinions on microblogs. Nowadays, Weibo is not only a platform for people sharing their daily life or anecdotes, but also becomes an important medium for people to express views or feelings on specific news, events or products and engage in the discussions with other users. Therefore, how to effectively extract useful information from those short microblog texts becomes a hot research topic.

A. Motivation

Sentiment analysis on microblogs has received extensive attention recently since the valuable features obtained from sentimental analysis can be used in a broad range of applications, such as opinion detection, political promotion and decision making. In particular, Weibo has become an important source for Chinese sentiment study since the anonymity of Weibo makes people being willing to express their real sentiments. Many existing techniques of sentiment analysis are mainly based on sentiment lexicons and traditional feature engineering [1]–[8]. Most of these methods need resort to external resource or manually preprocess features of words.

However, sentiment analysis on microblogs has several challenges.

1) For a sentiment analysis task, the same word in different contexts may express opposite orientations.
2) Because of the colloquial writing style of Weibo, sentiments are usually expressed by abbreviations or slang words rather than formal sentimental words, consequently resulting in the challenge in sentiment analysis.
3) The microblog data is usually length-limited, sparse, and fragmented. The microblog content can be composed of several words or even just a sentence. The microblog data has many implicit context features that are usually hard to discovery.

For example, Examples 1 and 2 show that microblog posts contain emotional slang terms.

Example 1: "为祖国疯狂打 call！" (English translation: "Cheer for my country!")

Example 2: "陈独秀，请你坐下！" (English translation: "Your idea was quite brilliant!")

The above examples indicate that slang terms are critical for sentiment expression in different cases. Therefore, it is impossible to manually collect and design features for the slang terms one by one; this nevertheless poses a challenge for sentiment analysis of Weibo. Although there are a number of studies concentrating on dealing with the sentiment analysis task by enriching semantic features of Chinese words [9]–[15], few studies explore the long-distance dependencies between Chinese words. Essentially, the long-distance dependencies

This work is partially supported by National Natural Science Foundation of China(NFSC) (No.61672170); NSFC-Guangdong Joint Fund (No.U1401251); the Science and Technology Planning Project of Guangdong Province (No.2015B090923004 and No. 2017A050501035), Science and Technology Program of Guangzhou (No. 201807010058), Guangdong Science and Technology Plan (No. 2015B090923004). (*Corresponding author: H.-N. Dai and H. Wang)*

J. Zhou, Y. Lu and H.-N. Dai are with Faculty of Information Technology, Macau University of Science and Technology, Macau SAR (email: junhao_zhou@qq.com; yuelu_@hotmail.com; hndai@ieee.org)

H. Wang is with Department of Computer Science, Norwegian University of Science and Technology, Gjøvik, Norway (email: hawa@ntnu.no)

H. Xiao is with Faculty of Computer, Guangdong University of Technology, Guangzhou, China (email: wh_red@gdut.edu.cn)

[1]http://weibo.com/

constitute to the *contextual features* of a post, which is important to determine the sentiment orientation. Consider another example as given in Example 3:

Example 3: "为什么要这么苛刻呢？8 分钟展现出这么多中国元素，中国科技，展现出中国的热情和自信。张艺谋导演真的是鞠躬尽瘁了。搞不懂这些人！" (English translation: "Why are they so mean? This 8-minute show exhibited so many Chinese elements, Chinese technologies as well as our people's enthusiasm and confidence. The director Yimou Zhang has already tried his best. I really can't understand these people!")

Example 3 excerpts a comment about the Beijing's eight-minute show at the closing ceremony of the 2018 Pyeong Chang Winter Olympics. In Example 3, the first sentence expresses a clear negative sentiment toward the critics of the 8-minute show. Then the middle two sentences applauds the design and the effort made by the director Yimou Zhang (who is one of the most successful directors in China). Even if the middle two sentences seem to have inconsistent sentiments with the previous sentence, they only serve to emphasize the preceding topic "These people are too hard on the director Yimou Zhang". Therefore, with the reference of the context, we can infer that the sentiment orientations of the middle two sentences are also negative. The last sentence expresses the same sentiment orientation as the first sentence. As a result, we can conclude that the sentiment orientation of Example 3 is negative. From this example, we can find that the sentiment of a post is highly *context-dependent*. However, most existing methods are insensitive to the contextual information and fail to handle these long-distance dependencies of Chinese words; this constitutes another challenge for sentiment analysis.

B. Contributions

To address the above challenges, in this paper, we put forth a Chinese sentiment analysis method that incorporates a Continuous Bag-of-Words (CBOW) model and a Stacked bi-directional Long short-term memory (Stacked Bi-LSTM) to extract both word semantic features and word sequence features for Chinese sentiment analysis. *To the best of our knowledge, we are the first to use the composite method in Chinese sentiment analysis*.

The main contributions of this paper can be summarized as follows:

- We originally put forth a Chinese sentiment analysis method with integration of CBOW and Stacked Bi-LSTM models. Our proposed model gains the benefits from CBOW and Stacked Bi-LSTM such as capabilities of learning rich semantic information and rich contextual information effectively.
- We then apply the proposed model to analyze Chinese microblog's sentiment. We conduct extensive experiments on real dataset collected from Weibo via the crawler developed by ourselves. This real dataset consists of 3,000 annotations of the microblog comments. There are 1,514 comments labeled as positive and 1,486 comments labeled as negative, respectively.
- We evaluate performance of our model comprehensively. We also evaluate the impact of different parameters in our model, such as word embedding models (CBOW and Skip-gram), sentence lengths. The experimental results show that our proposed approach outperforms other machine learning methods.

The remainder of this paper is organized as follows. Section II reviews related works on sentiment analysis. In Section III, we describe the main approaches that we adopt. The details of the proposed model are presented in section IV. Then the empirical results are discussed in Section V. Finally, we conclude our work and discuss future research direction in Section VI.

II. Related Work

This section reviews recent advances in Chinese sentiment analysis. We roughly categorize the existing studies into two types: 1) Sentiment analysis based on traditional machine learning methods and 2) Sentiment analysis based on deep learning approaches.

A. Sentiment analysis based on traditional method

The methods on Chinese sentiment analysis can be roughly subdivided into two categories: i) sentiment lexicon based methods and 2) machine learning methods. The main idea of the sentiment lexicon based methods is to use a list of sentiment words with *polarities* and *intensities* as important references to identify the sentiment orientation of words. To improve the effectiveness of the lexicon based methods, it is necessary to gather as many sentiment lexicons as possible. In particular, the work in [1] built a microblog-specific Chinese sentiment lexicon based on existing lexicons for subjectivity detection and sentiment polarity classification. Zhang et al. [2] combined degree adverb dictionary, negative word dictionary, network word dictionary and other related dictionaries to further extend the sentiment dictionary. Moreover, many studies combined sentiment lexicon with supervised machine learning methods. For example, Li et al. [3] employed Vector Space Model (VSM) and Term Frequency–Inverse Document Frequency (TF-IDF) to represent texts and extracted sentiment features via the aid of sentimental lexicons. In addition, this method also combines two features as inputs of machine learning models. Chen et al. [4] treated sentiments target extraction as a sequence labeling problem. They solved the problem by incorporating opinion lexicon and other features into Conditional Random Fields (CRF) model.

Besides sentimental lexicon, various types of supervised machine learning methods such as Naive Bayes (NB), Support Vector Machines (SVM), Maximum Entropy (ME), Logistic Regression (LR), etc. and feature combinations have been applied in sentiment analysis research. In particular, Zheng et al. [5] and Ficamos et al. [6] adopted part-of-speech (POS) n-grams features of machine learning models. Liang et al. [7] proposed an Auxiliary-Sentiment Latent Dirichlet Allocation (AS-LDA) model for sentiment classification. Particularly, the model exploited targets of the opinion, polarity words and modifiers of polarity words as sentiment orientation. Li et al. [8] applied TF-IDF and Latent semantic analysis (LSA) to extract features for machine learning models.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access

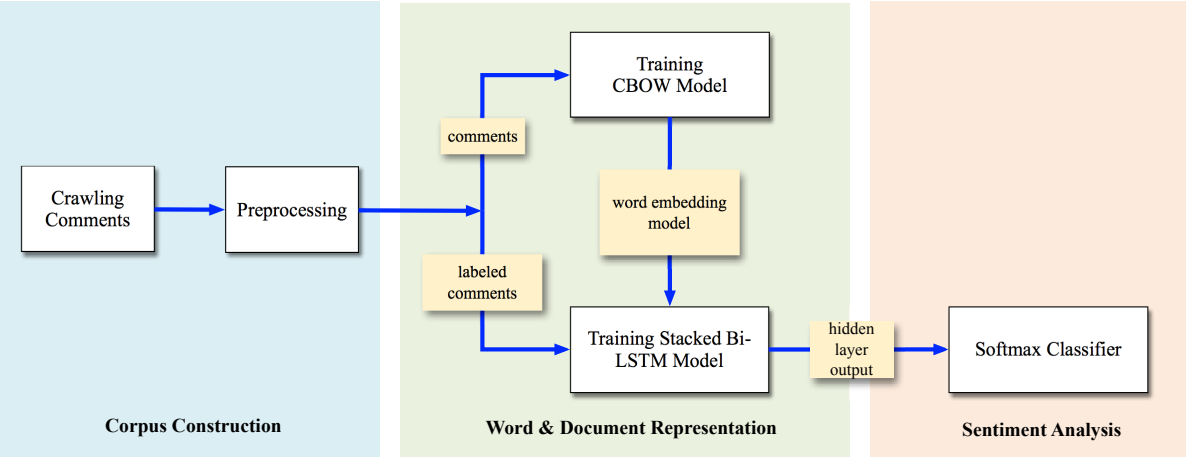


Fig. 1: Overview of methodology

Most of these methods have to use many external resources or manually-configured features for sentiment recognition. However, there are a limited number of lexicons for Chinese sentiment study. Moreover, daily updated buzzwords used in microblogs are difficult to be completely collected; this would restrict the scale of study. In addition, most methods can hardly encode semantic features of words as they just relied on explicit properties of words. Considering the colloquial style of microblogs, the sentiments are not always expressed in a usual way. For instance, it can be expressed by abbreviations or slang words rather than formal words. All these issues result in challenges in the sentiment analysis study.

B. Sentiment analysis based on deep learning

Word embedding techniques based on neural networks can overcome the difficulties of traditional word representation methods. In particular, they can encode the semantic and syntactic properties of words to provide relatively precise information for document representation. In recent years, applying word embedding models to sentiment analysis studies has received extensive attention. For example, Xue et al. [9] used Word2vec tools to extent sentiment lexicons and classified sentences according to a Sentiment Orientation Pointwise Similarity (SO-SD) model. Zhang et al. [10] and Lu et al. [11] also extended sentiment lexicon with word embedding models and introduced supervised SVM models for sentiment classification. Hao et al. [12] employed a Dynamic Conditional Random Field for subjectivity and polarity classification via using Word2vec technologies. In addition to non-neural models, neural networks based approaches can be used for documents representation without any designed features. Cao et al. [13] combined Continuous Bag-of-Words (CBOW) model [16] with a Convolutional Neural Network (CNN) to extract features of paragraphs as inputs of an SVM classifier. Wang et al. [14] proposed a multi-label classification model by using skip-gram for word embedding and CNN for sentiment representation. Chang et al. [15] constructed a Distributed Keyword Vector (DKV) model using keyword vectors and a single-layer neural network for document representation consequently referring to author information for measuring regional prejudice.

The previous studies considered semantic features of words to represent documents. However, the *long-range dependencies* of words (contextual features) have been typically ignored. Moreover, most of the methods need manually extracting features of contexts and collecting user information; this process is arduous and can hardly apply to huge amounts of data. To address these issues, Recurrent Neural Network (RNN) and Bi-directional Long Short Term Memory (Bi-LSTM) [17]–[19] Network were introduced to automatically extract the features of context [20], [21]. In these studies, document features were abstracted by single-layer networks. However, the features of sequential high-dimensional word vectors can be quite complex; this process may not be sufficient to be completed for one time.

In this paper, we propose a Stacked Bi-LSTM method with stacking more layers of RNN to conduct the sentiment analysis task for Chinese microblogs. One of advantages of our model is that it does not resort to any external resources or manual annotation to learn semantic and syntax features. Therefore, it has universal applicability. Furthermore, our model is capable of deeply extracting complex features of documents consequently providing more precise information for sentiment analysis task.

III. Overview of Methodology

Figure 1 shows the proposed method used for sentiment analysis of Chinese microblog. Firstly, the microblog texts (comment or status) can be obtained from Weibo by a crawler. In this paper, we select comments of microblogs as the corpus, and regard each of the comments as a document. Then several preprocessing steps can be carried out to process these texts. Once the texts have been prepared, all pre-treated texts are then fed into Continue Bag-of-Words (CBOW) model to map words of each text into a high dimensional vector. After that, Stacked Bi-directional Long Short Term Memory (Stacked Bi-LSTM) model utilizes sequential word vectors transformed from word embedding model as inputs to represent the labeled texts with contextual features. Lastly, a softmax classifier

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access

Rules	Raw Text	Processed Text
Removing Html Tags	**可是说是非常震撼和有科技感**<img src="//h5.sinaimg.cn/m/emoticon/icon/default/d_xingxingyan-34ecaee809.png" style="width:1em;height:1em;" alt="[憧憬]"><img src="//h5.sinaimg.cn/m/emoticon/icon/default/d_xingxingyan-34ecaee809.png" style="width:1em;height:1em;" alt="[憧憬]">**2022 相约北京**<img src="//h5.sinaimg.cn/m/emoticon/icon/others/l_xin-8e9a1a0346.png" style="width:1em;height:1em;" alt="[心]"><img src="//h5.sinaimg.cn/m/emoticon/icon/others/l_xin-8e9a1a0346.png" style="width:1em;height:1em;" alt="[心]">	**可是说是非常震撼和有科技感 2022 相约北京**
Removing Html Tags Removing Reply Symbol Removing @User	回复@千叼:**快让我插会儿腰，我们祖国太值得骄傲了**<img src="//h5.sinaimg.cn/m/emoticon/icon/others/l_xin-8e9a1a0346.png" style="width:1em;height:1em;" alt="[心]">	**快让我插会儿腰，我们祖国太值得骄傲了**
Removing Html Tags Removing HashTags	#北京第九分钟# **平昌冬奥结束啦~张艺谋导演又为我们带来了新时代的北京八分钟，真是振奋人心！2022我们在北京，不见不散~**<img src="//h5.sinaimg.cn/m/emoticon/icon/others/l_xin-8e9a1a0346.png" style="width:1em;height:1em;" alt="[心]">	**平昌冬奥结束啦~张艺谋导演又为我们带来了新时代的北京八分钟，真是振奋人心！2022我们在北京，不见不散~**

Fig. 2: An illustration of data preprocessing rules with examples

extracts information from the last hidden layer and conducts the prediction of sentiments for labeled texts.

A. Corpus Construction

Weibo provides us an open API[2] to crawl comments. We can simply get comments of one status by parsing the JavaScript Object Notation (json) files. After we get the raw texts from the json file, we first employed BeautifulSoup[3] to remove html tags in the raw texts, and then preprocessed of these texts. Microblog texts are different from traditional texts since they usually include some noises, such as hash-tags, reply symbols and references to user names (e.g., @user) and links. Figure 2 shows three examples after removing html tags, reply symbol, references and hash-tags in the comments. Meanwhile, we also filter out some topic-irrelevant comments such as "Repost", "Comment with pics" to reduce the amount of irrelevant features.

We then segment processed comments and remove the stop words. Specifically, we employ Jieba[4], a python library for Chinese text segmentation. Then we build a list of 1,946 Chinese stop words, including special symbols, numbers, English characters and Chinese words. Finally, we filter out all stop words in the segmented texts. Table I shows three examples after segmenting and processing stop words corresponding to the three examples in Figure 2.

B. Word Representation

The basic idea of word representation models is to map words into a high dimensional vector. The distance of word

[2]http://m.weibo.cn/api/comments

[3]https://pypi.org/project/beautifulsoup4/

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

TABLE I: An illustration of Chinese segmentation and stop word processing

Processed Text	Segmented Text
可是说是非常震撼和有科技感 2022 相约北京	说 震撼 科技 感 2022 相约 北京
快让我插会儿腰，我们祖国太值得骄傲了	插 会儿 腰 祖国 太 值得 骄傲
平昌冬奥结束啦 ～ 张艺谋导演又为我们带来了新时代的北京八分钟，真是振奋人心！2022 我们在北京，不见不散 ～	平昌 冬奥 结束 张艺谋 导演 带来 新 时代 北京 八分钟 振奋人心 2022 北京 不见不散

vectors in the space depends on their semantic or contextual similarity. In [16], CBOW model was introduced to learn word embedding based on the context of current words in the range of specific window size. To train this model, we first set the window size to 5 words and get contexts of current words from segmented comments within 5-word limits. The acquired contexts then enter into CBOW model and the current words become the features to be predicted by the proposed model. The CBOW algorithm is modified from the open source toolkit provided by Google[5]. The authors in [22], [23] showed that the top performance can be achieved in paragraph vector at 100 dimensions. Similarly, we adopt 100-dimensional word embeddings for Chinese microblog after the training.

C. Document Representation

In contrast to word representation models, document representation models are designed to capture the features of documents based on the contextual information. To train this model, we first normalize the length of network input after cutting down the exceeding words for inputs when the length of sentence is greater than K. Otherwise, we use zero padding to replenish inputs. Therefore, the inputs can keep most of context information. Then we get a sequential word vector by replacing words in a comment with embedding vectors. Next, Stacked Bi-LSTM model is employed to learn the features of sequential word vectors. Finally, we extract the value of the last hidden layer to represent documents. Since the inputs of the model are word-embedding vectors, the final document vectors contain both semantic information and contextual information. The details of Stacked Bi-LSTM are presented in Section IV.

D. Sentiment Analysis

We then apply a softmax classifier that uses the last output of Stacked Bi-LSTM model to conduct the sentiment prediction task. We employ the *cross entropy* to evaluate the difference between the predicted value and the real value, and calculate the loss value for each comment. Based on the loss values, we use Adam optimizer [24] to optimize parameters of the model. The details of the calculation of the sentiment prediction are discussed in Section IV-C.

IV. Sentiment Analysis Based on Stacked Bi-LSTM

This section presents the sentiment analysis based on Stacked Bi-LSTM model. Figure 3 illustrates the basic architecture of 2-layer Stacked Bi-directional LSTM to model

[5]https://code.google.com/p/word2vec

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access

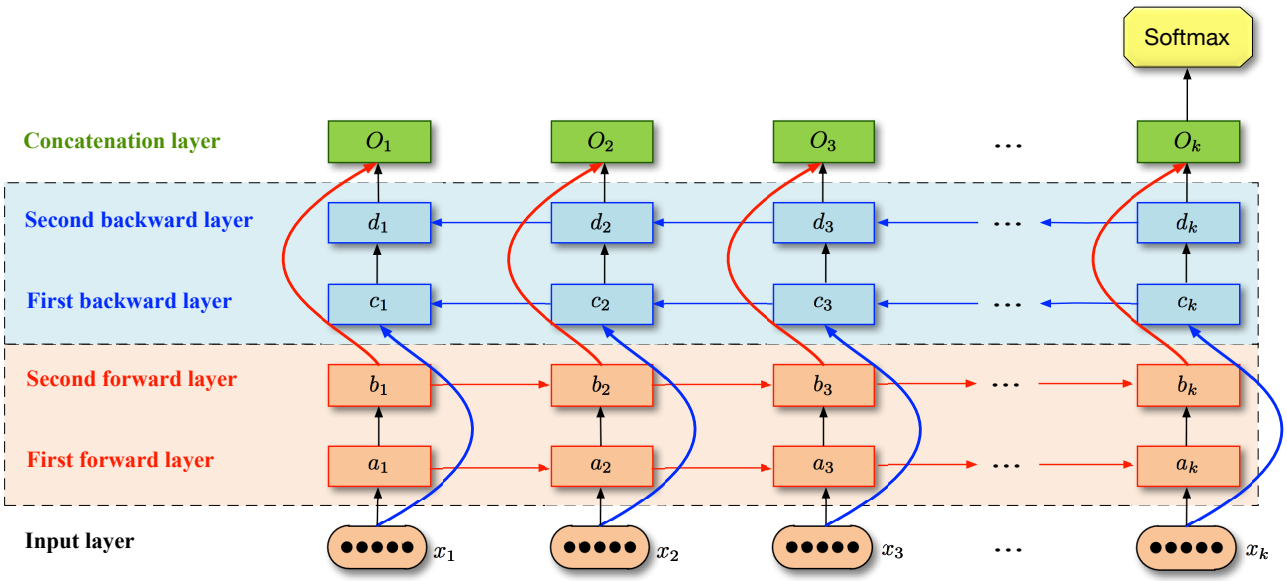


Fig. 3: An illustration of sentiment analysis model based on 2-layer Stacked Bi-LSTM

for sentiment analysis. Given a batch of N comments and each comment with K words, let $X = \{X_1, X_2, ..., X_N\}$ be a set of comments in a batch and $X_n = \{X_1, X_2, ..., X_K\}$ be a set of words in any comment X_n. It is worth mentioning that any word X_k as shown in Figure 3 is a D-dimensional embedding word vector. The goal of this model is to predict sentiment Y for each comment. We also screen out some comments with neutral, objective or conservative views in the beginning to make sure that all inputs have non-ambiguous sentiments. In other words, for any comment, its sentiment Y_n can only be negative or positive. In addition, both features of comments either negative or positive, appear as discrete values. Therefore, one hot is needed to encode these discrete features. We map 2 features of a comment into 2 bits one hot code. Specifically, $[1, 0]$ represents negative and $[0, 1]$ represents positive. After encoding, we can get sentiment label $Y_n (Y_n \in \{[1, 0], [0, 1]\})$ for any comment X_n.

There are three steps for sentiment prediction using Stacked Bi-LSTM model. Firstly, we perform the preprocessing of input comments. Secondly, we build a 2-layer Bi-LSTM model to deeply extract the contextual features to represent documents. Thirdly, we put the outputs of the last hidden layer into a softmax classifier to conduct the sentiment prediction. We will discuss these 3 steps in detail in the following subsections.

A. Preprocessing of Inputs

First of all, we figure out the average sentence length, which is represented as K, as a reference of the maximum size of network inputs. To make sure our model can get equal-sized inputs, we have to normalize the length of comments. For a comment having more than K words, we cut down inputs of the exceeding words. In another case, if the length of a comment is less than K, we replenish it with zero until its length reach K. Afterwards, we replace K words in comment with D dimensional vector values. If a word has no corresponding vector in the word embedding model, we replace this word with a zero vector with D dimension. After we replace all words in comment with their embedding values, we can get a series of $K \times D$ comment vectors. The final step is to combine N comment vectors into a $N \times K \times D$ vector.

B. Document Representation

We use a 2-layer Stacked Bi-LSTM model to obtain the document representation of each comment. Like Bi-directional LSTM (Bi-LSTM), Stacked Bi-LSTM can get rich contextual information from both past and future time sequences. However, different from Bi-LSTM, Stacked Bi-LSTM has more upper layers to conduct further feature extractions while Bi-LSTM only has a single hidden layer for each direction to extract features.

Figure 4 shows a peephole of a 2-layer Bi-LSTM. For time sequence T, the input sequence $\{x_1, x_2, ..., x_T\}$ enters into hidden layers in the forward direction $\{a_1, a_2, ..., a_T\}$ to obtain a complete information from all past time steps and hidden layers in the reverse direction $\{c_1, c_2, ..., c_T\}$ to get a complete information from all future time steps. After that, the upper hidden layers take the outputs from lower hidden layers at each time step as their inputs to extract further features. Specifically, the upper layers of forward hidden layers are $\{b_1, b_2, ..., b_T\}$ and the upper layers of backward hidden layers are $\{d_1, d_2, ..., d_T\}$. At last, output layers integrate two upper layers' hidden vector together as their output.

In Figure 4, each node of hidden layers represents an LSTM cell, which has a new memory, an input gate, a forget gate and an output gate, denoted as u_t, i_t, f_t and o_t, respectively (see the right subfigure in Figure 4). The new memory represents new candidate value after adding new input. The input gate represents the new information storing in the cell state. The forget gate means what information dumping from the cell state. The output gate decides which parts of the cell state to output. The control parameter C_t decides the information storing or dumping. We choose one of the variations of LSTM

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access

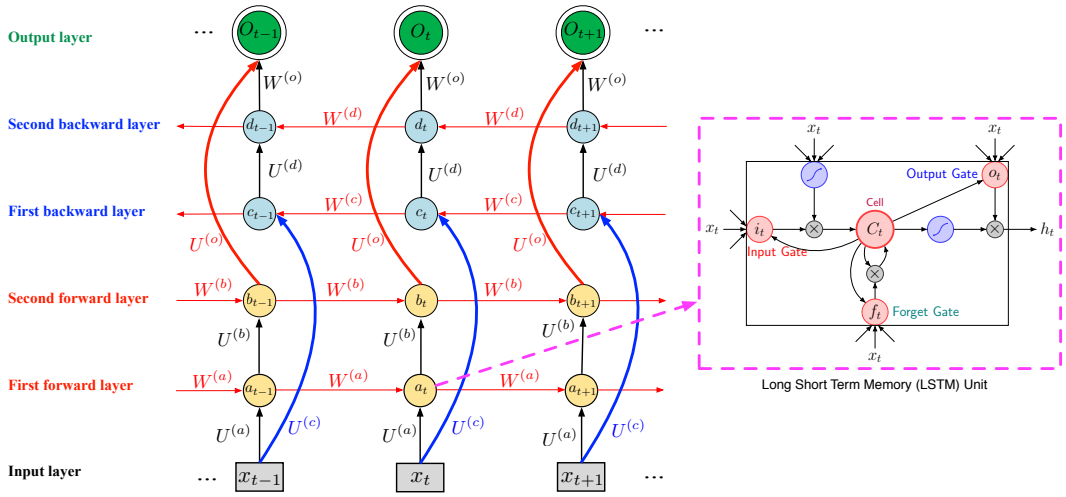


Fig. 4: Peephole of 2-layer Bi-LSTM

model to calculate hidden states a_t, b_t, c_t and d_t at each layer for each time step t.

For the first forward layer, hidden state a_t is given by the following equations:

$$i_t^{(a)} = \sigma(U_i^{(a)} x_t + W_i^{(a)} a_{t-1} + b_i^{(a)}), \quad (1)$$

$$f_t^{(a)} = \sigma(U_f^{(a)} x_t + W_f^{(a)} a_{t-1} + b_f^{(a)}), \quad (2)$$

$$o_t^{(a)} = \sigma(U_o^{(a)} x_t + W_o^{(a)} a_{t-1} + b_o^{(a)}), \quad (3)$$

$$u_t^{(a)} = \tanh(U_u^{(a)} x_t + W_u^{(a)} a_{t-1} + b_u^{(a)}), \quad (4)$$

$$C_t^{(a)} = i_t^{(a)} \odot u_t^{(a)} + f_t^{(a)} \odot C_{t-1}^{(a)}, \quad (5)$$

$$a_t = o_t^{(a)} \odot \tanh(C_t^{(a)}). \quad (6)$$

For the second forward layer, hidden state b_t is calculated by the following equations:

$$i_t^{(b)} = \sigma(U_i^{(b)} a_t + W_i^{(b)} b_{t-1} + b_i^{(b)}), \quad (7)$$

$$f_t^{(b)} = \sigma(U_f^{(b)} a_t + W_f^{(b)} b_{t-1} + b_f^{(b)}), \quad (8)$$

$$o_t^{(b)} = \sigma(U_o^{(b)} a_t + W_o^{(b)} b_{t-1} + b_o^{(b)}), \quad (9)$$

$$u_t^{(b)} = \tanh(U_u^{(b)} a_t + W_u^{(b)} b_{t-1} + b_u^{(b)}), \quad (10)$$

$$C_t^{(b)} = i_t^{(b)} \odot u_t^{(b)} + f_t^{(b)} \odot C_{t-1}^{(b)}, \quad (11)$$

$$b_t = o_t^{(b)} \odot \tanh(C_t^{(b)}). \quad (12)$$

For the first backward layer, hidden state c_t is given by:

$$i_t^{(c)} = \sigma(U_i^{(c)} x_t + W_i^{(c)} c_{t+1} + b_i^{(c)}), \quad (13)$$

$$f_t^{(c)} = \sigma(U_f^{(c)} x_t + W_f^{(c)} c_{t+1} + b_f^{(c)}), \quad (14)$$

$$o_t^{(c)} = \sigma(U_o^{(c)} x_t + W_o^{(c)} c_{t+1} + b_o^{(c)}), \quad (15)$$

$$u_t^{(c)} = \tanh(U_u^{(c)} x_t + W_u^{(c)} c_{t+1} + b_u^{(c)}), \quad (16)$$

$$C_t^{(c)} = i_t^{(c)} \odot u_t^{(c)} + f_t^{(c)} \odot C_{t-1}^{(c)}, \quad (17)$$

$$c_t = o_t^{(c)} \odot \tanh(C_t^{(c)}). \quad (18)$$

For the second backward layer, hidden state d_t is given by the following equations:

$$i_t^{(d)} = \sigma(U_i^{(d)} c_t + W_i^{(d)} d_{t+1} + b_i^{(d)}), \quad (19)$$

$$f_t^{(d)} = \sigma(U_f^{(d)} c_t + W_f^{(d)} d_{t+1} + b_f^{(d)}), \quad (20)$$

$$o_t^{(d)} = \sigma(U_o^{(d)} c_t + W_o^{(d)} d_{t+1} + b_o^{(d)}), \quad (21)$$

$$u_t^{(d)} = \tanh(U_u^{(d)} c_t + W_u^{(d)} d_{t+1} + b_u^{(d)}), \quad (22)$$

$$C_t^{(d)} = i_t^{(d)} \odot u_t^{(d)} + f_t^{(d)} \odot C_{t-1}^{(d)}, \quad (23)$$

$$d_t = o_t^{(d)} \odot \tanh(C_t^{(d)}). \quad (24)$$

For each time step t, the output O_t is generated by the combination of hidden vectors of the second forward layer b_t and the second backward layer d_t. In particular, we have

$$O_t = U^{(O)} b_t + W^{(O)} d_t + b^{(O)}. \quad (25)$$

In our case, the input is a 3-dimensional matrix with a size of $N \times K \times D$. Stacked Bi-LSTM represents the past and future context, and combines both parts' features together as outputs of the model.

C. Sentiment Prediction

Softmax classifier takes the output at the last step K and O_K serves as its input. As noted above, given N comments with K words, we predict the sentiment y for each comment. Real annotations of comments are represented by $Y (Y = Y_1, Y_2, ..., Y_N)$. The predicted values y' can be calculated by:

$$p(y|X) = \mathrm{softmax}(W^{(s)} O_K + b^{(s)}), \quad (26)$$

and

$$y' = \arg\max_y p(y|X). \quad (27)$$

We then use the cross entropy to train the loss function. We first derive the loss of each labeled comment and the final loss

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access

TABLE II: Summary of Testing Data Sets

	Testing Ratio = 25%			Testing Ratio = 30%			Testing Ratio = 40%		
Experiment No.	1	2	3	4	5	6	7	8	9
# of Positive comments	381	378	381	455	445	465	597	611	599
# of Negative comments	369	372	369	445	455	435	603	589	601

is averaged over all the labeled comments by the following equation:

$$\text{Loss} = -\frac{1}{N}\sum_{n=1}^{N} Y_n \cdot \log p\left(y_n \,|X_n\right) \tag{28}$$

where the subscript n indicates the n^{th} input comment.

We then use Adam optimizer [24] to adaptively adjust learning rate and optimize parameters of the model. At each hidden layer, we also introduce dropout [25] with 70% keeping ratio to avoid over-fitting.

V. Experimental Results

In this section, we conduct the experiments to evaluate the performance of the proposed model in Chinese sentiment analysis. In Section V-A, we describe basic experimental settings. Section V-B presents the performance comparison of our proposed approach with other existing methods. We then investigate the impacts of parameters in Section V-C.

A. Experimental Settings

1) Dataset description: As far as we know, there are a limited number of Chinese corpora especially for sentimental study. Therefore, we establish a Chinese corpus to dedicate for sentiment analysis. In particular, we develop a crawler based on Weibo API to collect comments of Chinese microblogs. In summary, we crawl 65,536 comments from 120 statuses as inputs of CBOW model. To train classifiers, we randomly select and annotated 3,000 comments out of the original comments. Among the data set, there are 1,514 comments labeled as positive and 1,486 comments labeled as negative.

2) Model Setting: We use 100-dimensional word embedding vectors as the inputs of all models (our model and other baseline models). Moreover, we set the maximum sentence length to 13 words [26]. The network weights are randomly initialized via using a truncated normal distribution (with mean $\mu = 0$ and variance $\sigma = 1.0$). We then develop single-layer LSTM, single-layer Bi-LSTM and 2-layer Bi-LSTM. All these models have the similar settings. Particularly, each layer of the above models has a hidden size of 64. We fix the batch size for every epoch as 300. Furthermore, we also evaluate the performance by choosing a parameter namely Random State (RS), where RS is the seed used to generate the random number in our experiments. Accordingly, the test sets were generated in various groups according to different RS values. After evaluating different groups with different RS values, we can get the best performance from the experiments.

3) Experimental Performance Evaluation: We randomly select training sets and testing sets so as to reduce the imbalanced impact. Table II lists the descriptive statistics of testing data sets. In particular, we conduct 9 groups of experiments. The first 3 groups of experiments (i.e., experiments No. 1 to No. 3) choose 25% as testing ratio. Experiments No. 4 to No. 6 choose 30%. Experiments No. 7 to No. 9 choose 40% as testing ratio.

We adopt the *prediction accuracy* to measure performance in this paper. In particular, the prediction accuracy of a model can be calculated by the following equation:

$$\text{Accuracy} = \frac{1}{N}\sum_{n=1}^{N} (\arg\max_{y} p(y_n|X_n) \bar{\vee} \arg\max_{y} Y_n), \tag{29}$$

where $\bar{\vee}$ is Exclusive-NOR (aka XNOR) operation. Recent studies such as [27], [28] on Natural Language Processing (NLP) adopted the similar metric. Since exclusive NOR gate is another exclusive gate in the logic gates, this equation can effectively count the frequency of positive and negative words in our experiment. Note that all the variables in Eq. (29) are defined in Section IV.

In order to ensure the comparison fairness, we choose the same number of training times for all machine learning models.

B. Comparison with baseline models

1) Baseline models: We then compare the proposed method with other baseline models in terms of the prediction accuracy. For this purpose, we implement the following baseline models:

- **SVM** [29] is a basic support vector classifier (SVC) with radial basis function (RBF) kernel. Regarding to representing documents, we use Bag-of-Words to get the frequency of the words as the feature of each comment.
- **CBOW + Logistic Regressiong (LR)** is a combination of CBOW and LR models. It uses word embedding vectors to represent documents by averaging the embedding vectors' value for each dimension. We implement LR model to classify the 100-dimensional comment vectors.
- **CBOW + SVM** is a combination of CBOW and SVM. We represent Chinese microblog comments in the same way as LR model. We implement a basic support vector classifier with RBF kernel and take the features of comments as inputs consequently conducting the sentiment classification.
- **CBOW + Convolutional Neural Network (CNN)** is a combination of CBOW and CNN. CNN models show the advantages in learning complicated data. We implement CNN model with word embedding model on sentiment classification.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access

TABLE III: Prediction Accuracy of Sentiment Analysis

Models	Prediction Accuracy (%)									
	Testing Ratio = 25%			Testing Ratio = 30%			Testing Ratio = 40%			Average Accuracy
	No.1	No. 2	No.3	No.4	No.5	No.6	No.7	No.8	No.9	No.1 $\sim$ No.9
SVM	50.9	50.4	50.8	50.5	49.4	48.4	49.8	51	49.9	49.6
CBOW + SVM	84	85.3	83.1	84.9	84.3	83	84.3	83.6	83	83.2
CBOW + LR	85.5	86.4	85.6	85.8	86	84.9	85.8	84.8	85.3	85.1
CBOW + CNN	85.7	86.8	85.7	86.4	86.3	85	86.1	85	85.9	85.4
CBOW + Stacked CNN	86	87.1	85.9	87.2	86.8	85.8	86	85.8	86.1	85.9
CBOW + LSTM	87.2	88.3	86.4	88.3	88.7	86.7	87.2	87.1	86.8	87
CBOW + Bi-LSTM	88	88.9	86.7	89.1	89	87.1	88.3	87.4	87.5	87.6
CBOW + Stacked Bi-LSTM (2 Layers)	**89.5**	**91.7**	**88.7**	**90.8**	**90.4**	**89.1**	**90**	**88.9**	**89.1**	**89**

- **CBOW + Stacked CNN** is a combination of CBOW and Stacked CNN. Stacked CNN models contain multiple layers with a large number of parameters. We implement Stacked CNN model with 2 layers to conduct the sentiment prediction.
- **CBOW + LSTM** is a combination of CBOW and LSTM. In particular, CBOW + LSTM takes the same input as our Stacked Bi-LSTM model. Moreover, it utilizes the last hidden state h_t for sentiment prediction.
- **CBOW + Bi-LSTM** is a combination of CBOW and Bi-LSTM. It can be regarded as a special case of our Stacked Bi-LSTM model with the removal of stacked layers. The difference between CBOW + LSTM and CBOW + Bi-LSTM lies in different RNN architectures. In LSTM, there is only one forward layer and no backward layer. In Bi-LSTM, there is one forward layer and one backward layer.

2) Experiment Results: Table III shows the experimental results of sentiment prediction in 9 groups of experiments. Note that, the size of network input is 390,000 ($13\times100\times300$). From Table III, we have the following findings:

(a) *Integration with CBOW model can achieve better performance than the models without CBOW.* For example, CBOW + SVM have much higher prediction accuracy than SVM without CBOW.
(b) *Deep learning models outperform conventional machine learning methods.* For example, CBOW + LSTM and CBOW + Bi-LSTM outperform conventional machine learning methods such as SVM and LR.
(c) *CBOW + LSTM outperforms CBOW + CNN and CBOW + stacked CNN in terms of prediction accuracy.* Due to the recurrent neural network, LSTM can better deal with text data than CNN model.
(d) *CBOW + Bi-LSTM outperforms CBOW + LSTM in terms of prediction accuracy.* This is because Bi-LSTM has an additional hidden layer in backward direction so that it can extract more features than LSTM model.
(e) *Our Stacked Bi-LSTM outperforms all other models in all the experiments.* It implies that stacking more layers is beneficial to the feature extraction.
(f) *The testing ratio has the little effect on the performance of all the models.* Prediction accuracy fluctuates a little with the increment of testing ratio as shown in 9 groups of experiments.

C. Impacts of Parameters

We next evaluate the impacts of parameters of our Stacked Bi-LSTM. In particular, we consider the following parameters: 1) the number of Stacked Bi-LSTM Layers, 2) the number of LSTM cells, 3) the maximum sentence length.

TABLE IV: Impact of Number of Stacked Bi-LSTM Layers

No. of Stacked Bi-LSTM Layers	Loss
2	0.09713
3	0.08906
4	0.08442

1) Impact of number of Stacked Bi-LSTM Layers: We first investigate the impact of the number of stacked layers. In particular, we first fix the number of the hidden cells to 64 and the maximum sentence length to 13, then vary the number of Bi-LSTM layers from 2 to 4. Table IV shows the prediction accuracy and loss values when the maximum prediction accuracy values are achieved (i.e., 88.0%, 88.7% and 89.2%) corresponding to the number of layers to 2, 3, 4, respectively. We observe from Table IV that the prediction accuracy increases and prediction loss decreases with the increased number of Stacked Bi-LSTM layers. This is because stacking more layers can help to extract more features consequently improving the performance.

2) Impact of Different word Embedding methods: Table V shows the experimental results of sentiment prediction in two types of Word2vec models: CBOW and Skip-gram, both of which are widely used in word embeddings. As shown in Table V, we can observe that Skip-gram outperforms CBOW in terms of prediction accuracy. In addition, our proposed Stacked Bi-LSTM model with either CBOW or Skip-gram has better predication accuracy than other existing models. This improvement may owe to the stacked architecture in our model.

TABLE V: Prediction Accuracy of Sentiment Analysis with Different Word Embedding Models

Testing Ratio = 25% (No.1)		
Models	**Prediction Accuracy (%)**	
	Skip-Gram	CBOW
SVM	87.6	84
LR	88.5	85.5
CNN	88.7	85.7
Stacked CNN	88.4	86
LSTM	88.9	87.6
Bi-LSTM	89.3	88.1
Stacked Bi-LSTM	**90.3**	**89.5**

3) Influence of Different Factors: The performance of our Stacked Bi-LSTM model is affected by different factors in varying degrees either internally or externally. Since the testing ratio has the little effect on performance, we conduct the following experiments based on experiment No. 1 (as shown in Table II). Note that both the loss and the prediction accuracy can be calculated by Eqns. (28) and (29) in Section IV-C and Section V-A3, respectively.

Regarding to the model itself, the changed structure may have the immediate impact on the performance. To investigate this effect, we fix the number of hidden layers as 2 and the maximum sentence length as 13. We then vary the number of hidden units and observe the performance variation.

Figure 5 shows the performance comparison for different number of hidden cells. It is shown in Figure 5 that the more LSTM cells brings the better performance. For example, the average loss with 128 LSTM cells (the blue curve) is much lower than that with 32 LSTM cells. Moreover, Figure 5 also shows that the model with 128 LSTM cells achieves faster convergence than the other two models. In addition, both training accuracy and validation accuracy are enhanced as the number of LSTM cells increases. This result confirms the observation that the larger number of the hidden cells, the richer information can be extracted consequently the model performing better.

We next investigate the impact of the maximum sentence length. In particular, we first fix the number of hidden cells to 64 and the number of hidden layers to 2. We then vary the maximum sentence length from 7 words to 13 words. Figure 6 shows the results.

Figure 6 shows us that the loss declines slightly as the maximum sentence length increases. At the same time, both the training accuracy and validation accuracy rise slightly with the increased maximum sentence length. This increment may owe to the effect that more context features can be extracted from the longer sentence.

VI. Conclusion

In this paper, we integrate Continuous Bag-of-Words (CBOW) model and stacked bidirectional LSTM (Stacked Bi-LSTM) model to conduct sentiment analysis on Chinese microblog. In this method, the CBOW model can represent semantic features of words while Stacked Bi-LSTM model can extract context features from sequential word embedding vectors. At last, feature vectors provided by Bi-LSTM are fed into a binary softmax to conduct the sentiment prediction. Experimental results demonstrate that our proposed method (CBOW + Stacked Bi-LSTM) achieves better performance over other machine learning and deep learning models, consequently verifying the effectiveness of applying CBOW and Stacked Bi-LSTM model.

In the future, we will further investigate user-attention mechanism and improve the learning capability of long-range dependencies. We may exploit other preprocessing methods (such as Glove) to improve the quality of input dataset. In addition, we also consider using the pre-trained word embedding on other large corpuses like COAE-2015. Meanwhile, we will evaluate the applicability of our model to other languages, especially for East Asia languages (such as Japanese, Vietnamese). In particular, we will first establish a corpus library with multiple languages and then conduct comprehensive evaluation on the proposed model.

References

[1] F. Wu, Y. Huang, Y. Song, and S. Liu, "Towards building a high-quality microblog-specific chinese sentiment lexicon," *Decision Support Systems*, vol. 87, pp. 39–49, 2016.

[2] S. Zhang, Z. Wei, Y. Wang, and T. Liao, "Sentiment analysis of chinese micro-blog text based on extended sentiment dictionary," *Future Generation Computer Systems*, vol. 81, pp. 395–403, 2018.

[3] W. Li, Y. Li, and Y. Wang, "Chinese microblog sentiment analysis based on sentiment features," in *Asia-Pacific Web Conference*. Springer, 2016, pp. 385–388.

[4] B. Chen, Z. Hao, R. Cai, W. Wen, and S. Du, "Sentiment target extraction based on crfs with multi-features for chinese microblog," in *Asia-Pacific Web Conference*. Springer, 2016, pp. 29–41.

[5] L. Zheng, H. Wang, and S. Gao, "Sentimental feature selection for sentiment analysis of chinese online reviews," *International journal of machine learning and cybernetics*, vol. 9, no. 1, pp. 75–84, 2018.

[6] P. Ficamos, Y. Liu, and W. Chen, "A naive bayes and maximum entropy approach to sentiment analysis: Capturing domain-specific data in weibo," in *Big Data and Smart Computing (BigComp), 2017 IEEE International Conference on*. IEEE, 2017, pp. 336–339.

[7] J. Liang, P. Liu, J. Tan, and S. Bai, "Sentiment classification based on as-lda model," *Procedia Computer Science*, vol. 31, pp. 511–516, 2014.

[8] Y. Li and B. Shen, "Research on sentiment analysis of microblogging based on lsa and tf-idf," in *Computer and Communications (ICCC), 2017 3rd IEEE International Conference on*. IEEE, 2017, pp. 2584–2588.

[9] B. Xue, C. Fu, and Z. Shaobin, "A study on sentiment computing and classification of sina weibo with word2vec," in *Big Data (BigData Congress), 2014 IEEE International Congress on*. IEEE, 2014, pp. 358–363.

[10] D. Zhang, H. Xu, Z. Su, and Y. Xu, "Chinese comments sentiment classification based on word2vec and svmperf," *Expert Systems with Applications*, vol. 42, no. 4, pp. 1857–1863, 2015.

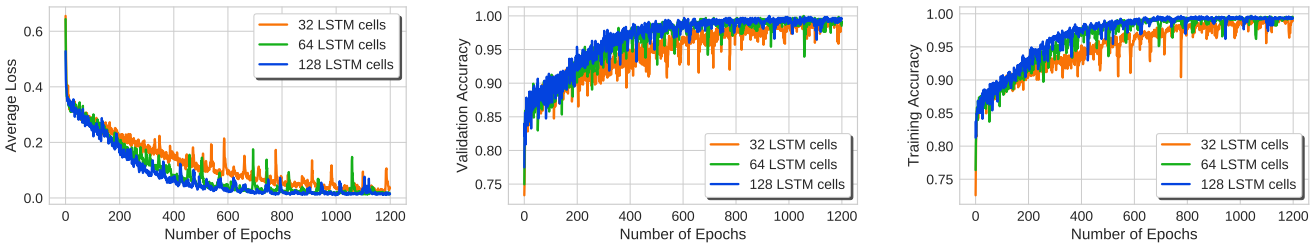
[11] L. Xing, L. Yuan, W. Qinglin, and L. Yu, "An approach to sentiment analysis of short chinese texts based on svms," in *Control Conference (CCC), 2015 34th Chinese*. IEEE, 2015, pp. 9115–9120.

[12] Z. Hao, R. Cai, Y. Yang, W. Wen, and L. Liang, "A dynamic conditional random field based framework for sentence-level sentiment analysis of chinese microblog," in *Computational Science and Engineering (CSE) and Embedded and Ubiquitous Computing (EUC), 2017 IEEE International Conference on*, vol. 1. IEEE, 2017, pp. 135–142.

[13] Y. Cao, Z. Chen, R. Xu, T. Chen, and L. Gui, "A joint model for chinese microblog sentiment analysis," in *Proceedings of the Eighth SIGHAN Workshop on Chinese Language Processing*, 2015, pp. 61–67.

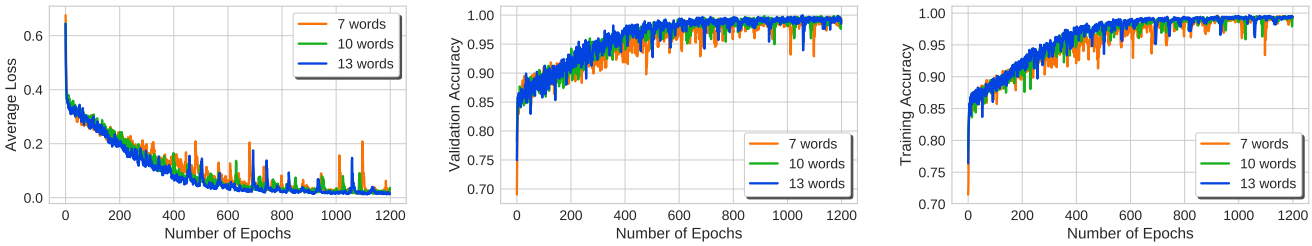
[14] Y. Wang, S. Feng, D. Wang, G. Yu, and Y. Zhang, "Multi-label chinese microblog emotion classification via convolutional neural network," in *Asia-Pacific Web Conference*. Springer, 2016, pp. 567–580.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2905048, IEEE Access



(a) Average Loss (b) Validation Accuracy (c) Training Accuracy

Fig. 5: Performance of Stacked Bi-LSTM Model with different number of hidden cells



(a) Average Loss (b) Validation Accuracy (c) Training Accuracy

Fig. 6: Performance of Stacked Bi-LSTM Model with different sentence lengths

[15] Y.-C. Chang, C.-S. Chou, Y. Zhang, X. Wang, and W.-L. Hsu, "Sentiment analysis of chinese microblog message using neural network-based vector representation for measuring regional prejudice." in *PACIS*, 2016, p. 307.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[17] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[18] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013, pp. 6645–6649.

[19] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 273–278.

[20] Y. Zhang, Y. Jiang, and Y. Tong, "Study of sentiment classification for chinese microblog based on recurrent neural network," *Chinese Journal of Electronics*, vol. 25, no. 4, pp. 601–607, 2016.

[21] Y. Wang, S. Feng, D. Wang, Y. Zhang, and G. Yu, "Context-aware chinese microblog sentiment classification with bidirectional lstm," in *Asia-Pacific Web Conference*. Springer, 2016, pp. 594–606.

[22] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[23] A. M. Dai, C. Olah, and Q. V. Le, "Document embedding with paragraph vectors," *arXiv preprint arXiv:1507.07998*, 2015.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

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

[26] X. Sun, C. Li, and F. Ren, "Sentiment analysis for chinese microblog based on deep neural networks with convolutional extension features," *Neurocomputing*, vol. 210, pp. 227–236, 2016.

[27] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations." *Journal of Machine Learning Research*, vol. 18, no. 187, pp. 1–30, 2017.

[28] C. Xu, J. Yao, Z. Lin, W. Ou, Y. Cao, Z. Wang, and H. Zha, "Alternating multi-bit quantization for recurrent neural networks," *arXiv preprint arXiv:1802.00150*, 2018.

[29] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.