Hu F, Li L, Zhang ZL *et al.* Emphasizing essential words for sentiment classification based on recurrent neural networks. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 32(4): 785–795 July 2017. DOI 10.1007/s11390-017-1759-2

Emphasizing Essential Words for Sentiment Classification Based on Recurrent Neural Networks

Fei Hu [1,2], *Student Member, CCF*, Li Li [1,*], *Senior Member, CCF, Member, ACM*
Zi-Li Zhang [1], *Distinguished Member, CCF, Member, ACM*, Jing-Yuan Wang [1], *Student Member, CCF*
and Xiao-Fei Xu [1], *Student Member, CCF*

[1] *College of Computer and Information Science, Southwest University, Chongqing 400715, China*
[2] *Network Centre, Chongqing University of Education, Chongqing 400065, China*

E-mail: jackyetz@email.swu.edu.cn; {lily, zhangzl}@swu.edu.cn; {wjykim, nakamura}@email.swu.edu.cn

Received December 20, 2016; revised June 3, 2017.

Abstract With the explosion of online communication and publication, texts become obtainable via forums, chat messages, blogs, book reviews and movie reviews. Usually, these texts are much short and noisy without sufficient statistical signals and enough information for a good semantic analysis. Traditional natural language processing methods such as Bow-of-Word (BOW) based probabilistic latent semantic models fail to achieve high performance due to the short text environment. Recent researches have focused on the correlations between words, i.e., term dependencies, which could be helpful for mining latent semantics hidden in short texts and help people to understand them. Long short-term memory (LSTM) network can capture term dependencies and is able to remember the information for long periods of time. LSTM has been widely used and has obtained promising results in variants of problems of understanding latent semantics of texts. At the same time, by analyzing the texts, we find that a number of keywords contribute greatly to the semantics of the texts. In this paper, we establish a keyword vocabulary and propose an LSTM-based model that is sensitive to the words in the vocabulary; hence, the keywords leverage the semantics of the full document. The proposed model is evaluated in a short-text sentiment analysis task on two datasets: IMDB and SemEval-2016, respectively. Experimental results demonstrate that our model outperforms the baseline LSTM by 1%∼2% in terms of accuracy and is effective with significant performance enhancement over several non-recurrent neural network latent semantic models (especially in dealing with short texts). We also incorporate the idea into a variant of LSTM named the gated recurrent unit (GRU) model and achieve good performance, which proves that our method is general enough to improve different deep learning models.

Keywords short text understanding, long short-term memory (LSTM), gated recurrent unit (GRU), sentiment classification, deep learning

1 Introduction

Short texts are very prevalent on today's websites, such as forums, tweets, microblogs, commodity reviews and short messages. They have played an increasingly important role in our daily lives. Compared with long texts, dealing with short texts is a big challenge according to the following characteristics.

1) Short texts do not always observe the syntax. Methods that work on long and well-organized texts fail on short texts, such as Part-of-Speech (POS) tagging and dependency parsing methods[1-4].

2) Short texts are less than 140 words, having limited, even ambiguous context. Thus, short texts usually do not contain sufficient statistical signals. Bow-of-Word (BOW) based probabilistic latent semantic mod-

Regular Paper
Special Issue on Deep Learning
The work was supported by the Scientific and Technological Research Program of Chongqing Municipal Education Commission of China under Grant No. KJ1501405, the National Natural Science Foundation of China under Grant No. 61170192, and the Chongqing Science and Technology Commission (CSTC) under Grant No. cstc2015gjhz40002.
*Corresponding Author

els do not work better in tasks with short texts than with long texts[5-8].

These characteristics bring a significant amount of ambiguity for understanding the semantics of short texts. Many approaches were introduced to relax them[2,9-18]. In [2], models are improved from three aspects: text segmentation, POS tagging, and concept labeling. In the aspect of text segmentation, a short text is divided into a sequence of meaningful components, and this method is improved from the Longest-Cover method[19-20]; in the aspect of POS tagging, not only lexical features but also the semantics is considered as tag POS; in the aspect of labeling, type detection is incorporated into the framework of short text understanding, and instance disambiguation is conducted based on all types of contextual information[19-20]. In [12], an n-gram model is suggested. In [14, 16-17], topic models are used to analyze the latent topics of short texts that help to extract potential semantic structures. In [15, 18], features are extended and carefully selected.

All these improved methods mentioned above are still BOW-based methods. They do not consider the term dependencies, or just consider adjacent term dependencies like n-gram models[21-22]. Term dependencies contribute significantly to the semantics of short texts. For example in Table 1, text 1 and text 2 have the same vocabularies, yet the wording of the sentences is shuffled. As a result, the meanings of them are totally different. Term dependencies hide in sequences, but BOW-based models fail to capture the hidden dynamics in sequences. On the contrary, the long short-term memory (LSTM) network, an extension of recurrent neural network (RNN), which was proposed by Hochreiter and Schmidhuber[23], is promising. It is explicitly designed to capture the dependencies between words, even in a long distance. Unlike n-gram models that suffer from long-term memory property, the nature of LSTM is observed to remember the information for long periods of time, which is consistent with long-term memory requirement. The LSTM model takes every single word as the input for processing of temporal patterns rather than BOW tokens or sentences. Imitating human memory mechanism, it memorizes words one by one.

Table 1. Lower-Cased Sample Texts

	Text
1	tom help a dog win the game.
2	the game help tom win a dog.

By analyzing texts, we find that a number of essential words (we call essential words as keywords in the rest of the paper) contribute greatly to the semantics of them. It is similar to the process of human memorization. For example, when reading a text, people are always impressed by a few words or phrases. For example, from the sentence "Such a wonderful day. I like it.", we can see that the words "wonderful" and "like" contribute more to understanding the semantics than the rest of the sentence.

We developed a latent semantic model based on LSTM, called Memory-Enhanced Latent Semantic model (MLSM@DASFAA) in [24]. MLSM@DASFAA pays more attention to the keywords, and leaves more room in the memory block for them. As a result, keywords provide a leverage to the semantics of the full document, and short texts can be better understood. In this paper, we improve MLSM@DASFAA by introducing the exponential decay of keyword impression, and incorporate this idea into a variant of LSTM named gated recurrent unit (GRU) which achieves better effect.

The contribution of this paper is non-trivial in that we put forward a solution of exploiting topics of multi-granularity and present a systematic way to seamlessly integrate the topics and produce discriminative features for short text classification. This is the first time of leveraging multi-granularity topics for classification as far as we know.

The rest of the paper is organized as follows. We first discuss related work in Section 2. Then we propose our models (MLSM based on LSTM and MLSM based on GRU, where the former one is improved from MLSM@DASFAA and the latter one is developed based on GRU) in Section 3. In Section 4, we describe the two datasets, present a keyword selection method, construct a deep neural network with our model, and obtain the results. In Section 5, we discuss the importance of term dependencies in memorizing and understanding short texts, and discuss the role of keywords in distinguishing memory. Finally, we draw the conclusion.

2 Related Work

2.1 Latent Semantic Models

Latent semantic models (LSMs) can measure the similarity between documents, no matter whether they have the same words or not. They address the language discrepancy between documents by grouping different terms which frequently occur in similar contexts into

the same semantic cluster[21]. For example, the latent semantic analysis (LSA) projects high-dimensional vector space representation of a document into a lower dimensional vector space[25]. The low-dimensional representation is full of semantics and can be understood by the system. The singular value decomposition (SVD) is introduced to realize this process by retaining a certain number of the largest characteristic values and eliminating the noises of the middle matrix. The refactored middle matrix describes the semantics more exactly[26]. Extended from LSA, probabilistic topic models such as the Probabilistic LSA (PLSA)[27], the Latent Dirichlet Allocation (LDA)[28], and the Bi-lingual Topic model (BLTM)[29] have been proposed and successfully been used to mine the semantics of documents. In recent years, neural networks, especially deep learning networks[30], have been introduced to promote the development of LSMs[21,31-32]. Salakhutdinov and Hinton demonstrated that hierarchical semantic representations of a document can be extracted via a deep learning structure[33]. RNN-based models are a kind of deep learning structures. They utilized the contextual information of a document to help understand the document. Our work is based on an extension of RNN: LSTM.

2.2 Long-Term Dependencies

RNN has a hidden layer which is capable of memorizing recent words. The hidden layer connects previous information to the present task that is like a memory block flowing through time. The memory mechanism helps capture term dependencies, further to understand a document. It has been successfully applied in many natural language processing (NLP) tasks, such as predicting next word given contextual information[34-35]. Compared with feed-forward neural networks, RNNs are characterized by the ability of encoding past information, and hence are suitable for sequential models. The Back-Propagation Through Time algorithm (BPTT) and the Real-Time Recurrent Learning algorithm (RTRL)[36] extend the ordinary BP algorithm to suit the recurrent neural architecture; however, RNNs have the problem of memorizing long past information. With BPTT or RTRL, error signals tend to vanish after steps of flowing, incapable of changing weights. As a result, long-term dependencies are hard to learn[37] (see detailed analysis in (1)∼(3)[38]). The error signal for the j-th unit at time step $t-1$ can be formulated as follows:

$$\vartheta_j(t-1) = f'_j(net_j(t-1))\Sigma_i w_{ij}\vartheta_i(t), \tag{1}$$

where ϑ is the error signal, $f'(\cdot)$ is the partial derivative of the activation function, $net(\cdot)$ is the input, and w_{ij} is a weight parameter for two neurons from the $t-1$ and the t time step. The target error has been removed from (1), because it is zero when the unit belongs to the non-output layer.

Derived from the above equation, we get the error signal for the j-th neuron at the time step of $t-q$ ($q \geqslant 1$). The error is scaled by:

$$\frac{\partial\vartheta_j(t-q)}{\partial\vartheta_i(t)} = \begin{cases} f'_j(net_j(t-1))w_{ij}, & \text{if } q=1, \\ f'_j(net_j(t-q))\Sigma_{l=1}^{n}\dfrac{\partial\vartheta_l(t-q+l)}{\partial\vartheta_i(t)}w_{lj}, & \text{if } q>1. \end{cases} \tag{2}$$

The above equation is expanded as follows:

$$\frac{\partial\vartheta_j(t-q)}{\partial\vartheta_i(t)} = \Sigma_{l_1}\cdots\Sigma_{l_{q-1}}\Pi_{m=1}^{q}f'_{l_m}(net_{l_m}(t-m))\times w_{l_m l_{m-1}},$$

$$T = \Pi_{m=1}^{q}f'_{l_m}(net_{l_m}(t-m))w_{l_m l_{m-1}}. \tag{3}$$

If $|T|>1$, the error will increase exponentially with the increase of q; if $|T|<1$, the error will vanish with the increase of q. In practice, we will always meet the error vanishing problem.

LSTM has fixed this problem by introducing a memory cell, named constant error carrousel (CEC). It is a self-connected device that keeps constant error signals, even from a far distance.

2.3 Long Short-Term Memory Network

The LSTM network[23], as an extension of RNN, plays an important role in text understanding. It is born with the ability to retain information over a much longer period of time than 10∼12 time steps, which is the limit of RTRL and BPTT models. Fig.1 depicts the LSTM structure[39], and the symbols are illustrated in Table 2, where subscripts t and $t-1$ are the current time step and the previous time step, respectively. $\boldsymbol{C}$ is like a conveyor belt. Along the belt, information flows. Through $\boldsymbol{f}$ and $\boldsymbol{i}$, old information is attenuated and new information is inserted. Through $\boldsymbol{o}$, the model produces the output for current time step. $\boldsymbol{C}$ is the key of the

LSTM structure. It keeps the information for long periods of time. Thus this structure supports long-term dependencies. The following equations mathematically abstract this process[41].

$$\boldsymbol{f}_t = \delta(\boldsymbol{W}_{\mathrm{f}} \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_{\mathrm{f}}), \tag{4}$$

$$\boldsymbol{i}_t = \delta(\boldsymbol{W}_{\mathrm{i}} \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_{\mathrm{i}}), \tag{5}$$

$$\boldsymbol{o}_t = \delta(\boldsymbol{W}_{\mathrm{o}} \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_{\mathrm{o}}),$$

$$\boldsymbol{C}'_t = \tanh(\boldsymbol{W}_C \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_C),$$

$$\boldsymbol{C}_t = \boldsymbol{f}_t \times \boldsymbol{C}_{t-1} + \boldsymbol{i}_t \times \boldsymbol{C}'_t,$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \times \tanh(\boldsymbol{C}_t),$$

where vectors inside the square brackets are concatenated.

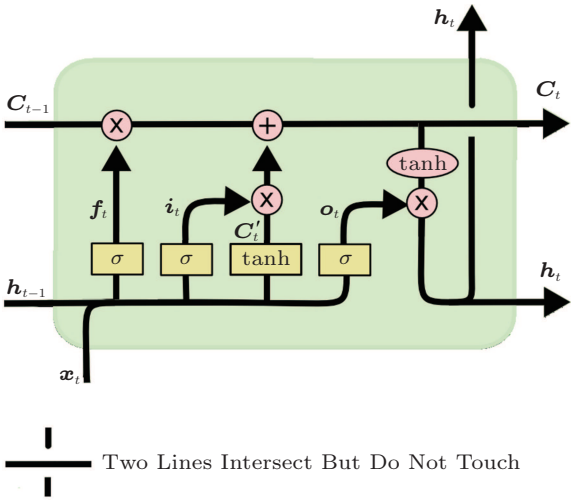


Fig.1. LSTM network.

Table 2. Symbols for LSTM

Symbol	Description
$\boldsymbol{W}_{\mathrm{f}}$, $\boldsymbol{W}_{\mathrm{i}}$, $\boldsymbol{W}_{\mathrm{o}}$, $\boldsymbol{W}_z$	Weight vectors for the forget, input, output and update gates respectively
$\boldsymbol{W}_C$	Weight vector for the input
$\boldsymbol{b}_{\mathrm{f}}$, $\boldsymbol{b}_{\mathrm{i}}$, $\boldsymbol{b}_{\mathrm{o}}$, $\boldsymbol{b}_C$ $\boldsymbol{b}_z$	Offset values
$\boldsymbol{h}$	Output
δ	Sigmoid function
tanh	Hyperbolic tangent function
$\boldsymbol{x}$	Input
$\boldsymbol{f}$, $\boldsymbol{i}$, $\boldsymbol{o}$, $\boldsymbol{z}$, $\boldsymbol{r}$	Forget, input, output, update and reset gates respectively
$\boldsymbol{C}$	Memory cell (CEC)
$\boldsymbol{C}'$	New candidate value
$\times$, $\otimes$	Element-wise product of two vectors
$*$	Product of a vector and a scalar, or product of two scalars
$\oplus$	Element-wise addition of two vectors
$1-$	Each element of a vector is subtracted from 1

There are many variants of LSTM which promote the development of learning long-term dependencies. In [40], a sigmoid layer called the "forget gate layer" is proposed to eliminate the part of old information from CEC, and free up more memory space for newly-incoming information. In [41], gate layers monitor the cell state through "peephole connections". In [42], Greff *et al.* coupled the forget and input gates into one. The coupling leads to reduced computational complexity and slightly higher accuracy. They also observed similar effects by removing the peephole connections. In [43], a simplification of the LSTM architecture is proposed, named GRU. It removes all peephole connections and the output gate, couples the forget and input gates into an update gate, and introduces a reset gate to control the information flowing through the recurrent connections.

In this paper, we develop a novel variant of LSTM to enhance the key information of the document with long-term dependencies. For the convenience of description, the model improved from LSTM will be called the LSTM-based model, and the model improved from GRU will be called the GRU-based model.

3 Memory-Enhanced Latent Semantic Model

We improve MLSM@DASFAA by introducing exponential decay of keyword impression, and develop a new model based on GRU. Experimental results show that our idea works well not only on the baseline LSTM but also on variants of LSTM.

3.1 Enhancing Memory of Keywords

Fig.2 depicts MLSM based on LSTM. Keywords are introduced to affect the forget and input gates. The red line illustrates this affection. *Mat* is matching function. $\boldsymbol{E}$ is key-memory-enhancing function. $\boldsymbol{E}'$ is context-weakening function. They are formulated as follows:

$$Mat(w_t, K) = \begin{cases} 0, & \text{if } w_{\mathrm{t}} \notin K, \\ k_j^v | w_t = k_j, & \text{otherwise}, \end{cases} \tag{6}$$

where w_t is current input word, K is a dictionary whose keys make a list of keywords and whose values are levels of importance. $K = \{k_1 : k_1^v, k_2 : k_2^v, \ldots, k_j : k_j^v, \ldots\}$. In our task of sentiment analysis, a keyword is a sentiment word, and the value represents the corresponding word's sentiment polarity. k_j is the j-th keyword, k_j^v is the value of sentiment polarity of k_j. k_j^v is in the interval between 0.0 and 1.0. The larger the value, the

more the word contributing to the semantics. If a current word does not belong to K, the function returns zero. The construction of the keyword dictionary will be discussed in Subsection 3.2.

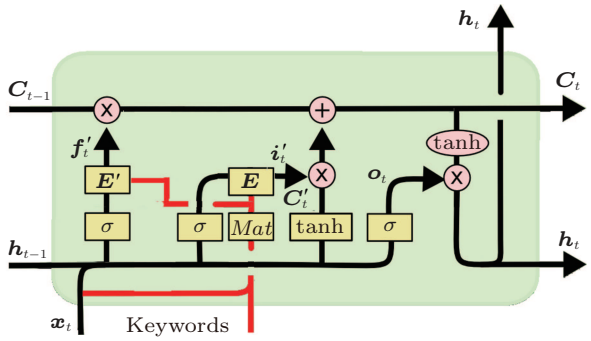


Fig.2. MLSM structure based on LSTM.

By further analyzing user memorization, we find that users not only are stuck in the mind by keywords but also have some impression on following adjacent words. We therefore assume that the impression conforms to the exponential decay. Fig.3 illustrates this assumption and (6) is modified as follows:

$$Mat(w_t, K) = \begin{cases} Mat(w_{t-1}, K) * e^{-Mat(w_{t-1}, K)}, & \text{if } w_t \notin K, \\ k_j^v | w_t = k_j, & \text{otherwise}, \end{cases} \quad (7)$$

where w_t is current input word and w_{t-1} is previous input word, Mat conforms to the exponential decay until it meets another keyword.

$$\boldsymbol{E}(\boldsymbol{i}, Mat) = (1 + Mat) * \boldsymbol{i},$$
$$\boldsymbol{E}'(\boldsymbol{i}, Mat) = (1 - \frac{Mat}{2}) * \boldsymbol{i},$$

where $\boldsymbol{i}$ is the input gate of baseline LSTM ((5)).

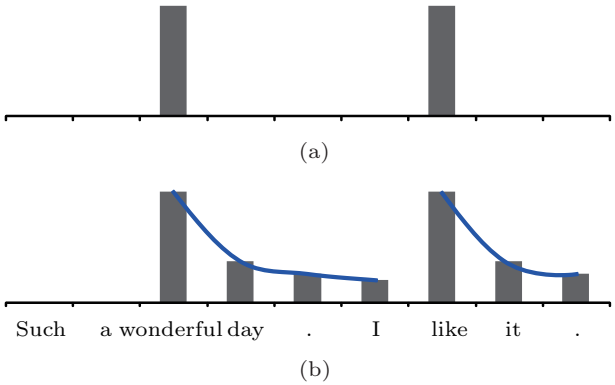


Fig.3. (a) User is only impressed by the keywords ((6)). (b) User's impression conforms to the exponential decay ((7)).

Then we get a revised input gate and a forget gate:

$$\boldsymbol{i}_t = (1 + Mat) * \delta(\boldsymbol{W}_{\mathrm{i}} \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_{\mathrm{i}}), \quad (8)$$
$$\boldsymbol{f}_t = (1 - \frac{Mat}{2}) * \delta(\boldsymbol{W}_{\mathrm{f}} \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_{\mathrm{f}}). \quad (9)$$

This revised input gate carefully controls the flows of newly-incoming information. The more important the newly-incoming word, the more the information getting through. The revised forget gate weakens the reserved old information. The more important the newly-incoming word, the less the old information retained, i.e., the more memory blocks emptied for newly-incoming information. Thus, keyword information flows are enhanced and normal word information flows are weakened. This model enhances the memory of keywords.

GRU is an extension of LSTM. It combines the forget and the input gates into a single update gate, merges the cell state and the hidden state, and introduces a reset gate $\boldsymbol{r}_t$ to control the flow of old information. If the newly-incoming word belongs to the list of keywords, less old information would be retained and more newly-incoming information would get through. Thus, we divide the update gate into two gates: $\boldsymbol{z}_t$ controls the flow of newly-incoming information and $\boldsymbol{z}'_t$ controls the flow of old information. The two gates are mathematically formulated as follows:

$$\boldsymbol{z}_t = Mat + (1 - Mat) * \delta(\boldsymbol{W}_z \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_z), \quad (10)$$
$$\boldsymbol{z}'_t = (1 - \frac{Mat}{2}) * (1 - \delta(\boldsymbol{W}_z \times [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_z)). \quad (11)$$

The structure of our model based on GRU is shown in Fig.4.

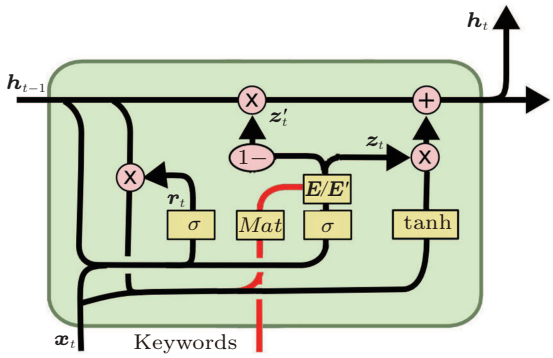


Fig.4. MLSM structure based on GRU.

The pseudo code of LSTM-based MLSM is shown in Fig.5. Here, *num_steps* is the loop count of the network, *inputsentence* is a sentence of the text whose words will

be fed into the model one by one, *inputtopics* is the keyword set, *topicpolarity* is the polarity value of the corresponding keyword which is calculated using (7), the input is generated using *inputsentence* and previous-step output, $\boldsymbol{f}$ and $\boldsymbol{i}$ (the subscripts are omitted for simplicity) are forget and input gates which come from (4) and (5), respectively, $\boldsymbol{f}'$ and $\boldsymbol{i}'$ (the subscripts are omitted for simplicity) come from (8) and (9), respectively, and *state* is the state of the hidden layer which has memorized the whole sentence.

```
Initialize inputsentence, inputtopics
state = self.initial_state
for time_step in range (num_steps):
    if inputsentence[time_step] in inputtopics:
        state = state × f' + input × i'
    else:
        state = state × f + input × i
return state
```

Fig.5. Pseudo code of LSTM-based MLSM.

The pseudo code of GRU-based MLSM is shown in Fig.6. Here, $\boldsymbol{zz}$ is the update gate of baseline GRU, $\boldsymbol{z}$ and $\boldsymbol{z}'$ (the subscripts are omitted for simplicity) come from (10) and (11), respectively, and other symbols are the same as above.

```
Initialize inputsentence, inputtopics
state = self.initial_state
for time_step in range (num_steps):
    if inputsentence[time_step] in inputtopics:
        state = state × z' + input × z'
    else:
        state = state × (1−zz) + input × zz
return state
```

Fig.6. Pseudo code of GRU-based MLSM.

3.2 Keyword Dictionary

Keywords make the most useful part for understanding a document. Especially with a short text which does not contain sufficient statistical signals, keywords emphasize the intent of a user who has presented the text. Thus keywords can be utilized to better interpret what the user wants to express. The sentiment analysis task is asked to pick keywords that would be good indicators of sentiment polarity. Many pairs of word and polarity make a sentiment dictionary, which is also called keyword dictionary. Table 3 lists several sentiment words and the corresponding polarity values, where polarity values have been normalized in the interval between 0.0 and 1.0. This table only considers the polar intensity, ignoring positive and negative factors.

Table 3. Keyword Dictionary for Sentiment Analysis

Keywords	Polarity
Ornery, crotchety, cantankerous	0.875
Good, do_good, goodness, not_bad, benefit	0.875
Hot	0.750
Sorry, sad, pitiful, grim, gloomy, dreary	0.625

In this paper, a keyword dictionary is built from SentiWordNet 3.0[44-45], and it is used to help enhancing the memory of the essential words in the text. SentiWordNet is a lexical resource for opinion mining. It assigns to each synset of WordNet[46] three sentiment scores: positivity, negativity and objectivity, where the objective score (i.e., ObjScore) is obtained by the equation "ObjScore = 1 − (PosScore + NegScore)", PosScore is the positive score, and NegScore is the negative score. If the positive score is greater than the absolute value of the negative score, we select the positive score as the sentimental polarity of the word. On the contrary, we select the absolute value of the negative score as the sentimental polarity of the word. In other words, the sentimental polarity of a particular word would be either positive or negative determined by which score is bigger. Table 3 lists several words with their polar values.

4 Experiments and Results

4.1 Experimental Settings

We evaluate the proposed models on two datasets. The detail is as follows.

1) IMDB[47] is a large movie review dataset including a set of 25 000 highly polar movie reviews for training and 25 000 for testing, totally 50 000 reviews, and the polarity labels are either positive or negative. All reviews are grouped into four subsets according to the length of texts:

- short texts with the length of less than (including) 140;
- medium length texts between 141 and 200;
- ordinary length texts between 201 and 400;
- long texts greater than 400.

In addition, a group of total reviews is considered.

2) SemEval-2016 Task 4[48] consists of 6 000 tweets, and three polarity labels are considered: positive, negative and neutral. All tweets are short because of the length less than 140 words.

In order to evaluate our model with a sentiment analysis task, we introduce our model into a sentiment analysis framework (Fig.7). The input sequence $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}\}$ represents a review. And each $\boldsymbol{x}_i$, $i = 0, 1, \ldots, n-1$, is a word encoded using one-hot representation①. The embedding layer projects $\boldsymbol{x}_i$ from a high dimensional vector space into a lower dimensional space which has 128 dimensions in our study. The CECs in the MLSM layer produce a sequence $\{\boldsymbol{h}_0, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_{n-1}\}$. Then $\boldsymbol{h}$ is obtained by drawing maximum elements of this sequence at each bit over all time steps in the max pooling layer. At last, $\boldsymbol{h}$ is fed to a logistic regression layer whose output is a binary classification label which represents positivity with 1 or negativity with 0.

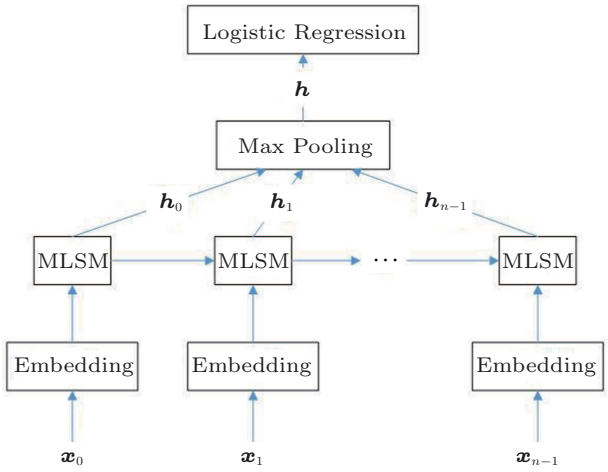


Fig.7. Framework used in the sentiment analysis task, into which MLSM is introduced.

We have five RNN-based frameworks: MLSM_LSTM, Baseline_LSTM, MLSM_GRU, Baseline_GRU, and MLSM@DASFAA[24], where MLSM_LSTM and MLSM_GRU are sentiment analysis frameworks into which our proposed LSTM-based and GRU-based models (MLSM layer in Fig.7) are incorporated, respectively, and basic LSTM and basic GRU models are incorporated into Baseline_LSTM and Baseline_GRU, respectively. For the convenience of description, we call the RNN-based frameworks as RNN-based models in the rest of the paper, i.e., the RNN-based model refers to any of the five frameworks, the MLSM-based model refers to MLSM_LSTM or MLSM_GRU, and the baseline model refers to Baseline_LSTM or Baseline_GRU. RNN-based models are trained using the Back Propagation algorithm (BP) and the mini-batch learning[49].

Comparative study is performed within the five RNN-based models and other non-RNN LSMs. Each RNN-based model has 128 neurons in the hidden layer and uses the optimization algorithm Adadelta[50]. The non-RNN LSMs are three LDA-based models and one word2vec-based model. The three LDA-based models are different in the vector representation of topic words: words are projected into a 30-dimensional vector space in lda1, 100-dimensional vector space in lda2, and 200-dimensional vector space in lda3. The word2vec-based model uses a 100-dimensional vector space to represent words. The four non-RNN models utilize support vector machine (SVM) to classify documents. IMDB and SemEval-2016 datasets are separately used with these models. The comparison of their accuracies is shown in Table 4. Table 5 lists the comparison of the four RNN-based models in the lost metric. The loss metric reflects the error between the real output and the expected output. The fewer the errors, the more robust the model. In both tables, the best score for each dataset/subset is marked in bold.

Table 4. Accuracy Study of LSMs with IMDB and SemEval-2016

Model	Dataset					
	IMDB					SemEval-2016
	$\leqslant 140$	$\leqslant 200$	$\leqslant 400$	> 400	All	
lda1-svm	0.785 7	0.804 9	0.816 9	0.816 8	0.800 8	-
lda2-svm	0.778 3	0.809 8	0.825 2	0.819 1	0.808 8	-
lda3-svm	0.773 9	0.807 4	0.816 1	0.808 4	0.808 9	-
word2vec-svm	0.802 0	0.826 0	0.847 0	0.822 0	0.828 0	-
Baseline_LSTM	0.865 3	0.864 7	0.864 3	0.864 8	0.864 9	0.708 5
MLSM_LSTM	0.877 4	0.878 8	**0.878 8**	0.877 9	**0.879 3**	0.724 1
Baseline_GRU	0.864 5	0.864 8	0.864 5	0.865 0	0.864 8	0.708 0
MLSM_GRU	**0.878 8**	**0.879 3**	0.878 6	**0.878 5**	0.878 5	**0.724 7**
MLSM@DASFAA	0.878 0	0.878 8	**0.878 8**	0.878 0	0.878 9	0.724 1

①One-hot representation is a word-encoding method that encodes every word into a long sparse vector. Each bit of this vector represents a unique word, and one bit will be set to 1 if a certain word is encoded but the rest bits will be set to 0. The length of the vector is the size of the vocabulary which usually includes all unique words in a dataset.

Table 5. Loss of LSTM-Based Models with IMDB and SemEval-2016

Model	Dataset					
	IMDB					SemEval-2016
	$\leqslant 140$	$\leqslant 200$	$\leqslant 400$	> 400	All	
Baseline_LSTM	0.457 2	0.456 3	0.456 7	0.456 6	0.456 6	0.591 1
MLSM_LSTM	0.451 1	0.451 4	0.451 2	0.451 1	0.451 0	0.580 8
Baseline_GRU	0.457 1	0.456 7	0.456 4	0.456 4	0.456 6	0.590 9
MLSM_GRU	**0.450 9**	**0.451 3**	**0.450 8**	**0.451 0**	**0.450 9**	**0.580 4**

Furthermore, several optimization algorithms are used in the four RNN-based models (MLSM_LSTM, Baseline_LSTM, MLSM_GRU and Baseline_GRU), respectively. And for simplicity, only datasets with short texts and the evaluation metric of accuracy are considered, i.e., texts including no more than 140 words and comparative study in term of accuracy. Comparison results are shown in Fig.8 (on IMDB, no more than 140 words) and Fig.9 (on SemEval-2016).

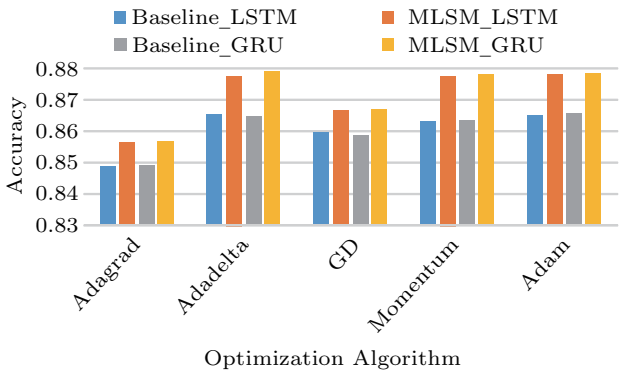


Fig.8. Accuracy study of RNN-based models with different optimization algorithms on texts including no more than 140 words of IMDB.

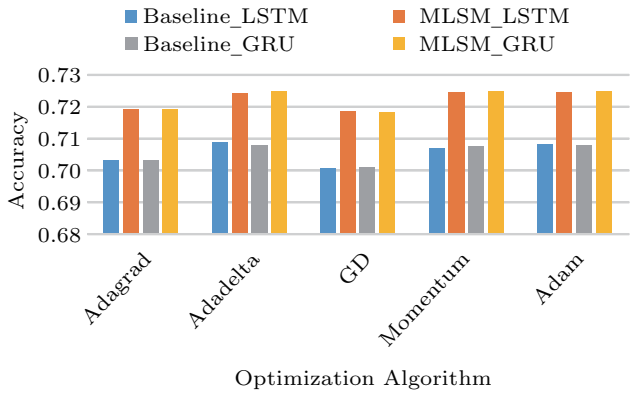


Fig.9. Accuracy study of RNN-based models with different optimization algorithms on SemEval-2016.

4.2 Experimental Results

In terms of accuracy of LSMs, we evaluate the accuracy metric within nine models which are five RNN-based models and four non-RNN LSMs. As shown in Table 4, RNN-based models have higher accuracy, i.e., they work better than non-RNN LSMs, especially with short texts. MLSM_LSTM, MLSM_GRU and MLSM@DASFAA enhance the effectiveness of baseline LSTM and GRU models, which shows the keyword impression helps the baseline models better understand texts. In most experiments (IMDBs where the length is less than 140, less than 200 and greater than 400, and SemEval-2016), MLSM_GRU achieves better scores than MLSM_LSTM and MLSM@DASFAA, which shows our development on GRU works better.

In order to measure the loss of RNN-based models, we evaluate the loss metric within four RNN-based models (MLSM_LSTM, Baseline_LSTM, MLSM_GRU, Baseline_GRU). Table 5 lists the evaluation results: MLSM-based models have less validation loss, i.e., they work better than baseline models.

Furthermore, in order to evaluate the robustness and flexibility of our improvements, we evaluate the four RNN-based models (MLSM_LSTM, Baseline_LSTM, MLSM_GRU, Baseline_GRU) with several optimization algorithms to demonstrate the effectiveness of our improvements. Five optimization algorithms are used, they are Adagrad[51], Adadelta[50], Gradient-Desent (GD), Momentum, and Adam[52]. Fig.8 and Fig.9 show the comparative results in term of accuracy: models with Adadelta and Adam work better; MLSM-based models have higher accuracy, in other words, obviously our method is superior to the baseline models.

Results above show that the proposed models are effective and robust, especially in dealing with short texts. The non-RNN LSMs decline in effectiveness with short texts, but the proposed models work stably and outperform the baseline LSTM and GRU.

5 Discussions

Compared with traditional LSMs, RNN-based models take into account the dependencies between words. The dependencies are conducive to understanding texts, and they are significantly effective especially when dealing with short texts. As shown in Table 4, the RNN models have higher accuracy than all the non-RNN LSMs. When dealing with short texts, we find the accuracy of the non-RNN LSMs drops sharply, and the effectiveness of the RNN-based models remains stable. Term dependencies could be utilized through the memory of words. The more words memorized, the more contextual information counted. Therefore, previous researches have focused on how to extend the length of memory[42,53].

Actually, memory length is not the sole factor to be considered in understanding texts. And every single word in a text does not contribute equally to the semantics. Some words in the text usually get more attention. In other words, our brain prefers to pick up keywords and to draw on memory to understand texts. Therefore, the keywords are more likely to catch readers' eyes. Through long-term training, people can enhance the semantic understanding by showing different memory for different words in texts. For example, when reading the sentence mentioned in Section 1: "Such a wonderful day. I like it.", we are impressed by the two words "wonderful" and "like", and other words in the sentence may be ignored unconsciously. The mechanism of distinguishing memory is helpful for people to grasp the key points, thus enhancing the understanding of texts. As we can see in Table 4, MLSM_LSTM and MLSM_GRU have the ability of distinguishing memory, and thus, they have higher accuracies than baselines. And the rest of the experiments show similar results.

6 Conclusions

RNN-based models are sequential models, capable of memorizing sequential data such as texts. In this paper, we showed the importance of keywords. They are beneficial for understanding the semantics. The proposed models (MLSM based on LSTM and MLSM based on GRU) are further used to perform sentiment analysis. MLSM outperforms the baseline LSTM and GRU respectively by 1%∼2% in terms of accuracy. Especially in dealing with short texts, the performance upgrade still exists. In addition, we obtained quite similar gains with different optimization algorithms. It shows the generalizability of our method. In other words, our method is general enough to be applied to different RNN-based models. In the future, we would like to apply our work in dealing with other NLP tasks, such as query suggestion, web search recommendation, and movie recommendation.

As we can see in Fig.3, the exponential decay of memory is unidirectional, yet people have an impression not only on the post-words but also on the pre-words. In the future, we would associate our model with bi-directional RNNs to address the problem.

References

[1] Wang G, Zhang Z, Sun J S, Sun J S, Yang S L, Larsonc C A. POS-RS: A random subspace method for sentiment classification based on part-of-speech analysis. *Information Processing & Management*, 2015, 51(4): 458-479.

[2] Hua W, Wang Z Y, Wang H X, Zheng K, Zhou X F. Short text understanding through lexical-semantic analysis. In *Proc. Int. Conf. Data Engineering*, April 2015, pp.495-506.

[3] Zou H, Tang X H, Xie B, Liu B. Sentiment classification using machine learning techniques with syntax features. In *Proc. Int. Conf. Computational Science and Computational Intelligence*, Dec. 2015, pp.175-179.

[4] Davuth N, Kim S R. Classification of malicious domain names using support vector machine and bi-gram method. *International Journal of Security and its Applications*, 2013, 7(1): 51-58.

[5] Bao S H, Xu S L, Zhang L, Yan R, Su Z, Han D Y, Yu Y. Mining social emotions from affective text. *IEEE Trans. Knowledge and Data Engineering*, 2012, 24(9): 1658-1670.

[6] Rao Y H, Lei J S, Liu W Y, Li Q, Chen M L. Building emotional dictionary for sentiment analysis of online news. *World Wide Web*, 2014, 17(4): 723-742.

[7] Stoyanov V, Cardie C. Annotating topics of opinions. In *Proc. the 6th International Conference on Language Resources and Evaluation*, May 31-June 1, 2008, pp.3213-3217.

[8] Cheng X Q, Yan X H, Guo Y Y, Guo J F. BTM: Topic modeling over short texts. *IEEE Trans. Knowledge and Data Engineering*, 2014, 26(12): 2928-2941.

[9] Wang Z Y, Zhao K J, Wang H X, Meng X F, Wen J R. Query understanding through knowledge-based conceptualization. In *Proc. the 24th Int. Conf. Artificial Intelligence*, July 2015, pp.3264-3270.

[10] Cheng J P, Wang Z Y, Wen J R, Yan J. Contextual text understanding in distributional semantic space. In *Proc. the 24th ACM Int. Conf. Information and Knowledge Management*, Oct. 2015, pp.133-142.

[11] Cui W Y, Zhou X Y, Lin H Y, Xiao Y H. Verb pattern: A probabilistic semantic representation on verbs. In *Proc. the 30th AAAI Conf. Artificial Intelligence*, March 2016, pp.2587-2593.

[12] Zhang X W, Wu B. Short text classification based on feature extension using the n-gram model. In *Proc. the 12th Int. Conf. Fuzzy Systems and Knowledge Discovery*, Aug. 2015, pp.710-716.

[13] López G J, Ruiz I M. Character and word baselines systems for irony detection in Spanish short texts. *Procesamiento de Lenguaje Natural*, 2016, 56: 41-48.

[14] Song G, Ye Y M, Du X L, Huang X H, Bie S F. Short text classification: A survey. *Journal of Multimedia*, 2014, 9(5): 635-643.

[15] Wang M, Lin L F, Wang F. Improving short text classification through better feature space selection. In *Proc. the 9th Int. Conf. Computational Intelligence and Security*, December 2013, pp.120-124.

[16] Wang B K, Huang Y F, Yang W X, Li X. Short text classification based on strong feature thesaurus. *Journal of Zhejiang University Science C*, 2012, 13(9): 649-659.

[17] Kim K, Chung B S, Choi Y, Lee S, Jung J Y, Park J. Language independent semantic kernels for short-text classification. *Expert Systems with Applications*, 2014, 41(2): 735-743.

[18] Fan X H, Hu H G. Construction of high-quality feature extension mode library for Chinese short-text classification. In *Proc. WASE Int. Conf. Information Engineering*, Aug. 2010, pp.87-90.

[19] Song Y Q, Wang H X, Wang Z Y, Li H S, Chen W Z. Short text conceptualization using a probabilistic knowledgebase. In *Proc. the 22nd Int. Joint Conf. Artificial Intelligence*, July 2011, pp.2330-2336.

[20] Kim D, Wang H X, Oh A. Context-dependent conceptualization. In *Proc. the 23rd Int. Joint Conf. Artificial Intelligence*, Aug. 2013, pp.2654-2661.

[21] Huang P S, He X D, Gao J F, Deng L, Acero A, Heck L. Learning deep structured semantic models for web search using clickthrough data. In *Proc. the 22nd ACM Int. Conf. Information & Knowledge Management*, Oct. 2013, pp.2333-2338.

[22] Shen Y L, He X D, Gao J F, Deng L, Mesnil G. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proc. the 23rd ACM Int. Conf. Information and Knowledge Management*, Nov. 2014, pp.101-110.

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

[24] Hu F, Xu X F, Wang J Y, Yang Z B, Li L. Memory-enhanced latent semantic model: Short text understanding for sentiment analysis. In *Proc. Int. Conf. Database Systems for Advanced Applications*. March 2017, pp.393-407.

[25] Hofmann T. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 2001, 42(1/2): 177-196.

[26] Wang J, Peng J X, Liu O. A classification approach for less popular webpages based on latent semantic analysis and rough set model. *Expert Systems with Applications*, 2015, 42(1): 642-648.

[27] Ke X H, Luo H J. Using LSA and PLSA for text quality analysis. In *Proc. Int. Conf. Electronic Science and Automation Control*, Jan. 2015, pp.289-291.

[28] Anoop V S, Prem S C, Asharaf S, Alessandro Z. Generating and visualizing topic hierarchies from microblogs: An iterative latent dirichlet allocation approach. In *Proc. Int. Conf. Advances in Computing, Communications and Informatics*, Aug. 2015, pp.824-828.

[29] Gao J F, Toutanova K, Yih W T. Clickthrough-based latent semantic models for web search. In *Proc. the 34th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, July 2011, pp.675-684.

[30] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks. *Science*, 2006, 313(5786): 504-507.

[31] Bengio Y, Ducharme R, Vincent P, Janvin C. A neural probabilistic language model. *Journal of Machine Learning Research*, 2003, 3(2): 1137-1155.

[32] Huang E H, Socher R, Manning C D, Ng A Y. Improving word representations via global context and multiple word prototypes. In *Proc. the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, July 2012, pp.873-882.

[33] Salakhutdinov R, Hinton G. Semantic hashing. *International Journal of Approximate Reasoning*, 2009, 50(7): 969-978.

[34] Mikolov T, Karafiát M, Burget L, Ćernocký J, Khudanpur S. Recurrent neural network based language model. In *Proc. the 11th Annual Conference of the International Speech Communication Association*, Sept. 2010, 1045-1048.

[35] Mikolov T. Statistical language models based on neural networks. http://www.fit.vutbr.cz/∼imikolov/rnnlm/google.pdf, March 2015.

[36] Williams R J, Zipser D. Gradient-based learning algorithms for recurrent networks and their computational complexity. In *Backpropagation: Theory, Architectures, and Applications*, Chauvin Y, Rumelhart D E (eds.), Lawrence Erlbaum Associates, Inc., 1995, pp.433-486.

[37] Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. In *Proc. the 30th Int. Conf. Machine Learning*, June 2013, pp.1310-1318.

[38] Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 1998, 6(2): 107-116.

[39] Olah C. Understanding LSTM networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/, Sept. 2016.

[40] Gers F A, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 2000, 12(10): 2451-2471.

[41] Gers F A, Schmidhuber J. Recurrent nets that time and count. In *Proc. the IEEE-INNS-ENNS Int. Joint Conf. Neural Networks*, July 2000.

[42] Greff K, Srivastava R K, Koutník J, Steunebrink B R, Schmidhuber J. LSTM: A search space odyssey. *IEEE Trans. Neural Networks and Learning Systems*, 2015, PP(99): 1-11.

[43] Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*, 2014. http://arxiv.org/abs/1406.1078, Sept. 2016.

[44] Esuli A, Sebastiani F. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proc. the 5th Conf. Language Resources and Evaluation*, May 2006, pp.417-422.

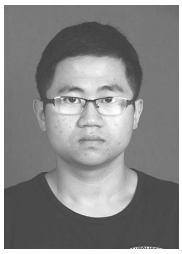
[45] Baccianella S, Esuli A, Sebastiani F. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proc. the 7th Conf. Int. Language Resources and Evaluation*, Jan. 2010, pp.2200-2204.

[46] Miller G A. WordNet: A lexical database for English. *Communications of the ACM*, 1995, 38(11): 39-41.
[47] Maas A L, Daly R E, Pham P T, Huang D, Ng A Y, Potts C. Learning word vectors for sentiment analysis. In *Proc. the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, June 2011, pp.142-150.
[48] Nakov P, Ritter A, Rosenthal S, Sebastiani F, Stoyanov V. Evaluation measures for the semeval-2016 task 4: Sentiment analysis in twitter. http://alt.qcri.org/semeval2016/task4/, Feb. 2017.
[49] LeCun Y, Bottou L, Orr G B, Müller K R. Efficient backprop. In *Neural Networks: Tricks of the Trade*, Orr G B, Müller K R (eds.), Springer, 2012, pp.9-50.
[50] Zeiler M D. ADADELTA: An adaptive learning rate method. *arXiv:1212.5701*, 2012. https://arxiv.org/abs/1212.5701, Sept. 2016.
[51] Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011, 12: 2121-2159.
[52] Kingma D, Ba J. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. https://arxiv.org/abs/1412.6980, Sept. 2016.
[53] Graves A, Wayne G, Danihelka I. Neural Turing machines. *arXiv:1410.5401*, 2014. https://arxiv.org/abs/1410.5401, Sept. 2016.

Fei Hu is a Ph.D. candidate in the College of Computer and Information Science, Southwest University, Chongqing. His research interests include deep learning technologies and natural language processing.

Li Li received her Ph.D. degree in computer science from Swinburne University of Technology, Melbourne, in 2006. She is currently a professor in the College of Computer and Information Science at Southwest University, Chongqing. Her research interests include data analysis, data mining and machine learning. She is a senior member of CCF and CAAI, and a member of ACM and IEEE CS.

Zi-Li Zhang is a professor at Southwest University, Chongqing. He received his B.S. degree from Sichuan University, Chengdu, in 1986, M.E. degree from Harbin Institute of Technology, Harbin, in 1989, and Ph.D. degree from Deakin University, Victoria, in 2001, all in computing. He authored or co-authored more than 130 refereed papers in international journals or conference proceedings, one monograph, and six textbooks. His research interests include bio-inspired artificial intelligence, agent-based computing, and big data analysis.



Jing-Yuan Wang is a graduate student in the College of Computer and Information Science, Southwest University, Chongqing. His research interest primarily includes deep learning, attention models and machine translation.

Xiao-Fei Xu is an undergraduate student in the College of Computer and Information Science, Southwest University, Chongqing. His research interest primarily includes word embedding, deep learning, and big data management.