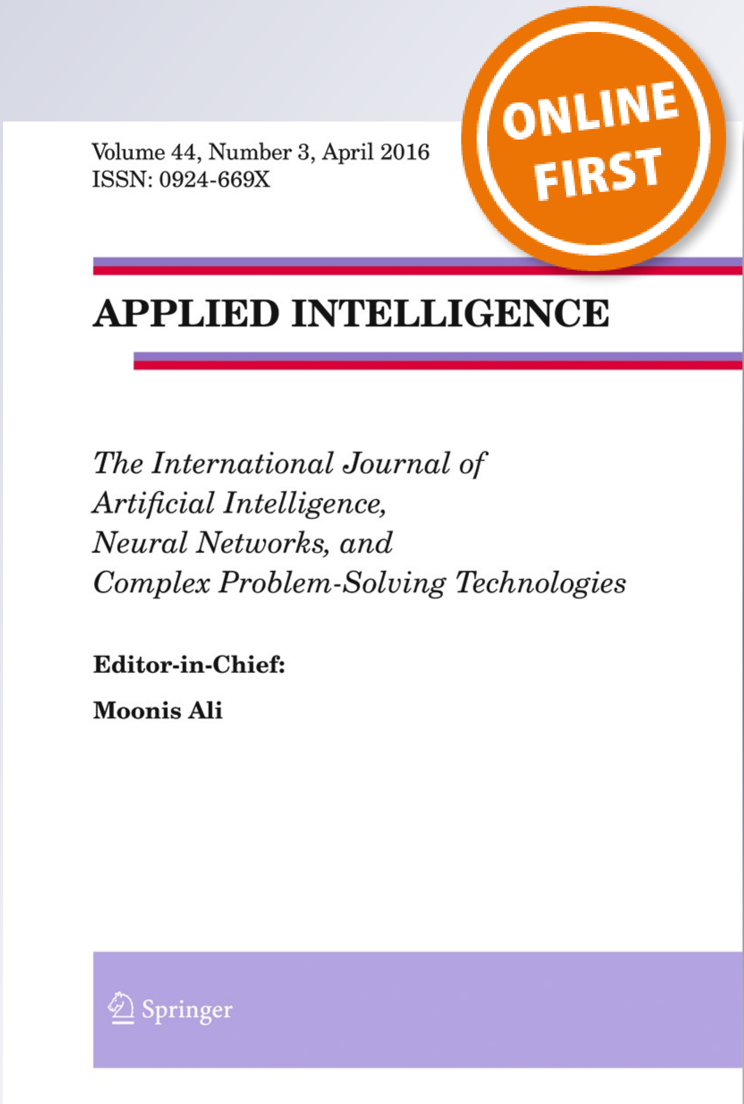
Exploring sentiment parsing of microblogging texts for opinion polling on chinese public figures

Jiajun Cheng, Xin Zhang, Pei Li, Sheng Zhang, Zhaoyun Ding & Hui Wang

Applied Intelligence
The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies

ISSN 0924-669X

Appl Intell
DOI 10.1007/s10489-016-0768-0



Springer

Appl Intell
DOI 10.1007/s10489-016-0768-0

Exploring sentiment parsing of microblogging texts for opinion polling on chinese public figures

Jiajun Cheng[1] · **Xin Zhang**[1] · **Pei Li**[1] · **Sheng Zhang**[1] · **Zhaoyun Ding**[1] · **Hui Wang**[1]

Abstract Microblogging websites such as twitter and Sina Weibo have attracted many users to share their experiences and express their opinions on a variety of topics, making them ideal platforms on which to conduct electronic opinion polls on products, services and public figures. However, conventional sentiment analysis methods for microblogging messages may not meet the demands of opinion polls for public figures. Therefore, in this study, we focus mainly on the problem of sentiment analysis for opinion polling on Chinese public figures. We propose a sentiment parsing-based architecture, which represents and labels opinion targets and their corresponding sentiments jointly to avoid the mismatching of them, for opinion poll of public figures using microblogs. Furthermore, we formulate sentiment parsing of microblogging sentences as a sequence labeling problem and adapt different Recurrent Neural Network (RNN) models to train and infer the model. Our experimental results demonstrate that the proposed sentiment parsing-based methods achieve better performance than conventional sentiment score-based methods in opinion polling on public figures using microblogs.

Keywords Microblogs · Opinion poll · Sentiment parsing · Sequence labeling · RNN

✉ Jiajun Cheng
jiajun.cheng@nudt.edu.cn

Xin Zhang
zhangxin@nudt.edu.cn

Pei Li
peili@nudt.edu.cn

Sheng Zhang
zhangsheng@nudt.edu.cn

Zhaoyun Ding
zyding@nudt.edu.cn

Hui Wang
huiwang@nudt.edu.cn

[1] College of Information Systems and Management, National University of Defense Technology, Changsha, People's Republic of China

1 Introduction

Microblogging websites—e.g., Twitter and Sina Weibo—allow users to share small digital content and maintain contact with the world through various communication services such as smart-phones and Web interfaces. The easy accessibility, free message format and loose content censorship make them ideal platforms for people to freely express their opinions on a variety of topics and discuss current issues. In particular, an increasing number of microblogging users comment on the products and services that they use, public figures, including politicians, singers, actors, or sports stars, and personal, local or global events. Hence, microblogging websites become invaluable sources of public opinions and have motivated researchers to investigate how to conduct electronic polls using microblog data for products [1, 2] or election candidates [3–7], because they are considerably faster and less expensive, and can provide much larger samples compared with conventional surveys and polling methodology.

Generally, electronic polling based on microblogging data consists of three steps—namely, message retrieval, (message-level) sentiment analysis, and sentiment aggregation. The first step is to identify from the sheer amount of streaming microblogging messages, henceforth called microblogs for short, those that are subjective and related to

Published online: 12 March 2016

(1) At Hillary's worst last month she wasn't a joke, Jeb is a joke.
(2) I'm telling you, Russell Westbrook is better than Steph curry.
(3) 邓紫棋拍出来的效果没有汤唯郭采洁拍出来的感觉有质感。(Gloria Tang did not take out the effect of the shot out of the texture feeling as Tang Wei and Amber Kuo did.)
(4) 哈哈哈，我所有的电子产品的背景全是邓紫棋，莱昂纳多现在太胖了。(Ha ha ha, I have all the electronic products in the background of Gloria Tang, Leonardo is now too fat.)

Fig. 1 Example messages collected from Twitter and Weibo, which have different sentiments on different subjects. Message (1) expresses positive sentiment toward "Hillary" and negative sentiment toward "Jeb." Message (2) has positive sentiment toward "Russell Westbrook" and negative sentiment toward "Steph." Message (3) expresses positive sentiment toward "Tang Wei" and "Amber Kuo" and negative sentiment toward "Gloria Tang." Message (4) expresses positive sentiment toward "Gloria Tang" and negative sentiment toward "Leonardo"

the polling subjects (e.g., the candidates in a political election or products and brands of interest). The second step aims at extracting from each microblog the sentiment—i.e., positive, negative, or neutral— born by the author toward the polling subjects. The last one is to estimate the collective sentiments of microblogging users by aggregating message-level sentiment analysis results. Among these steps, as revealed in [5] and [8], sentiment analysis is very critical for improving the final polling results. However, nearly all existing techniques are based on the implicit hypothesis that the sentiments expressed in a single microblog by the author are the same toward all mentioned polling subjects, because they assign a sentiment score, or rather a sentiment label, to each microblog and assess the collective approval rating of each subject by aggregating all sentiment scores of microblogs mentioning it (or him or her). This is truly problematic for accurate sentiment evaluation for two reasons: First, although microblogs are innately very terse, there indeed exist many that express different (often opposing) sentiments toward competing subjects in a single post. See Fig. 1 for some examples collected from Twitter and Weibo. Second, in contrast to news articles, microblogs are authored by ordinary users and hence are frequently subject to typos, slang, informal abbreviations, indirect expressions, and so on (as exemplified in Fig. 2), which all may render lexicon-based sentiment determination ineffective.

The issues raised above motivate the work presented in this paper, which focuses mainly on the problem of sentiment analysis for opinion polling on Chinese public figures and manages to improve the overall accuracy. Specifically, to address the first issue, we identify a new research problem, called *sentiment parsing*, aiming at extracting ⟨sentiment, subject⟩ pairs from sentences in microblogs discussing public figures of interest. To address the second issue, we propose formulation of sentiment parsing as a sequence labeling problem to mine contextual information that is helpful in enhancing sentiment determination accuracy, particularly when indirect expressions are adopted. Further, because Recurrent Neural Networks (RNNs) have achieved excellent performance in many sequence labeling [9, 10] and nature language processing [11, 12] tasks, we adapt RNNs to train and infer our sequence labeling model. More concretely, this paper makes the following contributions:

– As mentioned above, we identify a new research problem—i.e., sentiment parsing—which is defined to be the process of identifying all ⟨sentiment,

(1) Lakers lost again, this is making me sad.
(2) damn, the San Antonio Spurs lost to Detroit on a buzzer beater.
(3) 真尼玛无聊，怪到成龙大哥身上了。(What fuck boring to blame on Jackie Chan.)
(4) 哇咔咔，我已经把所有 360 软件都卸了。(Wow Kaka, I've uninstalled all softwares of 360.)

Fig. 2 Example messages collected from Twitter and Weibo, which express sentiment toward subjects implicitly. Messages (1)–(3) express negative sentiment in the overall message, but it can be inferred that they have a positive sentiment toward "Lakers," "San Antonio Spurs," and "Jackie Chan," respectively. Message (4) has a positive sentiment in the overall message, but expresses positive sentiment toward "360" subject⟩ pairs from each input (microblogging) sentence and hence promising for improving the accuracy of message-level sentiment analysis.

- To better model the contextual information, we formulate sentiment parsing of microblogging sentences as a sequence labeling problem, and adapt RNNs to train and infer the model. To select the best-performing RNN model, we empirically compare different models with different parameter settings with respect to recurrent units, arc direction and network depth.
- To the best of our knowledge, this paper is the first to investigate electronic polling of Chinese public figures using microblogs. Accordingly, in our work, we focus mainly on microblogs in Chinese. Because no existing corpus can be used, we construct a Chinese microblogs dataset for sentiment parsing by manually labeling messages collected from Weibo.
- The experimental results demonstrate that our proposed RNN-based sentiment parsing method increases the F-Score from 0.634 of the conventional classification-based method to 0.736.

The rest of this paper is structured as follows. Section 2 overviews the related work. Section 3.2 presents the general definition of sentiment parsing and an adapted version for electronic polling on public figures using microblogs. Section 3 details the methodology we proposed. Section 4 describes the experimental results. Finally, Section 5 concludes the paper and discusses future work.

2 Related work

Three broad classes of related work pertinent to this study—i.e., microblog based opinion poll, sentiment analysis and recurrent neural networks—are briefly surveyed in this section.

2.1 Microblog based opinion poll

Many studies have aimed to conduct opinion polling on public figures, products and services using microblogs. Tumasjan et al. [4] and Sang et al. [6] simply used the numbers of tweets that mentioned the corresponding candidates to predict the 2009 German federal election and the 2011 Dutch senate election, respectively. However, Metaxas et al. [13] pointed out that their methods are inaccurate and that sentiment analysis of tweets is an important factor in opinion polling via Twitter. O'Connor et al. [3] and Kagan et al. [14] computed a sentiment score for each tweet and summarized the scores of tweets containing the candidates to predict the approval ratings in elections. In addition, Luong et al. [15] used a sentiment lexicon to compute a sentiment score for each tweet and conducted an opinion poll on the rail-way-service of Los Angeles. Most prior studies of microblog-based opinion polling considered sentiments of tweets but extracted sentiments and targets separately, which may cause mismatch between sentiments and targets. In addition, Jansen et al. [2] considered not only the sentiment polarity of products but also the sentiment patterns of different brands to investigate the word-of-mouth process of products on Twitter. However, sentiment pattern extraction is often domain specific and may overcome some problems inherent to opinion polling on public figures. Therefore, in this paper, we focus on the study of sentiment parsing in microblogs, which extracts sentiments and targets jointly to better support opinion polling on public figures.

2.2 Sentiment analysis

Sentiment analysis, also called *opinion mining*, is used to discover all opinions expressed by the author of a given document [16]. In such a process, the opinion is often formulaically represented to be a quintuple $\langle H, E, A, S, T\rangle$, where H is the opinion holder, E is the targeted entity, A is the targeted aspect of E, S is the sentiment, and T is the time when the opinion is expressed. In this study, we call each element in the above quintuple an opinion element, and we call E and A opinion targets.

Most existing studies focus only on S by sentiment classification methods, which assign a sentiment score or a sentiment polarity to each input document. Some studies used lexicon-based methods [17, 18] for sentiment classification, which often require an expert-defined sentiment lexicon and predict the polarity of a sentence or document by analyzing the patterns of occurrence for such words in text. Lexicon-based methods perform well in domain-specific sentiment classification tasks, but the lexicon and patterns are difficult to transfer from one domain to another. Therefore, Pang et al. [19] used supervised learning for sentiment classification, and various learning-based methods have since been put forward for sentiment classification [20–22]. Learning-based methods need only to construct a new training set when the task changes, which is much easier than defining a new sentiment lexicon and patterns. However, microblog data bring new challenges to sentiment classification because of the terse, colloquial, and sometimes obscure writing style therein. Therefore, some other researchers took full advantage of resources in social media and proposed various other methods to classify the sentiment of social media users [23, 24].

Sentiment classification is significant in overall opinion trend analysis, but it is insufficient to construct opinion quintuples because opinion targets are unknown. Therefore, some research extracts opinion targets by keywords as supplementary [3, 6]. However, this pipelined paradigm, which extracts sentiment and opinion target sequentially and separately, may cause notable errors when the sentiment is not about the target extracted or there are different sentiments to different targets in the text. Other researches have used rule-based methods [25, 26] and sequence labeling-based methods [2, 27, 28] to extract opinions. However, rule-based approaches are often domain specific and expensive, if not difficult, to extend to new domains. Moreover, these techniques often require much prior knowledge such as sentiment lexicon and semantic rules, which may be impossible to adapt properly to Chinese microblogs. Sequence labeling based methods are often used for aspect extraction [27] of reviews and coarse-grained opinion expression detection [2, 28]. To the best of our knowledge, no study has used sequence labeling-based methods to extract opinions in microblogging texts. Therefore, to fill the void left by prior studies, we attempt to represent and extract opinion targets and sentiments jointly with a sequence labeling method.

2.3 Recurrent neural network

In recent years, artificial neural networks have achieved important empirical successes in a number of applications such as computer vision [29, 30], natural language processing [31, 32], knowledge construction [33–35] and reasoning [36, 37]. Particularly, Garcez et al. [33] analyzed the problems of symbolic knowledge extraction from trained neural networks and presented a new approach for knowledge extraction from trained networks. Rosaci et al. [34] proposed a neural symbolic network-based architecture to build agent ontologies for supporting agent mutual monitoring. Recurrent Neural Networks (RNNs), which constitute an important class of naturally deep neural network architectures, and has been applied to many sequential prediction tasks. Recent studies have shown that RNNs perform well in many sequence processing tasks such as speech recognition [38, 39] and handwriting recognition [10, 40]. In natural language processing tasks, RNNs view a sentence or document as a sequence of tokens and achieve excellent performance in language modeling [11], semantic parsing [12, 41] and named entity recognition [42, 43]. In addition, some variant models of RNN such as LSTM [44, 45] and GRU [46, 47] have been proposed to overcome the gradient vanishing problem and achieved good performance in different tasks. Moreover, Irsoy et al. [28] explored bidirectional and deep RNN models in opinion expression detection and achieved a better result than the state-of-the-art method. These studies have shown that RNN-based methods outperform conventional sequence labeling methods such as CRFs and HMMs in most sequence labeling tasks in natural language processing. Because we plan to use sequence labeling to extract opinion elements jointly, we choose RNN as the sequence labeling model. In addition, because different RNN models achieve the best performance in different tasks, we explore different RNN models in three kinds of parameters: different recurrent units, unidirectional or bidirectional models and the depth of a deep RNN.

3 Methodology

The main steps to perform opinion polling of Chinese public figures via sentiment parsing of microblogging texts are described in this section. First, we briefly introduce the architecture in Section 3.1. We then define the sentiment parsing task of microblogging sentences in Section 3.2 and formulate it as a sequence labeling problem in Section 3.3. After that, we introduce basic theories of different RNN models in Section 3.4, and adapt RNN models to accomplish the sequence labeling task in Section 3.5. Finally, the postprocessing method, which removes the unrelated targets and maps the extracted words of targeted entities to their real names, is presented in Section 3.6.

3.1 Overview of the architecture

As shown in Fig. 3, there are three main steps in the architecture: data collection, sentiment parsing and opinion aggregation.

The purpose of data collection is to collect microblogs related to given public figures. Public figures are usually involved in some hot topics, and microblog users can

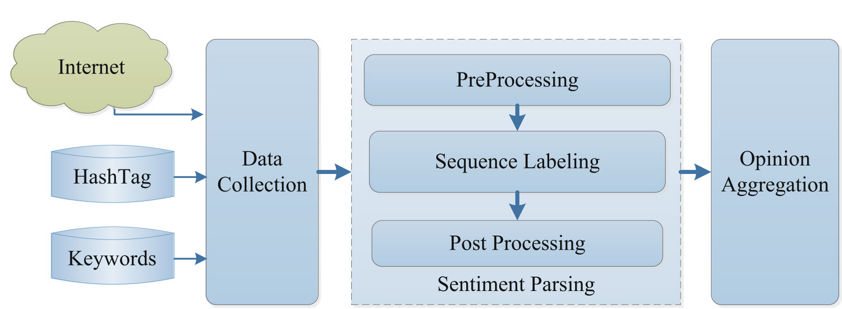


Fig. 3 Architecture of sentiment parsing based opinion poll of public figures in microblogs

express their opinions through discussion. Thus, we focus on hot topics related to given public figures to conduct opinion polling efficiently. Specifically, we use hashtags, which define the related topics, and keywords to collect the microblogs of interest.

In the sentiment parsing step, the opinions on public figures in microblogs are extracted. Sentiment parsing contains preprocessing, sequence labeling and postprocessing. In the first step, the sentences that mention the targeted public figures are selected, and the sentences are segmented to be token sequences in the preprocessing step. After that, sentiment parsing is represented as a sequence labeling problem, and an RNN-based sequence labeling method is proposed to extract the opinions in the sentence. Finally, the unrelated targets are removed, and the extracted opinion targets are mapped into their real names in the postprocessing step.

Finally, the extracted opinions on public figures are aggregated in the opinion aggregation step to generate the result of the opinion poll.

3.2 Definition of sentiment parsing

It is often a challenging problem to extract opinion quintuples from Web documents such as news, forums and microblogs, because they are always unstructured free texts, and opinion elements should thus be extracted according to the semantic meaning of the texts. Generally, a sentence expresses a relatively independent and complete thought, although it may make little sense if taken in isolation out of context. Opinion elements can be extracted at the sentence level and then summarized at the document level. Therefore, we define ***Sentiment Parsing (SP)*** *to be the task of extracting opinion elements in a given sentence, which can be considered as a special case of syntactic parsing—viz., sentiment oriented syntactic parsing*. Sentiment parsing should not only extract the opinion elements in a sentence but also determine the corresponding relationships among them.

For microblogs, the author and posting time can often be taken directly as H and T, respectively. This motivates us to identify the extraction of the remaining elements—i.e., $\langle E, A, S\rangle$—as the core task of sentiment parsing of microblogging sentences. Moreover, because the target aspects are often expressed implicitly (e.g., via the discourse context or the background topics) in Chinese microblogs, we further exclude aspect (i.e., A) extraction from the task of sentiment parsing. Thus, in this study, we focus on extracting E and its corresponding S. In addition, S can generally be a sentiment expression, sentiment word or merely sentiment polarity expressed by the opinion holder. Considering the fact that Chinese microblogging texts usually involve implicate expressions and the sentiment expressions are difficult to extract, we define S simply to be positive or negative.

In summary, we define sentiment parsing of Chinese microblogging sentences in this study to extract the mentioned entity E and the corresponding sentiment polarity S. Because S is defined to be positive or negative, it can be looked upon as an attribution of the corresponding E in sentences; thus, the two elements can be extracted jointly via sequence labeling.

3.3 Sentiment parsing via sequence labeling

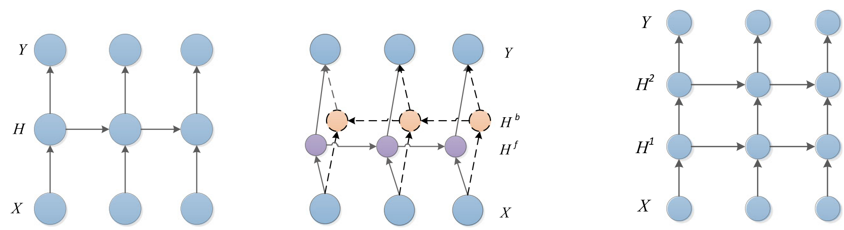
In this subsection, we define a novel label schema to formulate sentiment parsing as a sequence labeling task. Let $S = \{t_1, t_2, \ldots, t_s\}$ denote a sentence instance, where t_j represents the jth token in it, and $L = \{l_1, l_2, \ldots, ls\}$ denotes the label sequence of S, where $l_j \in \Omega$ is the label of t_j. The task of sequence representation is to define a suitable value set Ω which ensures that L can meet the demand of the labeling task. Sentiment parsing is a discovery problem, in which most of the labels in a sequence are the label which indicates that the token is out of any target. Because the words of a target may not be segmented into one token, the first task of sequence representation is to define the boundary of targets. In addition, aiming to avoid mismatching of opinion target and sentiment, we attempt to label the two elements jointly. Therefore, the labels should support different kinds of targets(i.e. target with positive sentiment and target with negative sentiment).

Considering that the forms of address of public figures are known and will be used in the postprocessing step to find the related targets from all extracted targets, we attempt to weaken the boundary and enhance the discrimination of different sentiments. Namely, we define $\Omega = \{P, N, O\}$, where P indicates tokens inside a positive target, N denotes tokens inside a negative target, and O indicates tokens outside any targets. We connect tokens with continuous Ps to be a candidate target with positive sentiment and words with continuous Ns in contrast. As shown in Table 1, the candidate opinion tuples of the sentence are ⟨*Person A*, positive⟩ and ⟨*Person B*, negative⟩. However, the "PBO" tagging scheme cannot affirm the boundary of each target accurately and will connect the targets into one candidate target when

Table 1 An example sentence labeled with the "PNO" tagging scheme

S	I	support	*Person*	*A*	,	*Person*	*B*	is	a	cheater	.
L	O	O	P	P	O	N	N	O	O	O	O

Without loss of generality, we use "*Person A*," "*Person B*," etc. to represent the real names of entities in examples and the follow-up of this paper



(a) SRNN. (b) BSRNN. (c) DSRNN.

Fig. 4 Structures of RNNs. **a** is the simple RNN, which has the basic structure of RNNs. **b** is the bidirectional version of SRNN and **c** is an instance of deep SRNN at depth 2

two or more targets with the same sentiments appear continuously. This problem will be solved in the postprocessing step (see Section 3.6).

3.4 RNN models

RNNs are a class of artificial neural network architectures that use iterative function loops to store information [45]. Benefiting from the recurrent network structure, RNNs can process sequences with various lengths better than other artificial neural networks. Because recent studies have shown that RNN-based methods outperform conventional sequence labeling methods such as CRFs and HMMs in most sequence labeling tasks in natural language processing [12, 28, 41–43], we choose RNN models to accomplish the sequence labeling task as well. In this subsection, we introduce popular RNN models briefly in three kinds of parameters: different recurrent units, unidirectional or bidirectional models and deep models.

3.4.1 Simple RNN

We first introduce Simple RNN (SRNN), which has the basic structure of RNN models. As shown in Fig. 4a, SRNN has three layers (i.e. input layer X, hidden layer H and output layer Y.). The input layer $X = \{x_1, x_2, ..., x_n\}$ represents the input sequence, where the jth node is represented as a fix-length vector x_j. The output layer $Y = \{y_1, y_2, ..., y_n\}$ represents the predict sequence of input sequence, where the jth node is a vector y_j, which determines the label of x_j. H and Y can be represented as follows:

$$h_j = f(Wx_j + Uh_{j-1} + b_h), \quad (1)$$

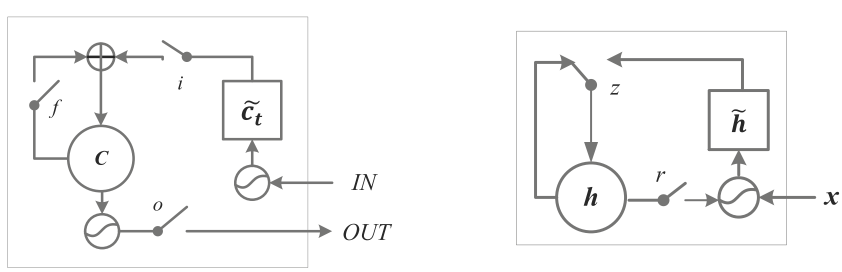
$$y_j = g(Vh_j + b_y), \quad (2)$$

where h_j is the vector of the jth node of the hidden layer. W, U, V are weight matrices, and b_h, b_y are bias vectors. f is the nonlinear active function (e.g., $tanh$ and sigmoid function), and g is the output nonlinearity—e.g., softmax function.

Generally, there exist two shortcomings of SRNN: (1) it suffers from the vanishing gradient problem, which means that the force of the first input decays over time as new inputs overwrite the activations of the hidden layer [44, 45]. Therefore, some problems may occur when processing sequences with long time association. (2) The output of each token is determined by itself and the tokens before it while ignoring the tokens after it. Therefore, it may not meet the demand when the label is also depending on the tokens after it.

3.4.2 Mutations of recurrent units

To solve problem (1), many different mutations have been proposed, among which Long Short Time Memory (LSTM) [44] is the most famous. LSTM attempts to avoid gradient vanishing by adding a memory block to each node in the hidden layer. Figure 5a displays a memory block, which



(a) LSTM. (b) GRU.

Fig. 5 Mutations of recurrent units: **a** the memory block of LSTM, **b** gated recurrent unit contains a memory cell c and three gates (the input gate i, the forget gate f and the output gate o). For the memory block of the jth node of the hidden layer, the input "IN" is $\langle x_j, h_{j-1}\rangle$, and the output "$OUT$" is h_j. The memory block first calculates the candidate memory $\tilde{c_j}$ in time j, which is similar to h_j in SRNN in (1), by

$$\tilde{c_j} = tanh(W_c x_j + U_c h_{j-1} + b_c). \tag{3}$$

After that, the memory c_j in time j is built by the candidate memory $\tilde{c_j}$, controlled by an input gate i_j, and the last memory c_{j-1}, controlled by a forget gate f_j:

$$i_j = sigmoid(W_i x_j + U_i h_{j-1} + V_i c_{j-1} + b_i), \tag{4}$$
$$f_j = sigmoid(W_f x_j + U_f h_{j-1} + V_f c_{j-1} + b_f), \tag{5}$$
$$c_j = i_j \odot \tilde{c_j} + f_j \odot c_{j-1}. \tag{6}$$

Finally, the memory c_j is outputted to h_j through an output gate o_j by

$$o_j = sigmoid(W_o x_j + U_o h_{j-1} + V_o c_j + b_o), \tag{7}$$
$$h_j = o_i \odot tanh(c_j). \tag{8}$$

In (3) to (8), W_*, U_* and V_* are weighted matrices, and b_* are the biases. The $\odot$ operation denotes the element-wise vector product. As we can see, LSTM keeps c_j and h_j two units for each time step. The memory cell c keeps the long-term memory, and the three gates determine whether long-term memory or short-term memory is used for each time step. Therefore, with a memory block to each hidden node, LSTM can overcome the gradient vanishing problem well and better use the features from long-term memory and short-term memory.

However, the memory block of LSTM brings many new parameters, which may be difficult to justify and may not be needed for a good result. Therefore, Cho et al. [46] proposes the Gated Recurrent Unit (GRU) as the activation of the hidden layer, and research shows that it outperforms LSTM on a slot of tasks [47, 48]. As shown in Fig. 5b, GRU does not have a special memory cell and keeps h_j only for each hidden node. Apart from h_j, a gated recurrent unit contains a candidate hidden node $\tilde{h_j}$ and two gates (the reset gate r and the update gate z). For time step j, GRU first computes the reset gate r_j, which determines whether to use the previous hidden node when calculating the candidate hidden node $\tilde{h_j}$ by

$$r_j = sigmoid(W_r x + U_r h_{j-1} + b_r). \tag{9}$$

The candidate hidden node $\tilde{h_j}$ is then computed by

$$\tilde{h_j} = tanh(W_h x + U_h(r_j \odot h_{j-1}) + b_h). \tag{10}$$

After that, the update gate z_j, which determines the weights of candidate hidden node $\tilde{h_j}$ and the previous hidden node h_{j-1}, is computed and the hidden node h_j is updated according to update gate z_j as

$$z_j = sigmoid(W_z x + U_z h_{j-1} + b_z), \tag{11}$$
$$h_j = z_j h_{j-1} + (1 - z_j)\tilde{h_j}. \tag{12}$$

GRU removes the memory cell of the LSTM memory block, making the model much more concise. However, with the reset gate and update gate, GRU can still control whether to use previous long-term memory or new short-term memory. If the update gate z is near 1, the output is very biased toward the previous (or long-term) memory. In contrast, if the update gate and the reset gate are both near 0, the output overweighs the current time step (or short-term memory).

In addition, Jozefowicz et al. [48] conducted a thorough architecture search and found the three best architectures, which outperformed LSTM and GRU in some tasks, from thousands of mutations of RNN architectures. The formulations of the three architectures to calculate the hidden layer are listed from (13) to (24). Because the three architectures are quite similar to GRU, we call them Mutational GRUs(MGRUs) in this paper.

MGRU1:

$$r_j = sigmoid(W_r x + U_r h_{j-1} + b_r), \tag{13}$$
$$z_j = sigmoid(W_z x + b_z), \tag{14}$$
$$\tilde{h_j} = tanh(tanh(x) + U_h(r_j \odot h_{j-1}) + b_h), \tag{15}$$
$$h_j = z_j \odot h_{j-1} + (1 - z_j) \odot \tilde{h_j}. \tag{16}$$

MGRU2:

$$r_j = sigmoid(x + U_r h_{j-1} + b_r), \tag{17}$$
$$z_j = sigmoid(W_z x + U_z h_{j-1} + b_z), \tag{18}$$
$$\tilde{h_j} = tanh(W_h x + U_h(r_j \odot h_{j-1}) + b_h), \tag{19}$$
$$h_j = z_j \odot h_{j-1} + (1 - z_j) \odot \tilde{h_j}. \tag{20}$$

MGRU3:

$$r_j = sigmoid(W_r x + U_r h_{j-1} + b_r), \tag{21}$$
$$z_j = sigmoid(W_z x + U_z tanh(h_{j-1}) + b_z), \tag{22}$$
$$\tilde{h_j} = tanh(W_h x + U_h(r_j \odot h_{j-1}) + b_h), \tag{23}$$
$$h_j = z_j \odot h_{j-1} + (1 - z_j) \odot \tilde{h_j}. \tag{24}$$

3.4.3 Bidirectional RNN

To solve problem (2), Bidirectional RNN (BRNN) was proposed [49] by adding a backward sequence in the hidden layer; thus, the output of each token is determined by the whole sequence. It is worth noting that all RNN models we introduced above have their bidirectional versions. Taking Bidirectional Simple RNN (BSRNN) as an example, as shown in Fig. 4b, BRNN has two parallel hidden layers (one is in the forward direction, and another is in the backward direction), and the output layer is computed by the combination of two hidden layers:

$$h_j^f = f\left(W^f x_j + U^f h_{j-1}^f + b_h^f\right), \tag{25}$$

$$h_j^b = f\left(W^b x_j + U^b h_{j-1}^b + b_h^b\right), \tag{26}$$

$$y_j = g\left(V^f h_j^f + V^b h_j^b + b_y\right), \tag{27}$$

where h_j^f and h_j^b are the vectors of the jth node of the hidden layer in the forward and backward directions. W^f, U^f, V^f and b_h^f are the forward parameters, and W^b, U^b, V^b and b_h^b are the backward parameters.

3.4.4 Deep RNN

As is knowm, neural networks can be stacked to construct deep neural networks, which may have better learning capacity than shallow neural networks [50]. Irsoy et al. [28] explored different depths of SRNN and BSRNN for the task of opinion expression detection and obtained the best performance with a bidirectional deep model. This motivates us to explore the deep models of all mutational RNN models we introduced above and their bidirectional versions.

There are different ways to construct Deep RNNs (DRNNs) [8]. In this study, we use the simplest way, the stacking method, to construct deep RNNs, i.e., by taking the output of the lower hidden layer as the input of the higher hidden layer. We define the depth of a DRNN as the number of its hidden layers. Figure 4c displays the structure of a deep SRNN with depth 2, which has two hidden layers (H^1 and H^2), and the output of H^1 is the input of H^2:

$$h_j^1 = f\left(W^1 x_j + U^1 h_{j-1}^1 + b_h^1\right), \tag{28}$$

$$h_j^2 = f\left(W^2 h_j^1 + U^2 h_{j-1}^2 + b_h^2\right), \tag{29}$$

$$y_j = g\left(V h_j^2 + b_y\right), \tag{30}$$

where W^* and U^* are the weight matrices of the hidden layers, and b^* are the biases of the hidden layers.

3.5 RNN-based sequence labeling

With regard to the sequence labeling task of sentiment parsing using RNN, a sentence must be transformed into an input sequence X, and the label sequence should be determined with the output layer Y of the RNN.

Input After preprocessing, a sentence $S = \{t_1, t_2, ..., t_s\}$ becomes a token sequence. We define D as the dictionary containing all tokens of the dataset and $\Phi \in R^{l\times|D|}$ as the word-vector matrix that contains a special vector of regular length l for each token in D. In this process, the token sequence of each sentence can be mapped to a vector sequence, which can be used as the input X of RNN. In addition, the project matrix Φ can be initiated randomly or trained with unsupervised data. In the training process of RNN, Φ is regarded as a parameter matrix and will be updated in the training iterations. We will further discuss this in the experiments.

Output The dimension of the output layer is determined by the presentation schema of result sequence. In the PNO tagging schema, there are three different sentiment statuses of a token (i.e., positive target P, negative target N and others O). Therefore, a three-dimensional vector is used to represent the label value of each token in a sentence. For the label l_j of token t_j,

$$\hat{y}_j = \begin{cases} (0,0,1), & where\ l_j = P, \\ (0,1,0), & where\ l_j = N, \\ (1,0,0), & where\ l_j = O. \end{cases} \tag{31}$$

In this process, each element in $\hat{y}_j$ can be seen as the probability of its related label. For instance, a vector $(0.1, 0.3, 0.6)$ denotes that the label of the corresponding token has the probability of 0.6 to be P, 0.3 to be N, and 0.1 to be O. Because a token sequence and its related label sequence of a sentence are transformed to the input and output layer of RNN models, the token sequence of a sentence can be predicted by trained RNN models.

Train & predict In the training process, let $\theta = \{\Phi, W_*^*, U_*^*, V_*^*, b_*^*\}$ be the set of model parameters. For each θ, an RNN model will calculate an output Y for a sentence S. Each column y_j in the output layer is a three-dimensional vector, which determines the label of t_j in the sentence. Because y_j is the output of the softmax function, the summary of the elements in y_j is 1. We use the cross-entropy error of $\hat{y}_j$ and y_j as the loss of token t_j in sentence S. The whole loss function is the average value of all tokens of all sentences in the dataset. Finally, the models are trained via the Adam optimizer [51]. In the prediction process, RNN models obtain a matrix Y for each sentence, and the maximum value of each column denotes the corresponding label of the related token.

3.6 Postprocessing

Although the sentences we process are all related to the target public figures, there are some other opinion targets in the sentences as well. Theoretically, the sequence labeling step will label all opinion targets in the sentences. In addition, the PNO tagging schema does not define clear boundaries of different targets. Therefore, we must find the targeted public figures and map the tokens to the real targets. For instance, for the sentence "I don't like *Nickname A Nickname B*, I support *Person C*," "*Nickname A*" and "*Nickname B*" are the nicknames of "*Person A*" and "*Person B*," and the targeted public figures are "*Person A*" and *Person B*." Theoretically, the PNO tagging schema will result in two tuples(i.e., ⟨*Nickname A Nickname B*, negative⟩ and ⟨*Person C*, positive⟩). In the post processing step, opinion tuples ⟨*Nickname A*, negative⟩ and ⟨*Nickname B*, negative⟩ are first extracted from ⟨*Nickname A Nickname B*, negative⟩ using the keywords "*Nickname A*" and "*Nickname B*". The unrelated target ⟨*Person C*, positive⟩ is then removed. Finally, the resulting targeted public figures related opinion tuples are mapped to their real names ⟨*Person A*, negative⟩ and ⟨*Person B*, negative⟩.

4 Experiments

4.1 Dataset and evaluation metrics

To evaluate our architecture and methods, we construct a Chinese microblog dataset for the experiments. We first select hot topics in Sina Weibo according to two principles; i.e., (1) the topics have attracted significant attention, and thus we can obtain sufficient data; (2) the topics are controversial. Because many people may be involved in the discussions of these topics, opinion polling on these topics is more meaningful. Five series of hot topics are selected, and 67,033 microblogs and replies to these hot topics are collected. After that, microblogs and replies are segmented into sentences, and sentences without keywords pertaining to targeted public figures or with less than five characters are removed. Finally, 1000 sentences are selected randomly for each series of hot topics, and we label the opinion tuples in the sentences and the whole sentiment polarity of the sentences manually.

Statistics on the labeled data show that some problems may occur when using sentiment scores of sentences to construct opinions on public figures. In the labeled 5000 sentences, 3052 sentences express sentiment toward the targeted public figures, 1412 sentences lack clear sentiment toward the targeted public figures, and 536 sentences contain unrelated data. In the sentences that contain clear sentiments toward public figures, 386 sentences contain different sentiments toward different public figures. In addition, 303 sentences contain positive sentiment toward some public figures, but the overall sentiment is not positive; 263 sentences contain negative sentiment toward some public figures, but the overall sentiment is not negative. That is, more than 18 % of sentences may encounter problems when using sentiment scores of sentences to represent opinions on public figures. Therefore, it is of great significance to extract opinions with fine-grained methods rather than define the sentiment toward targets as the sentiment of the sentence that mentions it.

Theoretically, better performance of sentiment parsing indicates better results in opinion polling on the same dataset. Therefore, considering that there are no acknowledged opinion poll on public figures about their related topics, we evaluate the performance of the opinion poll on public figures indirectly by evaluating the sentiment parsing task. We use F-Score as the evaluation metric. Suppose that RE is the number of correctly extracted opinion tuples about targeted public figures, AE is the number of all extracted opinion tuples about targeted public figures, and AL is the number of all labeled opinion tuples about targeted public figures; the F-Score is then defined as follows:

$$precision = \frac{RE}{AE}, \tag{32}$$

$$recall = \frac{RE}{AL}, \tag{33}$$

$$F - Score = \frac{2 * precision * recall}{precision + recall}. \tag{34}$$

We use the labeled sentences as the dataset, and all sentences are segmented by ICTCLAS.[1] All RNN models are constructed with the python-based sequence processing tool Keras.[2]

4.2 Base line

Because most existing work on microblog-based opinion polling uses sentiment score-based methods to extract opinions in microblogging messages, we take that method as the baseline. Because the sentiment of our dataset is defined as positive or negative, we classify the sentences as positive or negative using a Doc2Vec-based sentiment classification method [52], which is the state of the art in sentiment classification. More specifically, we first train the Doc2Vec model with the Gensim[3] tool on all 67,033 microblogs in the dataset. We then obtain the vectors of the sentences in the training set and build a logistic sentiment classifier. After that, each sentence in the test set is classified as positive or negative, and all targeted public figures mentioned in the sentence are extracted using keywords. Finally, each target constructs an opinion tuple with the sentiment of

[1] http://ictclas.nlpir.org/downloads.

[2] https://github.com/fchollet/keras.

[3] http://radimrehurek.com/gensim/.

Table 2 F-Score of four different word vectors on three basic RNN models

	SRNN	LSTM	GRU
Random	0.647	0.639	0.639
Wiki	0.654	0.646	0.673
Sina	0.667	0.667	0.676
Wiki & Sina	0.663	0.649	0.665

the sentence. The F-Score of the sentiment classification of microblog sentences is 0.693, and the F-Score of opinion extraction is 0.634.

4.3 Word vectors

In this subsection, we explore the performance of different word vectors for the initiation of Φ. As noted in [53], training corpus is one of the most important factors in word embedding, so we mainly explore the effect of different corpuses for training word vectors and aim to select a training scheme for the follow-up experiments. We use the Chinese Wiki corpus and the 67,033 microblogs we collected,called the Sina corpus hereinafter, as the corpus to train word vectors, because the Wiki corpus is a comprehensive open domain corpus, and the Sina corpus is a domain-specific corpus related to the labeled corpus. We take a random initialized Φ as a control scheme, and word vectors trained on the Chinese Wiki corpus and the Sina corpus as the two other schemes. In addition, we add a new training scheme that trains using the Chinese Wiki corpus first and then trains using the Sina corpus, attempting to balance comprehensiveness and domain specificity. The three training schemes are accomplished on the word2vec model [54], which is the most popular word embedding model, with the tool Gensim. After that, the four kinds of word vectors are used on three basic models (i.e., SRNN, LSTM and GRU) to select the most suitable word vector to the follow-up experiments.

Table 2 displays the F-Score of four different word vectors on three basic RNN models. The results show that the random initiated Φ obviously obtains a lower F-Score than the other three pretrained word vectors. In addition, word vectors trained on the Sina corpus gain better performance than those of the Chinese Wiki corpus, which is similar to the result of [53] in which the domain of the training corpus is more important than the size of the corpus. In addition, the Wiki&Sina scheme does not achieve the expected result, which may be caused by the problem that the size of the Sina corpus is far smaller than the Chinese Wiki corpus. As a result, because the Sina corpus achieves the best performance, we will use word vectors trained on the Sina corpus to initiate Φ in the follow-up experiments.

4.4 Different models

In this subsection, we evaluate the performance of different RNN models in the sentiment parsing task on the dataset that we constructed. We explore the 6 basic RNN models (see Sections 3.4.1 and 3.4.2) and their bidirectional models in different depths. Figure 6 displays the results of different models in different versions.

Unidirectional & bidirectional As expected, the bidirectional models gain better performance than their basic models in the overall trend. Most bidirectional models gain higher F-Scores than the basic models at different depths. In addition, the bidirectional models are much more stable than the basic models when the depth changes. This indicates that the opinions expressed in a sentence are not only related to the former tokens of the targets, but also determined by the whole sentence.

Depth From the lines in Fig. 6, it can be seen that the F-Scores of most basic models tend to change smoothly when the depth increases at first and decreases sharply when the depth is larger than 3. Meanwhile, the F-Scores of the bidirectional models all increase at depth 2, then change smoothly at depth 2 to 4, and decrease after that. All the best F-Scores of the basic models are gained at depth 1 or 2, whereas the bidirectional models achieve the best F-Scores in depth 2 to 4.

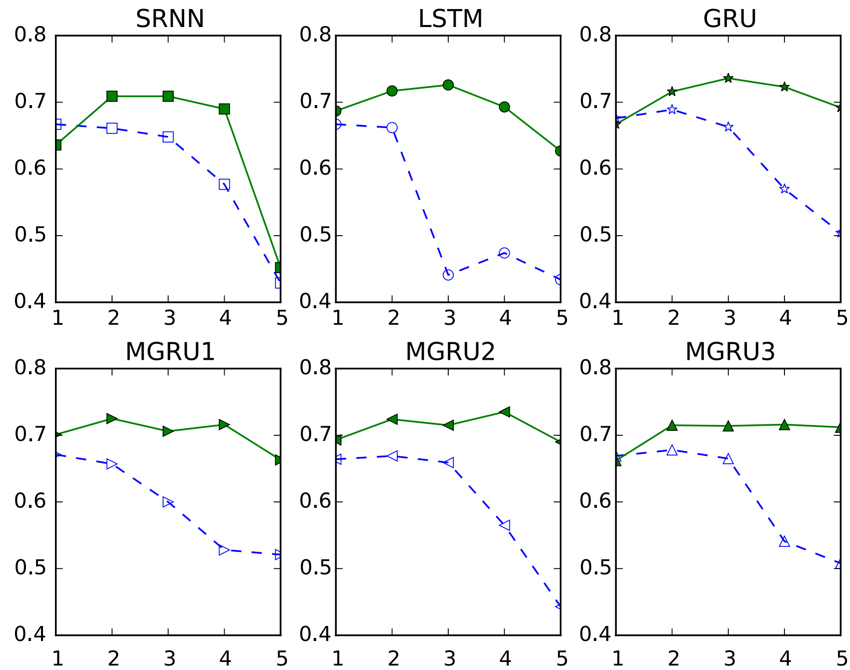
Best performance The best performances of the different models are shown in Table 3. It can be seen that most of the best performances are achieved by GRU, and the best performance of different recurrent units improves when the model become bidirectional and deep. The bidirectional GRU achieves the highest F-Score of 0.736 at depth 3. Finally, it can be seen that most RNN models gain better F-Scores than the base line, as expected, which indicates that it makes more sense to represent and label opinion targets and sentiments jointly in sentiment parsing of public figures in microblogs.

Table 3 Best performances of different models

	F-Score	Basic model	Depth
Shallow+Unidirection	0.675	GRU	–
Shallow+Bidirection	0.701	MGRU1	–
Deep+Unidirection	0.689	GRU	2
Deep+Bidirection	**0.736**	GRU	3
Baseline	0.634	–	–

The bold entry means the F-score is the highest and it is the best result

Fig. 6 The F-Score of different models. Each subfigure is the result of a basic model and its bidirectional version at different depths, where the *dash line* denotes the basic model and the *solid line* denotes its bidirectional model. In each sub figure, the x-axis is the depth and the y-axis is the F-Score



5 Conclusions and future work

In this paper, a sentiment parsing-based architecture is proposed for opinion polling on public figures in microblogs. At first, sentiment parsing is defined as a fine-grained subtask of sentiment analysis at the sentence level. A novel representation of opinions, which label opinion targets and their corresponding sentiments jointly, is then put forward to avoid the mismatching of opinion targets and sentiments. After that, an RNN-based sequence labeling method is used to label the opinion targets and their corresponding sentiments in microblogging sentences. Finally, a fine-grained dataset for sentiment parsing of public figures in hot topics of Sina Weibo is constructed to evaluate the practicality and feasibility of the proposed method.

In the experiments, we take the sentiment score-based opinion poll architecture as a baseline. At first, different training schemes of initial word vectors are compared with the three basic RNN models to select initial word vectors. After that, different RNN models are evaluated in a three-dimensional perspective. The experiments explore the regulation of the performance of different RNN models, and indicate that it makes more sense to represent and label opinion targets and sentiments jointly in sentiment parsing of public figures in microblogs.

Our sentiment parsing method improves the accuracy of opinion extraction by avoiding the mismatching of opinion elements that are extracted separately and thus can better support microblog-based opinion polling on public figures. However, there are still some limits. At first, we do not extract the aspect A of opinion targets and simplified sentiment to be positive or negative and thus cannot meet the demands of some sentiment analysis tasks such as word-of-mouth tracking of products. Second, the PNO tagging scheme cannot represent the structure of opinions, so better tagging schemes must be proposed to represent the structure of opinions when other opinion elements are to be extracted. Finally, the data used in our experiments should be extended by adding more hot topics to evaluate the robustness of the method we proposed.

Future work is targeted at two aspects. First, we plan to explore some other sequence labeling models for sentiment parsing of microblogs. Second, we are interested in how to represent and extract the aspects of targets (i.e., A) together with entity (E) and sentiment (S) in sentiment parsing of microblogs.

Acknowledgments The research is supported by National Natural Science Foundation of China (No.71331008).

References

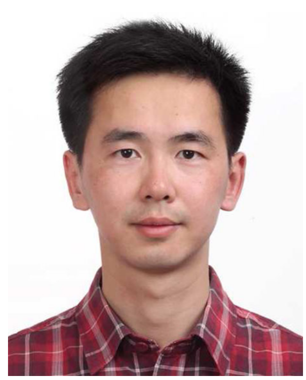
1. Jansen BJ, Zhang M, Sobel K, Chowdury A (2009) Microblogging as online word of mouth branding. In: CHI'09 Extended abstracts on human factors in computing systems. ACM, pp 3859–3864
2. Jansen BJ, Zhang M, Sobel K, Chowdury A (2009) Twitter power: tweets as electronic word of mouth. J Am Soc Inf Sci Technol 60(11):2169–2188

3. O'Connor B, Balasubramanyan R, Routledge BR, Smith NA (2010) From tweets to polls: linking text sentiment to public opinion time series. ICWSM 11(122–129):1–2
4. Tumasjan A, Sprenger TO, Sandner PG, Welpe IM (2010) Predicting elections with twitter: what 140 characters reveal about political sentiment. ICWSM 10:178–185
5. Chung JE, Mustafaraj E (2011) Can collective sentiment expressed on twitter predict political elections? In: AAAI
6. Sang ETK, Bos J (2012) Predicting the 2011 dutch senate election results with twitter. In: Proceedings of the workshop on semantic analysis in social media. Association for Computational Linguistics, pp 53–60
7. Contractor D, Chawda B, Mehta S, Subramaniam LV, Faruquie TA (2015) Tracking political elections on social media: applications and experience. In: Proceedings of the 24th International conference on artificial intelligence. AAAI Press, pp 2320–2326
8. Pascanu R, Gulcehre C, Cho K, Bengio Y (2013) How to construct deep recurrent neural networks. arXiv:1312.6026
9. Graves A, Mohamed A-r, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE International conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 6645–6649
10. Doetsch P, Kozielski M, Ney H (2014) Fast and robust training of recurrent neural networks for offline handwriting recognition. In: 2014 14th International conference on frontiers in handwriting recognition (ICFHR). IEEE, pp 279–284
11. Mikolov T (2012) Statistical language models based on neural networks. Thesis, Brno University of Technology
12. Yazdani M, Henderson J (2015) Incremental recurrent neural network dependency parser with search-based discriminative training. In: Proceedings of the 19th Conference on Computational Language Learning, pp 142–152
13. Metaxas PT, Mustafaraj E, Gayo-Avello D (2011) How (not) to predict elections. In: 2011 IEEE third international conference on privacy, security, risk and trust (PASSAT) and 2011 IEEE third international conference on social computing (SocialCom). IEEE, pp 165–171
14. Kagan V, Stevens A, Subrahmanian VS (2015) Using twitter sentiment to forecast the 2013 pakistani election and the 2014 indian election. IEEE Intell Syst 30(1):2–5
15. Luong TTB, Houston D (2015) Public opinions of light rail service in los angeles, an analysis using twitter data. iConference 2015 Proceedings
16. Liu B (2012) Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies 5(1):1–167
17. Zhang C (2009) Sentiment analysis of chinese documents: from sentence to document level. J Am Soc Inf Sci Technol
18. Bollegala D, Weir D, Carroll J (2013) Cross-domain sentiment classification using a sentiment sensitive thesaurus. IEEE Trans Knowl Data Eng 25(8):1719–1731
19. Pang B, Lee L, Vaithyanathan S (2002) Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 conference on empirical methods in natural language processing, vol 10. Association for Computational Linguistics, pp 79–86
20. Mullen T, Collier N (2004) Sentiment analysis using support vector machines with diverse information sources. In: EMNLP, vol 4, pp 412–418
21. Socher R, Pennington J, Huang EH, Ng AY, Manning CD (2011) Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Proceedings of the conference on empirical methods in natural language processing. Association for Computational Linguistics, pp 151–161
22. Socher R, Perelygin A, Wu JY, Chuang J, Manning CD, Ng AY, Potts C (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the conference on empirical methods in natural language processing (EMNLP), vol 1631. Citeseer, p 1642
23. Bravo-Marquez F, Mendoza M, Poblete B (2013) Combining strengths, emotions and polarities for boosting twitter sentiment analysis. In: Proceedings of the second international workshop on issues of sentiment discovery and opinion mining. ACM, p 2
24. Hu X, Tang J, Gao H, Liu H (2013) Unsupervised sentiment analysis with emotional signals. In: Proceedings of the 22nd international conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp 607–618
25. Yi J, Nasukawa T, Bunescu R, Niblack W (2003) Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques. In: Third IEEE international conference on data mining, 2003. ICDM 2003. IEEE, pp 427–434
26. Popescu A-M, Etzioni O (2007) Extracting product features and opinions from reviews. In: Natural language processing and text mining. Springer, pp 9–28
27. Li F, Han C, Huang M, Zhu X, Xia Y-J, Zhang S, Hao Y (2010) Structure-aware review mining and summarization. In: Proceedings of the 23rd international conference on computational linguistics. Association for Computational Linguistics, pp 653–661
28. Irsoy O, Cardie C (2014) Opinion mining with deep recurrent neural networks. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 720–728
29. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
30. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Li F-F (2014) Large-scale video classification with convolutional neural networks. In: 2014 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, pp 1725–1732
31. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. J Mach Learn Res 12:2493–2537
32. Le QV, Mikolov T (2014) Distributed representations of sentences and documents. arXiv:1405.4053
33. d'Avila Garcez AS, Broda K, Gabbay DM (2001) Symbolic knowledge extraction from trained neural networks: a sound approach. Artif Intell 125(1):155–207
34. Rosaci D (2007) Cilios: connectionist inductive learning and inter-ontology similarities for recommending information agents. Inf Syst 32(6):793–825
35. Chen D, Socher R, Manning CD, Ng AY (2013) Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. arXiv:1301.3618
36. d'Avila Garcez AS, Gabbay DM, Ray O, Woods J (2007) Abductive reasoning in neural-symbolic systems. Topoi 26(1):37–49
37. Socher R, Chen D, Manning CD, Ng A (2013) Reasoning with neural tensor networks for knowledge base completion. In: Advances in neural information processing systems, pp 926–934
38. Mesnil G, He X, Li D, Bengio Y (2013) Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding, pp 3771–3775
39. Mesnil G, Dauphin Y, Yao K, Bengio Y, Deng L, Hakkani-Tur D, He X, Heck L, Tur G, Yu D (2015) Using recurrent neural networks for slot filling in spoken language understanding. IEEE/ACM Trans Audio Speech Lang Process 23(3):530–539

40. Liwicki M, Graves A, Bunke H (2012) Neural networks for handwriting recognition. In: Computational intelligence paradigms in advanced pattern classification. Springer, pp 5–24
41. Legrand J, Collobert R (2014) Joint rnn-based greedy parsing and word composition. arXiv:1412.7028
42. Li L, Jin L, Jiang Z, Song D, Huang D (2015) Biomedical named entity recognition based on extended recurrent neural networks. In: 2015 IEEE International Conference on bioinformatics and biomedicine (BIBM), pp 649–652
43. Chiu JPC, Nichols E (2015) Named entity recognition with bidirectional lstm-cnns. arXiv:1511.08308
44. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
45. Graves A (2012) Supervised sequence labelling with recurrent neural networks, vol 385. Springer
46. Cho K, Van Merrinboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:1406.1078
47. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555
48. Jozefowicz R, Zaremba W, Sutskever I (2015) An empirical exploration of recurrent network architectures. In: Proceedings of the 32nd international conference on machine learning (ICML), pp 2342–2350
49. Schuster M, Paliwal KK (1997) Bidirectional recurrent neural networks. IEEE Trans Signal Process 45(11):2673–2681
50. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
51. Kingma D, Adam JB (2014) A method for stochastic optimization. arXiv:1412.6980
52. Le QV, Mikolov T (2014) Distributed representations of sentences and documents. arXiv:1405.4053
53. Lai S, Liu K, Xu L, Zhao J (2015) How to generate a good word embedding? arXiv:1507.05523
54. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. arXiv:1310.4546

Jiajun Cheng received his BS degree and MD degree in Management Science and Engineering from National University of Defense Technology (China), in 2012 and 2014, respectively. He is currently a PhD candidate at College of Information Systems and Management, National University of Defense Technology, Changsha, China. His research interests include sentiment analysis, text mining and artificial nueral network.

Xin Zhang received his BS and PhD degrees in system engineering from National University of Defense Technology (China), in 2000 and 2006, respectively. He is currently an associate professor with the Research Center of Computational Experiments and Parallel System Technology, College of Information Systems and Management, National University of Defense Technology. His research interests include text-based event extraction and sentiment analysis, object recognition in images, and machine learning techniques.



Pei Li received his BS degree and PhD degree in Computer Science from University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively. He is currently a lecturer at College of Information Systems and Management, National University of Defense Technology, Changsha, China. His research interests include social networks and wireless networks.

Sheng Zhang received his BS degree in system engineering from National University of Defense Technology (China), in 2015. He is currently a master degree candidate at College of Information Systems and Management, National University of Defense Technology, Changsha, China. His research interests include social networks and data mining.

Zhaoyun Ding received his PhD degree in Computer Science from National University of Defense Technology (China), in 2013. He is currently a lecturer with the Research Center of Computational Experiments and Parallel System Technology, College of Information Systems and Management, National University of Defense Technology. His research interests include data mining and social computing.

Hui Wang received his PhD degree in system engineering from National University of Defense Technology (China), in 2005. He is currently a professor with the Research Center of Computational Experiments and Parallel System Technology, College of Information Systems and Management, National University of Defense Technology. His research interests include social networking and data mining.