Front. Comput. Sci.
https://doi.org/10.1007/s11704-018-7150-9

RESEARCH ARTICLE

Stance detection via sentiment information and neural network model

Qingying SUN[1,2], **Zhongqing WANG**[1], **Shoushan LI**[1], **Qiaoming ZHU** (✉)[1], **Guodong ZHOU**[1]

1 Natural Language Processing Lab, Soochow University, Suzhou 215006, China
2 Huaiyin Normal University, Huai'an 223300, China

Abstract Stance detection aims to automatically determine whether the author is in favor of or against a given target. In principle, the sentiment information of a post highly influences the stance. In this study, we aim to leverage the sentiment information of a post to improve the performance of stance detection. However, conventional discrete models with sentimental features can cause error propagation. We thus propose a joint neural network model to predict the stance and sentiment of a post simultaneously, because the neural network model can learn both representation and interaction between the stance and sentiment collectively. Specifically, we first learn a deep shared representation between stance and sentiment information, and then use a neural stacking model to leverage sentimental information for the stance detection task. Empirical studies demonstrate the effectiveness of our proposed joint neural model.

Keywords natural language processing, machine learning, stance detection

1 Introduction

The aim of stance detection is to automatically determine from the text whether the author is in favor of, against, or neutral toward a given target [1]. People tend to express their stance toward targets through posts on social media, such as Facebook and Twitter. The target can be an individual, an organization, an event, a movement, or a policy. Automatically detecting the stance can be beneficial in understanding public opinion and identifying electoral issues. Previously, there exist many studies focusing on stance detection. Various linguistic and structural features have been employed to detect stances [2–6]. In addition, sentiment information plays an important role in stance detection [4,7,8]. Previous studies take sentiment information as features for stance detection owing to the assumption that sentiment information of a post highly influences the stance. Take the following instances as examples. The stance and polarity of E1 and E2 are consistent. However, different objectives cause diversity between stance and polarity. For example in E3, the negative opinion on damaging the environment indicates that we should be concerned with climate change. In E4, the poster praises and appreciates God to oppose atheism.

E1: Target: Feminist Movement

Tweet: *@rimmedlarry Actually, the tag was made by feminists so they can narcissistically post selfies to prove they're not ugly.*

Stance: *Against* Sentiment: *Negative*

E2: Target: Climate Change is a Real Concern

Tweet: *SO EXCITING! Meaningful climate change action is on the way!*

Stance: *Favor* Sentiment: *Positive*

E3: Target: Climate Change is a Real Concern

Tweet: *When the last tree is cut down, the last fish eaten & the last stream poisoned, you will realize that you cannot eat money.*

Stance: *Favor* Sentiment: *Negative*

E4: Target: Atheism

Received April 28, 2017; accepted September 26, 2017

E-mail: qmzhu@suda.edu.cn, qingying.sun@foxmail.com

Tweet: *dear lord thank u for all of ur blessings forgive my sins lord give me strength and energy for this busy day ahead.*

Stance: *Against* Sentiment: *Positive*

From the above examples, we can find that stances are expressed through different ways with different targets. Hence, it is very difficult to handle the interaction between sentiment and stance information using a feature-based discrete model.

To address the above challenges, we propose a novel neural network model to integrate stance and sentiment detection jointly. The advantages are that we can learn stance and sentiment information collectively, and stance and sentiment information can be influenced by each other. In particular, we learn the shared representation between stance and sentiment information and propose a neural stacking framework to incorporate sentiment information into the stance detection model. The experimental results show that the proposed joint learning model can effectively detect stance and leverage the sentiment information.

2 Related work

2.1 Stance detection

Early studies on stance detection mainly focus on congressional debates [9–11], company internal discussions [12], and online debate in public forums [2–5,13]. In those studies, various linguistic features such as morphology, syntactic, and discourse features, are used to predict the stance. In recent years, with the development of social media, more studies focus on detecting the political or rumor stance from tweets [14–17]. Different from the studies on online debate, which can obtain many automatically annotated data, annotated tweets for stance detection are limited. Hence, people tend to use some unsupervised or semi-supervised methods for detecting stance on tweets. For example, Rajadesingan and Liu proposed a semi-supervised framework to use a retweet-based label propagation algorithm for identifying users with differing opinions [18]. Johnson and Goldwasser proposed a weakly supervised method for understanding the stances held by politicians [14].

SemEval-2016 Task 6 is an important share task for stance classification [1,19]. In the share task, the best system employed a recurrent neural network (RNN) initialized with features learned via distant supervision on two large unlabeled datasets [20]. The second best system employed a convolutional neural network (CNN) and designed a voting scheme for prediction [21]. However, different from the above neural network models, the organizer proposed a support vector machine (SVM) model using only linguistic features, and it outperformed the best system among the participating systems [1]. In this study, we evaluate our approach using this SemEval-2016 Task 6 Twitter Stance Detection corpus. In contrast to previous research, we utilize neither complex linguistic features nor other additional resources. We detect the stance of each post through the basic textual information and the interaction with sentiments. The experimental results show that the proposed model can outperform the above works.

2.2 Stance classification with sentiment information

Sentiment analysis has grown to be one of the most active research areas in natural language processing (NLP) [22–27]. Sentiment information is very helpful for stance classification. In this study, we leverage the sentiment information of tweets to facilitate stance detection. In previous work, a straight forward approach is to use sentimental information as a feature for stance detection [4,7,28]. For example, Anand et al. simply took the positive and negative emotion words as features in rule-based JRip classifier [4]. Sun et al. combined the bigram/trigram words, part of speech, and the polarity of a post to generate features, and the experimental results showed a high improvement [7]. Sobhani et al. conducted several experiments to tease out the interaction of stance with sentiment and they proposed an oracle system that simply mapped sentiment gold labels to stance labels, and the experimental results showed that sentiment information was useful for stance classification, but it alone was not sufficient [28]. They also added the sentiment lexicon features into an SVM model, but the experimental results did not outperform the SVM *n*-gram baseline. Although sentiment information is helpful for stance detection, how to use it to facilitate stance detection effectively remains difficult. Therefore, in this study, we propose a joint neural model to utilize sentiment information to facilitate stance detection for the SemEval-2016 Task 6 A, and experimental results show that our approach is effective.

2.3 Joint neural model

In the last decade, many research has been using discrete models to solve related NLP tasks jointly. Titov et al. constructed a joint statistical model of text and sentiment ratings to obtain aspect-based sentiment summarization [29]. Watanabe et al. proposed a structured model for joint learning of argument roles and predicate senses [30]. Simova et al. presented several approaches toward constructing joint ensemble models for Bulgarian morphosyntactic tagging and dependency parsing [31]. In these joint models, a key task is to capture mutual information between the integrated tasks to form manual feature templates. For our problem, the stance and sentiment are semantically related, but it is rather difficult to extract non-local features that express the shared information through manual feature templates. In contrast, neural network models have been shown effective for inducing sentence-level semantic features automatically [32]. It makes them a useful tool for our joint model.

The neural network was used to realize multi-task learning mainly in two aspects. One aspect is to obtain a mutual semantic representation via parameter sharing between different tasks. For example, Collobert et al. used a single learning system that can discover adequate internal representations of multiple tagging tasks [33]. Liu et al. proposed a novel CNN embedded multi-task learning system to synthesize these tasks by learning both unique and shared representations for each task [34]. Another aspect is to use stacking, feeding the hidden layer output of one task as additional input of another task for joint learning. For example, Zhou et al. proposed a neural stacking model for semantic role labeling [35]. Chen et al. built a neural network counterpart to discrete stacking and multi-view learning for multiple tag sequence tasks [36]. Different from the above studies, we focus on the main task of stance detection with the help of an auxiliary task of sentiment analysis. We consider both shared representation and stacked learning.

In this study, we present a neural network model for integrating stance detection and sentiment analysis. To our knowledge, this is the first work to employ neural networks for collaborative stance detection and sentiment analysis.

3 Our approach

In this section, we propose our joint learning approach for stance detection with sentiment classification. Figure 1 shows the overall structure of the joint learning model. We consider stance detection as the main task, and sentiment classification as the auxiliary task. This approach leverages the auxiliary representation to assist the performance of the main task. One idea of our approach lies in that the auxiliary long short-term memory network (LSTM) layer is shared by both the main and auxiliary tasks so as to obtain a mutual semantic representation via parameter sharing. Another idea is to put the shared representation information to stack, feeding the hidden layer output of the auxiliary task as additional input of the main task for joint learning. Every tweet in our dataset is annotated with stance and sentiment information simultaneously.

In the following subsections, we first introduce the basic LSTM network.Then, we describe how to represent a tweet and how to use the basic LSTM network to classify stance and sentiment. Finally, we present the joint learning approach to perform stance detection with the help of sentiment analysis.

3.1 Basic LSTM network

LSTM is a special type of RNN. It is introduced by Hochreiter and Schmidhuber and is capable of learning long-term dependencies [37]. We use this network to obtain the document representation of each post. While there are different variations of LSTMs, we present here the one adopted by Graves [38]. Specifically, let $X = (x_1, x_2, \ldots, x_N)$ denote an input sequence, where $x_t \in R^l$ $(1 \leqslant t \leqslant N)$. At each position t, there is a set of internal vectors, including an input gate i_t, a forget gate f_t, an output gate o_t, and a memory cell c_t. All these vectors are used together to generate a d-dimensional hidden state h_t as follows (Eqs. (1)–(6)):

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \tag{1}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \tag{2}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \tag{3}$$

$$g_t = \tanh(W_g x_t + U_g h_{t-1} + b_g), \tag{4}$$

$$c_t = i_t \otimes g_t + f_t \otimes c_{t-1}, \tag{5}$$

$$h_t = o_t \otimes \tanh(c_t), \tag{6}$$

where $\otimes$ is the element-wise multiplication of two vectors, σ is the sigmoid function, and $W_* \in R^{d\times l}$, $U_* \in R^{d\times d}$, and $b_* \in R^d$ are weight matrices and vectors to be learned. The LSTM model is used to represent the tweets for both stance and sentiment classification.

3.2 Tweet representation

In this study, we turn each tweet into a word unigram and character n-grams (2–5) stream and then use bag-of-words features as the LSTM network's input representation [39]. The detailed processing procedure is as follows:

Given a set of tweets $T = \{t_1, t_2, \ldots, t_n\}$, each tweet is represented as a sequence of words $t_i = \{w_1, w_2, \ldots, w_l\}$ $(1 \leqslant i \leqslant n)$, and each word is represented as a sequence of characters $w_j = \{c_1, c_2, \ldots, c_m\}$ $(1 \leqslant j \leqslant l)$. For getting the tweet representation, we obtain the word unigram and character n-grams (2–5) representation as follows:

$$\mathrm{W}^{unigram} = \{w_1, w_2, \ldots, w_l\}, \tag{7}$$

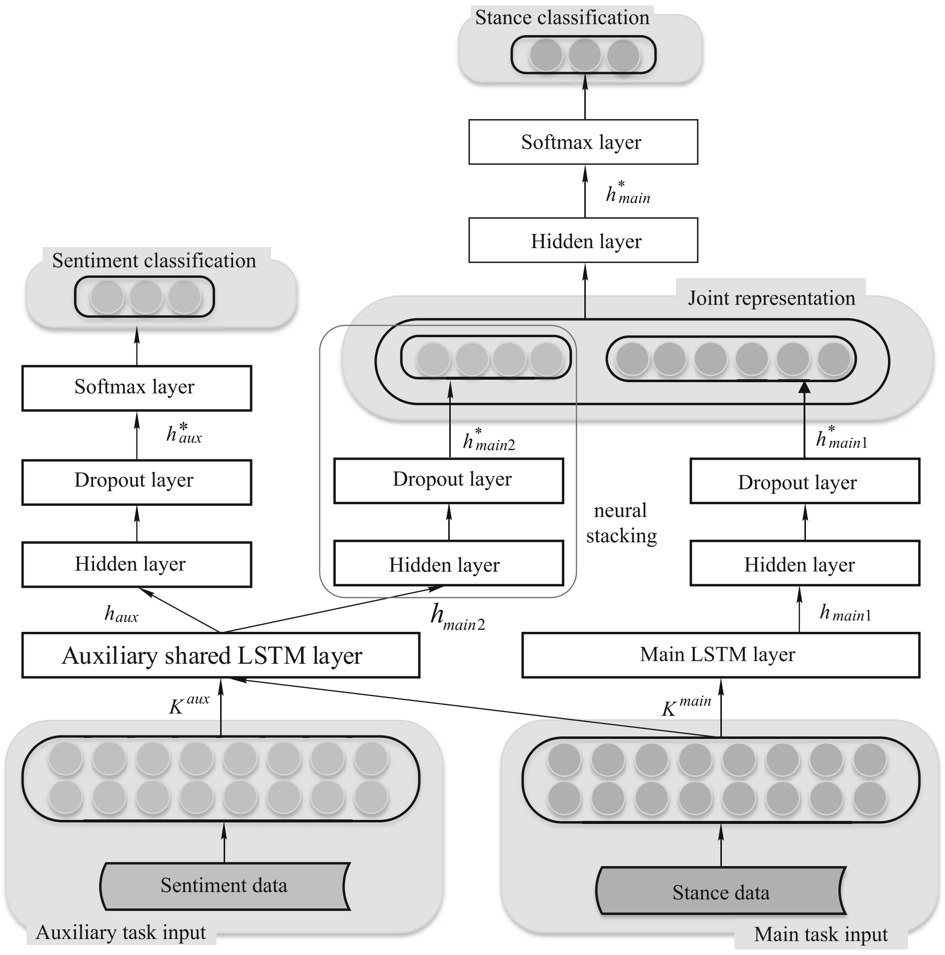


Fig. 1 Overall architecture of joint neural model

$$C^{ngrams} = \sum_{j=1}^{l}\{\{\sum_{r=1}^{m-1} c_{j,r} \oplus c_{j,r+1}\} + \{\sum_{r=1}^{m-2} c_{j,r} \oplus c_{j,r+1} \oplus c_{j,r+2}\}$$
$$+\{\sum_{r=1}^{m-3} c_{j,r} \oplus c_{j,r+1} \oplus c_{j,r+2} \oplus c_{j,r+3}\}$$
$$+\{\sum_{r=1}^{m-4} c_{j,r} \oplus c_{j,r+1} \oplus c_{j,r+2} \oplus c_{j,r+3} \oplus c_{j,r+4}\}\}, \quad (8)$$

where $W^{unigram}$ denotes the word unigram set, l the number of words in a tweet, C^{ngrams} the character n-grams (2–5) set, $\oplus$ the concatenate operator, m the number of characters in a word, and $c_{j,r}$ the rth character of the jth word. For each tweet, we combine the $W^{unigram}$ set and C^{ngrams} set to represent it.

Subsequently, we utilize all the tokens in $W^{unigram}$ and $C^{ngrams}$of the whole train set to create a dictionary. For each tweet, we use each token's index in the dictionary to form an index vector representation K as follows:

$$K = Dic(W^{unigram} + C^{ngrams}). \quad (9)$$

3.3 LSTM model for stance classification and sentiment classification

We first show the LSTM model for stance and sentiment classification individually, as shown in Fig. 2. In this architecture, we utilize K as the input vector and feed K into the LSTM layer to generalize a new representation h, i.e.,

$$h = LSTM(K). \quad (10)$$

Next, we feed h into a fully connected layer to obtain the representation h_c as follows:

$$h_c = dense(h) = \sigma(\theta^T h + b_c), \quad (11)$$

where σ is a non-linear activation function, and we use a rectified linear unit. θ^T denotes the weight to be learned and b_c is a bias term. h_c is the output vector of a fully connected layer.

Subsequently, a dropout layer is applied. Dropout is a regularization technique for reducing overfitting in neural networks by preventing complex co-adaptations on training data. It has been very successful in improving the performance of neural networks [40]. The computing function is given as follows:

$$h_d = h_c \cdot D(p^*), \tag{12}$$

where D denotes the dropout operation and p^* is the random selection probability.

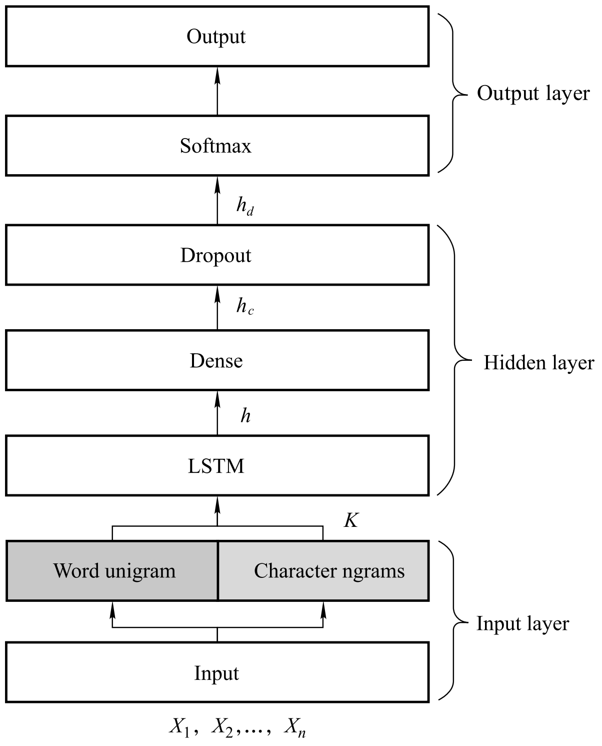


Fig. 2 LSTM-based classification model

Finally, a softmax layer is applied to obtain a categorical probability distribution. It works as follows:

$$p = soft\max(w_s h_d + b_s), \tag{13}$$

where p denotes the predicted categorical probability, w_s is the weight to be learned, and b_s is a bias term to be learned.

We employ a cross-entropy loss function to train the model. In particular, the loss function is defined as follows:

$$loss_C = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij}\log p_{ij}, \tag{14}$$

where $loss_C$ denotes the loss function of stance or sentiment classification, n is the total number of items of training data, m is the number of categories, y_{ij} indicates whether the ith sample belongs to the jth category, which is the ground-truth label, and p_{ij} represents the predicted probability.

3.4 Joint learning for stance detection

In this subsection, we will introduce how to consider the interaction between the stance and sentiment classification based on the proposed joint model.

1) Stance classification task

The joint representation of this task's input is composed of two parts shown in Fig. 1. One part is generated from the main LSTM layer, and the other is generated from the auxiliary shared LSTM layer.

$$h_{main1} = LSTM_{main}(K^{main}), \tag{15}$$

$$h_{main2} = LSTM_{aux}(K^{main}), \tag{16}$$

where h_{main1} denotes the output vector of the main task's LSTM layer, and K^{main} is the tweet representation set with stance information. On the other hand, h_{main2} is the output vector of the auxiliary task's LSTM layer, and the input data is the same as h_{main1}. Note that the auxiliary LSTM layer is a shared LSTM layer, which bridges the stance classification and sentiment classification to obtain a mutual semantic representation via parameter sharing.

Subsequently, we feed h_{main1} and h_{main2} into a fully connected layer followed by a dropout layer respectively to obtain the main representation h^*_{main1} and auxiliary representation h^*_{main2}, i.e.,

$$h^*_{main1} = dense(h_{main1}) \cdot D(p^*), \tag{17}$$

$$h^*_{main2} = dense(h_{main2}) \cdot D(p^*). \tag{18}$$

We concatenate the representation h^*_{main1} and h^*_{main2} to form a new joint representation, and feed it into a fully connected layer to obtain an output vector h^*_{main}:

$$h^*_{main} = dense(h^*_{main1} \oplus h^*_{main2}), \tag{19}$$

where $\oplus$ denotes the concatenate operator.

2) Sentiment classification task

In this auxiliary task, the tweet representation set K^{aux} with sentiment information is fed into the shared auxiliary LSTM layer to obtain an output vector h_{aux} as follows:

$$h_{aux} = LSTM_{aux}(K^{aux}). \tag{20}$$

Subsequently, a fully connected layer followed by a dropout layer and a sigmoid layer is employed to obtain the sentiment classification results. They are the same as those in the standard LSTM model, which have been elaborated in Section 3.3.

3.5 Training

In the main task, we have obtained the representation h^*_{main}. Subsequently, we feed it into a softmax layer to predict the probability of stance classification.

We employ a cross-entropy loss function,which has been introduced in section 3.3, to train the main task and auxiliary task models at the same time. We assign different weights to different tasks, which is a linear combination of the main task and auxiliary task as follows:

$$loss_C^{joint-LSTM} = \lambda \cdot loss_C^{main} + (1-\lambda) loss_C^{aux}, \tag{21}$$

where λ is the weight parameter between the main task and auxiliary task, $loss_C^{main}$ is the loss function of the main task, $loss_C^{aux}$ is the loss function of the auxiliary task, and $loss_C^{joint-LSTM}$ is the loss function of the joint model.

We take Rmsprop [41], which is an unpublished, adaptive learning rate method proposed by Tieleman and Hinton in their Coursera class as the optimization algorithm to tune the parameters. All the matrix and vector parameters are initialized with uniform distribution in $\left[-\sqrt{6/(r+c)}, \sqrt{6/(r+c)}\right]$, where r and c are the numbers of rows and columns in the matrices respectively [42].

4 Experiments

4.1 Dataset and statistics

In this study, the dataset of SemEval-2016 Task 6 A is used for experiments. Each tweet corresponds to a special target: "Atheism", "Climate Change is a Real Concern", "Feminist Movement", "Hillary Clinton", and "Legalization of Abortion". In [8], they further annotate the polarity label for each tweet. There are 2,914 tweets on the training set, and the remaining 1,249 tweets are for the testing dataset. Table 1 presents the distribution of the dataset.

To measure the correlations between stance and sentiment, Table 2 provides the distribution of the matching instances between stance and sentiment classification. From Table 2, we can find that negative opinion causes people to *against* the target in most domains. However, people always praise God to *against* atheism.

4.2 Experiment settings

• **Hyper-parameters** We utilize the 5-fold cross-validation on training data to tune the parameters. Specifically, the size of the hidden layers and the output dimension of LSTM layers are set as 128, the batch size as 32, and the epoch size as 30. We split the dataset into five subsets according to the five targets and train a separate classifier for each of these targets. In the joint model, we adjust the parameters of Rmsprop optimizer and the weight of cross-entropy loss between the main task and auxiliary task according to different targets.

Table 1 Distribution of instances in the stance train and test sets

Target	#Train	Instances in Train/%			#Test	Instances in Test/%		
		Favor	Against	None		Favor	Against	None
Atheism	513	17.9	59.3	22.8	220	14.5	72.7	12.7
Climate change is concern	395	53.7	3.8	42.5	169	72.8	6.5	20.7
Feminist movement	664	31.6	49.4	19.0	285	20.4	64.2	15.4
Hillary clinton	689	17.1	57.0	25.8	295	15.3	58.3	26.4
Legalization of abortion	653	18.5	54.4	27.1	280	16.4	67.5	16.1
Total	2914	25.8	47.9	26.3	1249	23.1	51.8	25.1

Table 2 Distribution of matching instances between stance and sentiment classification in the dataset

Sentiment	Atheism			Climate change is concern		
Stance	positive/%	negative/%	neither/%	positive/%	negative/%	neither/%
favor	2.6	13.0	1.4	18.8	**29.4**	11.2
against	**48.6**	12.7	2.0	0.0	4.6	0.0
none	8.9	9.5	1.4	12.2	16.1	7.6
Sentiment	Feminist movement			Hillary clinton		
Stance	positive/%	negative/%	neither/%	positive/%	negative/%	neither/%
favor	9.0	18.7	0.6	12.4	3.6	0.6
against	4.5	**47.8**	1.5	9.1	**47.2**	1.1
none	4.8	10.4	2.6	8.6	15.1	2.2
Sentiment	Legalization of abortion					
Stance	positive/%	negative/%	neither/%			
favor	4.5	12.6	0.8			
against	14.7	**40.9**	2.7			
none	7.1	14.4	2.4			

Table 3 presents the hyper-parameters.

• **Evaluation metrics** We adopt a macro-average F score to evaluate the performance of each target. For stance classification, the macro-average F score is computed for the two main classes as follows:

$$F_{avg} = \frac{(F_{favor} + F_{against})}{2}, \tag{22}$$

where F_{favor} and $F_{against}$ are calculated as

$$F_{favor} = \frac{2P_{favor}R_{favor}}{P_{favor} + R_{favor}}, \tag{23}$$

$$F_{against} = \frac{2P_{against}R_{against}}{P_{against} + R_{against}}. \tag{24}$$

Table 3 Parameters in joint model according to different targets

Target	lr	λ	$1-\lambda$
Atheism	0.001	0.65	0.35
Climate change is concern	0.002	0.75	0.25
Feminist movement	0.002	0.65	0.35
Hillary clinton	0.002	0.65	0.35
Legalization of abortion	0.0001	0.9	0.1

Note: lr denotes the learning rate, λ the weight of main task's cross-entropy loss, and 1-λ the weight of auxiliary task's cross-entropy loss

We average the F_{avg} calculated for each target separately to obtain the macro-average F score for all targets ($MacF_{avg}$ for short). Moreover, we also evaluate the F_{avg} score across all targets, which is defined as the micro-average F score ($MicF_{avg}$ for short). Systems that perform relatively better on the more frequent target classes will obtain a higher micro-average F score. On the other hand, to obtain a high macro-average F score, a system has to perform well on all target classes.

4.3 Comparison with baselines

We first compare the proposed joint model with the best (RNN) and second best system (CNN) in the SemEval-2016 Task 6 A, and with conventional discrete models such as SVM, Maximum Entropy (ME), and Naïve Bayes (NB):

- **SVM-ngram** This means using word n-gram (1–3) and character n-gram (2–5) features to train the SVM classifier. This model is proposed by the organizer [8].
- **ME** This is the maximum entropy classifier, which also uses word unigram and character n-gram (2–5) as features.
- **NB** This is the Bayesian classifier, and the setting of features is the same as ME.
- **MTTRE(RNN)** This is the best system in SemEval-2016 Task 6 A among 19 participating teams [20]. The approach employed two RNN classifiers: the first was trained to predict task-relevant hashtags on a very large unlabeled Twitter corpus, and this network was used to initialize the second RNN classifier, which was trained with the provided Task A data.
- **Pkudblab(CNN)** This is the second best system in the SemEval-2016 Task 6 A [21]. The team developed a CNN and designed a voting scheme for prediction, which appointed the most frequently appearing label as the result across all epochs. They used pre-trained word embedding published by word2vec team [43] as the input vector of the neural network.

Table 4 presents the experimental results between the above models and the joint model. By comparing these results, we can obtain the following findings:

1) Comparing with the three conventional discrete models, the SVM-ngram model obtains the best performance, which shows the advantage of the SVM model in stance classification.

2) The neural network model (**MTTRE** and **pkudblab**) results do not surpass the discrete SVM-ngram and ME model. It may be because the small number of samples limits the effectiveness of the neural network model.

3) The joint neural model beats the best system (**MTTRE**) in SemEval-2016 Task 6 A with 1.4% and 4.13% improvement in Micro F1 and Macro F1 scores respectively. This indicates that joint learning with sentiment information can provide great benefit for stance classification. Moreover, the joint neural model also improves the **SVM-ngram** model by 2.15% in Macro F1 scores. It indicates that the proposed joint neural network model is effective for inducing shared sentence-level semantic features between stance and sentiment automatically than other models.

4.4 Effectiveness of sentiment information

We then evaluate the effect of sentiment information in Table 5 with the following methods, and the first four models are proposed by the organizer [8].

- **SVM-unigram** means the model that only uses word unigram features to train the SVM model.
- **SVM-ngram** means the model that uses word n-gram (1–3) and character n-gram (2–5) features to train the SVM model.

- **SVM-sentiment** means the model that is based on the SVM-ngram model and adds sentiment features drawn from sentiment lexicons.
- **Oracle Sentiment** means the oracle system that simply maps sentiment gold labels to stance labels (mapping all positive instances to *favor* and all negative instances to *against*). This benchmark is informative because the gold sentiment labels are used to obtain an upper bound of the F-score in the oracle system.
- **LSTM-unigram** means using the basic LSTM model that is elaborated in Section 3.3 to predict the stance of each tweet individually. Particularly, the input features are only word unigrams.
- **LSTM-ngram** is the same as the LSTM-unigram model and one difference is the use of word unigram and character n-gram (2–5) as input features.

Table 4 Comparison with other systems

Model	Atheism	Climate	Feminism	Hillary	Abortion	$MacF_{avg}$	$MicF_{avg}$
	F_{avg}	F_{avg}	F_{avg}	F_{avg}	F_{avg}		
SVM-ngram	65.19	42.35	57.46	58.63	**66.42**	58.01	68.98
ME	64.04	43.08	57.40	60.86	63.43	57.76	68.92
NB	56.60	42.90	55.28	45.34	58.17	51.66	67.43
MTTRE (RNN)	61.47	41.63	**62.09**	57.67	57.28	56.03	67.82
Pkudblab (CNN)	63.34	**52.69**	51.33	**64.41**	61.09	58.57	67.33
Joint	**66.78**	50.60	59.35	62.47	61.58	**60.16**	**69.22**

Table 5 Comparison between basic models and models adding sentiment information

Model	Atheism	Climate	Feminism	Hillary	Abortion	$MacF_{avg}$	$MicF_{avg}$
	F_{avg}	F_{avg}	F_{avg}	F_{avg}	F_{avg}		
SVM-unigram	53.25	38.39	55.65	57.02	60.09	52.88	63.31
SVM-ngram	65.19	42.35	57.46	58.63	**66.42**	58.01	68.98
SVM-sentiment	65.20	40.10	54.50	60.60	61.70	56.40	66.80
Oracle Sentiment	65.80	34.30	**61.70**	62.20	41.30	53.10	57.20
LSTM-unigram	58.81	38.20	49.06	50.94	62.00	51.80	62.00
LSTM-ngram	63.25	41.80	56.29	57.82	61.80	56.20	67.17
Joint	**66.78**	**50.60**	59.35	**62.47**	61.58	**60.16**	**69.22**

Table 5 presents the experimental results of basic models (SVM-unigram, SVM-ngram, LSTM-unigram, and LSTM-ngram) and the models adding sentiment information (SVM-sentiment, Oracle Sentiment, and Joint) for all targets respectively. By comparing these results, we can obtain the following findings:

1) Comparing the **SVM-unigram** with **SVM-ngram** results, we can observe that using word and character n-gram features can outperform the basic word unigram features significantly in all targets. Considering that tweet texts are expressed in informal language, message-length constraints, and spelling errors, the character level n-gram representation can capture word morphology and reduce out-of-vocabulary words. For example, *pres* and *pic* are the abbreviated forms of *president* and *picture* respectively. We can get the same character level n-gram representation such as *pr*, *pre*, and *pres*.

2) Comparing the **LSTM-unigram** with **LSTM-ngram** results, we can observe that in the neural network model, using word and character n-gram features can also outperform the basic word unigram features significantly in all targets.

3) The **Oracle Sentiment** system obtains results that are not bad on all targets. This indicates that even though sentiment can play a key role in detecting stance, sentiment alone is not sufficient.

4) Adding sentiment features drawn from sentiment lexicons on the basis of SVM-ngram model leads to worse performance. This indicates that leveraging sentiment information to facilitate detecting stance in this simple form is not effective. Because the polarity of a post would be the same or opposite toward the stance, the drawback of this method is that such pipeline model would cause cascading of errors.

5) The **joint** model improves the performance of the **LSTM-ngram** model. This indicates that joint learning with sentiment information can provide great benefit for stance classification in the neural network model. Moreover, comparing the effect of sentiment information usage in SVM-sentiment, this form we propose is more effective.

We then analyze the effects of different sizes of sentiment samples on stance classification. We consider that the size of sentiment samples is from 0 to 600, increasing by 100 each time, employing the over sampling technique if the sample is less than 600. Meanwhile, we take the LSTM model and Micro F1 score as the reference model and metric respectively. Figure 3 shows the performance change.

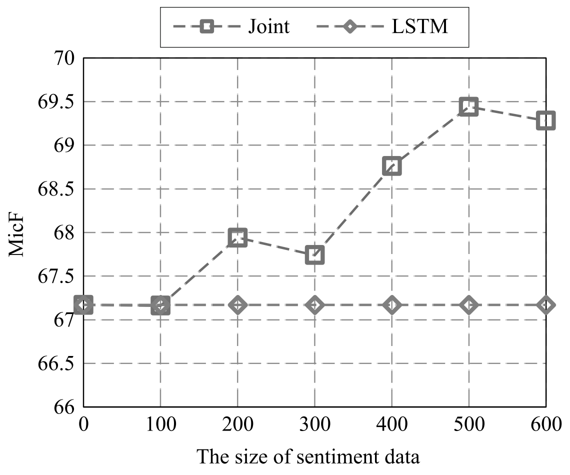


Fig. 3 Performance comparisons of using different size of sentiment samples in joint model

From Fig. 3, we can observe that 1) the contribution is limited, when the size of sentiment samples is small (<100), and 2) the sentiment information becomes more effective as the number of sentiment samples is increasing (>200).

4.5 Influence of neural network structure

In our joint model, the sentiment information is integrated in two aspects: one is forming a mutual semantic representation of stance and sentiment via parameter sharing, and the other is putting the output of prefix as one of the input of suffix to build a neural stacking framework. We analyze the effects of these two factors in stance detection respectively. **LSTM** means using the basic LSTM model to predict the stance of each tweet individually. **LSTM_SP** is the joint LSTM model with shared representation with stance and sentiment information, and **LSTM_SP_ST** is the proposed model with both shared representation and neural stacking. Table 6 provides the performance of these models on all targets.

Table 6 Performance comparisons of different factor effect

Model	F_{favor}	$F_{against}$	$MicF_{avg}$	$MacF_{avg}$
LSTM	59.88	74.47	67.17	56.19
LSTM_SP	60.88	74.49	67.69	58.39
LSTM_SP_ST	**63.54**	**74.89**	**69.22**	**60.16**

From Table 6, we can observe the following:

1) The LSTM_SP model, which only shares the representation of stance and sentiment classification information, can outperform the basic LSTM model. It indicates the effect of the shared representation learning between stance and sentiment information.

2) The LSTM_SP_ST model improves the LSTM_SP model by 1.53% and 1.77% in Micro F1 and Macro F1 scores respectively. It shows that feeding the representation from prefix into suffix is highly effective for the joint model. In addition, neural stacking is more efficient than shared representation. This may be because neural stacking allows deeper integration of the source model beyond one-best outputs, and further fine-tuning of the source model during the target model training.

4.6 Algorithm complexity analysis

In neural networks, the feed forward algorithm and back propagation algorithm are the primary time-consuming parts. The time complexity of the back propagation algorithm is $O(N_h \times N_w)$ (N_h is the number of hidden neurons and N_w is the number of weights) per time step for computing the Jacobian matrices, while the feed forward algorithm only takes $O(N_w)$ per time step. The vectorization of the tweets and the parameters that need to learn in the training period are the main space-consuming parts.

We compare the running time of the joint model with the LSTM and SVM-ngram models in Table 7. The computer specifications are as follows: GPU is GeForce GTX 980, graphics memory is 4G, CPU is Intel i7-4790, and memory is 8G. From Table 7, we can observe the following: 1) In the training period, the neural network models (joint and LSTM) consume more running time than the conventional discrete model (SVM-ngram). Even using GPU, the neural network models are still much slower than the discrete model. 2) However, the gap of the speed between the neural network model and the discrete model in the testing period is not as large as in the training period. 3) Comparing the joint model with the LSTM model, the running time consumed by the joint model is approximately twice as much as that of the LSTM model. This result is consistent with the actual situation, because the input data scale and model complexity in the joint model are approximately twice as much as those of the LSTM model.

4.7 Case study

In this section, we analyze the effects of sentiment information on stance detection through some examples drawn from the test set.

In Table 8, we list some examples with stance labels that are incorrectly inferred by the LSTM model but correctly predicted by the joint model. In E5 and E6, the opinion holders

Table 7 Running time comparisons (Unit: seconds)

Target	Joint				LSTM				SVM			
	Training		Test		Training		Test		Training		Test	
	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU
Atheism	38.5	361.5	3.6	2.8	20.6	180.7	1.7	1.0	0.0	0.8	0.0	0.2
Climate change is concern	35.0	281.9	3.3	2.6	18.6	141.7	1.7	0.9	0.0	0.6	0.0	0.1
Feminist movement	42.6	461.9	3.7	3.2	23.0	227.5	1.5	1.1	0.0	1.2	0.0	0.3
Hillary clinton	43.3	483.1	3.4	3.3	23.6	236.6	1.5	1.1	0.0	1.3	0.0	0.3
Legalization of abortion	42.7	463.9	3.4	3.5	23.0	341.0	1.5	5.9	0.0	1.2	0.0	0.3

Table 8 Examples drawn from the test set with stance labels that are incorrectly inferred by LSTM but correctly predicted by the joint model

ID	Tweet	Target	Gold-stance	Gold-sentiment	LSTM	Joint
E5	*@bbcweather Good to see more global awareness, thanks.*	Climate	Favor	Positive	None	Favor
E6	*Being a woman & being a feminist just means you not on your own side.*	Feminist	Against	Negative	Favor	Against
E7	*Do you Progressives know how dangerously close you are to suppressing free speech? Stop it. #inners #readyforhillary*	Hillary	Favor	Negative	Against	Favor
E8	*Congrats to my friends at @SBAList and @WRTL for their commitment in getting Pain-Capable Act passed. #wiright #theyfeelpain*	Abortion	Against	Positive	None	Against

Note: For each example, "Gold-stance" is the ground truth of stance, "Gold-sentiment" is the ground truth of sentiment, and "LSTM" and "Joint" mean the predicted stance label from the LSTM model and joint model respectively

directly express a positive or negative opinion to indicate their stance toward the target. In this case, it indicates the effectiveness of sentiment information. However, in E7 and E8, the stance toward the target and the opinion of the tweet are opposite. In E7, the opinion holder supports Hillary by criticizing suppression of free speech, which is opposed by Hillary, and expresses a negative opinion about suppressing free speech. In E8, the opinion holder opposes legalization of abortion by supporting Pain-Capable Act (Unborn Child Protection Act), which is opposed to the given target, and expresses a positive opinion about Pain-Capable Act. Obviously, the stance and sentiment are expressed toward different targets, and there exists an inherent connection between them. Therefore, sentiment information is also helpful in detecting the stance in this case.

We calculate the error rates of stance classification in the three-sentiment category (POSITIVE, NEGATIVE, and NEITHER) samples in the joint model. They are 30.2%, 32.5%, and 40%, respectively. An example of errors in NEITHER category is as follows:

E9: Target: *Climate Change is a Real Concern*

Tweet: *Closed door session begins. More after they decide on the entities.*

In E9, this tweet is predicted to be in favor of the given target while the ground truth is NONE. The poster refers irrelevant content to the given target and expresses no opinion about anything. Furthermore, we find that 68% of the samples with a stance label of NONE and a sentiment label of NEITHER were misclassified. Thus, we can observe that it is more difficult to detect stance without the help of sentiment information.

5 Conclusion

In this study, we proposed a novel neural network model to learn stance and sentiment jointly. Our approach thoroughly incorporates the interactions between stance and sentiment. Specifically, we employed a shared LSTM layer to learn the shared representation between stance and sentiment information. Furthermore, we fed the shared representation into a stance classification task as auxiliary input. Experiments demonstrated that our proposed joint learning approach can effectively detect stance and leverage the sentiment information. We also found that the number of sentiment samples plays a key role in improving the performance.

In our future work, we would like to collect massive domain samples annotated with sentiment to improve the performance. Furthermore, we plan to explore the influence of different domains and use one domain knowledge to help another one based on transfer learning.

Acknowledgements In building our system, we are grateful to all the people who have helped us. Jingjing Wang, Lu Zhang, Dong Zhang, and Suyang Zhu have provided help in programing; Zhenghua Li, Xing Wang, and Ziwei Fan have given us insightful comments etc. We also would like to thank the organizer of SemEval-2016 Task 6 for the hard work, especially in data annotation.

This work is supported by the National Natural Science Foundation of China (Grant Nos. 61331011, 61751206, 61773276, 61672366), Jiangsu Provincial Science and Technology Plan (BK20151222), Project of Natural Science Research of the Universities of Jiangsu province

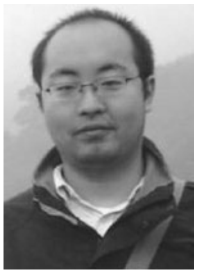
(16KJB520007) and Huaiyin Normal University Youth Talent Support Program (13HSQNZ07).

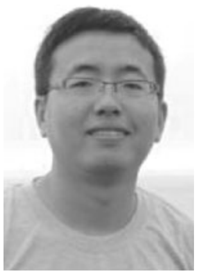
References

1. Mohammad S M, Kiritchenko S, Sobhani P, Zhu X, Cherry C. Semeval-2016 task 6: detecting stance in tweets. In: Proceedings of SemEval. 2016, 31–41
2. Somasundaran S, Wiebe J. Recognizing stances in online debates. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. 2009, 226–234
3. Murakami A, Raymond R. Support or oppose?: classifying positions in online debates from reply activities and opinion expressions. In: Proceedings of the 23rd International Conference on Computational Linguistics. 2010, 869–875
4. Anand P, Walker M, Abbott R, Tree J E, Bowmani R, Minor M. Cats rule and dogs drool!: classifying stance in online debate. In: Proceedings of the 2nd workshop on computational approaches to subjectivity and sentiment analysis. 2011, 1–9
5. Walker M A, Anand P, Abbott R, Grant R. Stance classification using dialogic properties of persuasion. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2012, 592–596
6. Hasan K S, Ng V. Stance classification of ideological debates: data, models, features, and constraints. In: Proceedings of the International Joint Conference on Natural Language Processing. 2013, 1348–1356
7. Sun Q, Wang Z, Zhu Q, Zhou G. Exploring various linguistic features for stance detection. In: International Conference on Computer Processing of Oriental Languages. 2016, 840–847
8. Mohammad S M, Sobhani P, Kiritchenko S. Stance and sentiment in tweets. 2016, arXiv preprint arXiv:1605.01655
9. Thomas M, Pang B, Lee L. Get out the vote: determining support or opposition from congressional floor-debate transcripts. In: Proceedings of the 2006 conference on empirical methods in natural language processing. 2006, 327–335
10. Bansal M, Cardie C, Lee L. The power of negative thinking: exploiting label disagreement in the min-cut classification framework. In: COLING (Posters). 2008, 15–18
11. Burfoot C, Bird S, Baldwin T. Collective classification of congressional floor-debate transcripts. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics. 2011, 1506–1515
12. Agrawal R, Rajagopalan S, Srikant R, Xu Y. Mining newsgroups using networks arising from social behavior. In: Proceedings of the 12th international conference on World Wide Web. 2003, 529–535
13. Sridhar D, Getoor L, Walker M. Collective stance classification of posts in online debate forums. In: Proceedings of the Joint Workshop on Social Dynamics and Personal Attributes in Social Media. 2014, 109–117
14. Johnson K, Goldwasser D. Identifying stance by analyzing political discourse on twitter. In: Proceedings of EMNLP Workshop on Natural Language Processing and Computational Social Science. 2016, 66–75
15. Volkova S, Bachrach Y, Armstrong M, Sharma V. Inferring latent user properties from texts published in social media. In: Proceedings of Association for the Advancement of Artificial Intelligence. 2015, 4296–4297
16. Lukasik M, Srijith P K, Vu D, Bontcheva K, Zubiaga A, Cohn T. Hawkes processes for continuous time sequence classification: an application to rumour stance classification in twitter. In: Proceedings of 54th Annual Meeting of the Association for Computational Linguistics. 2016, 393–398
17. Zubiaga A, Kochkina E, Liakata M, Procter R, Lukasik M. Stance classification in rumours as a sequential task exploiting the tree structure of social media conversations. 2016, arXiv preprint arXiv:1609.09028
18. Rajadesingan A, Liu H. Identifying users with opposing opinions in Twitter debates. In: International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction. 2014, 153–160
19. Mohammad S M, Kiritchenko S, Sobhani P, Zhu X, Cherry C. A dataset for detecting stance in tweets. In: Proceedings of 10th edition of the the Language Resources and Evaluation Conference (LREC). 2016, 3945–3952
20. Zarrella G, Marsh A. MITRE at semeval-2016 task 6: transfer learning for stance detection. In: Proceedings of SemEval. 2016, 458–463
21. Wei W, Zhang X, Liu X, Chen W, Wang T. Pkudblab at semeval-2016 task 6: a specific convolutional neural network system for effective stance detection. In: Proceedings of SemEval. 2016, 384–388
22. Pang B, Lee L, Vaithyanathan S. Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2002, 79–86
23. Yessenalina A, Yue Y, Cardie C. Multi-level structured models for document-level sentiment classification. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. 2010, 1046–1056
24. Brychcın T, Habernal I. Unsupervised improving of sentiment analysis using global target context. In: Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP. 2013, 122–128
25. Tang D, Qin B, Liu T. Document modeling with gated recurrent neural network for sentiment classification. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2015, 1422–1432
26. Khattri A, Joshi A, Bhattacharyya P, Carman M J. Your sentiment precedes you: using an author's historical tweets to predict sarcasm. In: Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. 2015, 25–30
27. Saif H, He Y, Fernandez M, Alani H. Contextual semantics for sentiment analysis of Twitter. Information Processing & Management. 2016, 52(1): 5–19
28. Sobhani P, Mohammad S M, Kiritchenko S. Detecting stance in tweets and analyzing its interaction with sentiment. In: Proceedings of the 5th Joint Conference on Lexical and Computational Semantics. 2016, 159–169
29. Titov I, McDonald R T. A joint model of text and aspect ratings for sentiment summarization. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. 2008, 308–316
30. Watanabe Y, Asahara M, Matsumoto Y. A structured model for joint learning of argument roles and predicate senses. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics. 2010, 98–102
31. Simova I, Vasilev D, Popov A, Simov K, Osenova P. Joint ensemble

model for POS tagging and dependency parsing. In: Proceedings of the 1st Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages. 2014, 15–25
32. Socher R, Perelygin A, Wu J Y, Chuang J, Manning C D, Ng A Y, Potts C. Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2013, 1631–1642
33. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. Journal of Machine Learning Research. 2011, 12(Aug): 2493–2537
34. Liu Y, Li S, Zhang X, Sui Z. Implicit discourse relation classification via multi-task neural networks. 2016, arXiv preprint arXiv:1603.02776
35. Zhou J, Xu W. End-to-end learning of semantic role labeling using recurrent neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. 2015, 1127–1137
36. Chen H, Zhang Y, Liu Q. Neural network for heterogeneous annotations. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2016, 731–741
37. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997, 9(8): 1735–1780
38. Graves A. Generating sequences with recurrent neural networks. 2013, arXiv preprint arXiv:1308.0850
39. Johnson R, Zhang T. Effective use of word order for text categorization with convolutional neural networks. 2014, arXiv preprint arXiv:1412.1058
40. Srivastava N, Hinton G E, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research. 2014, 15(1): 1929–1958
41. Tieleman T, Hinton G. Rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning. Technical Report, 2012
42. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the 13th International Conference on Artifical Intelligence and Statistics. 2010, 249–256
43. Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. 2013, arXiv preprint arXiv:1301.3781

Qingying Sun received the Master’s degree in July 2010 from the School of Computer Science and Technology, Nanjing University of Science & Technology, China. Since 2015, she has been a PhD candidate in the School of Computer Science and Technology, Soochow University, China. Her current research interests include natural language processing, sentiment analysis, stance detection and social computing.


Zhongqing Wang received his PhD degree in 2016 from the School of Computer Science and Technology, Soochow University, China. Since April 2016, he has been a postdoctoral research fellow at Singapore University of Technology and Design, Singapore. He is a lecturer in the School of Computer Science and Technology, Soochow University, China. His current research interests include natural language processing, sentiment analysis and social computing.


Shoushan Li received his PhD degree in 2008 from National Laboratory of Pattern Recognition, CASIA, China. He is a full professorin the School of Computer Science and Technology, Soochow University, China. His current research interests include natural language processing, social computing, and sentiment analysis.

Qiaoming Zhu received his PhD degree in 2006 from the School of Computer Science and Technology, Soochow University, China. He is a full professor in the School of Computer Science and Technology. His research interests include natural language processing, sentiment analysis, discourse analysis.

Guodong Zhou received his PhD degree in 1999 from the National University of Singapore, Singapore. He is a full professor in the School of Computer Science and Technology, and the Director of the Natural Language Processing Laboratory from Soochow University, China. His research interests include information retrieval, natural language processing.