Stance Prediction on Twitter Network

Advisor: Prof. Kevin Chen-Chuan Chang
Mentor: Amin Javari
Student: Zijie Huang

Abstract. With the proliferation of social media platforms, a large number of online users express their stances towards a given target such as Hillary Clinton to be "positive", "negative" or "neutral". Detecting users' stance has been a topic of interest in the past decade for policy-making process and public deliberation, etc. In recent years, deep learning has emerged as an effective means for solving stance prediction problems. However, previous deep learning models only focused on textual information regardless of the social tie among people. In this work, we present a novel deep learning framework that jointly considers **textual** information (i.e. what people say) and **social tie** among people (i.e. who they follows) to make prediction. The textual module extracts users' stances by using sequential information from what they say; the friends module explores friendship structure of a user's following list to make stance inference. Our proposed method can also be applied to the setting where people do not have textual information. This is useful in real-life scenarios because for a specific target, it is of great possibility that only a small group of people have opinionated content. We construct three datasets that has users' stances along with their textual and friendship information. We evaluate our approach on the SemEval 2016 Task 6 Twitter Stance Detection corpus and achieved better results compared with state-of-art methods.

Keywords: Stance prediction; social network mining; deep learning

1 Introduction

Stance classification is the task of determining whether a user is in favor of, against, or neutral towards a target of interest. This is an interesting task to study on social networks due to the abundance of opinionated language and social tie among people. Studying stance classification can be beneficial in various fields such as understanding public opinions on fundamental societal issues e.g, abortion or gun rights; mining public opinions about a certain product from a company, etc.

Popular stance classification methods generally fall into two categories: 1.) feature-based methods; 2.) deep learning methods. Feature-based methods often rely on explicitly engineering useful features to predict the stance of a user such as network structure and personal attributes information. Although these

methods have shown significant improvements in stance prediction performance, the feature engineering process requires much manual effort and extensive domain knowledge. Further more, feature-based methods fail to become generalized models, for they require different modeling under each application scenario.

With the development of deep learning techniques, recent work has proposed various methods to make stance prediction from textual information by utilizing NLP(natural language processing) frameworks such as RNN(recurrent neural network), CNN(Convolutional Neural Network), etc. However, stance prediction is a challenging task, and would require techniques that not only consider textual features or sentiment lexicon. To be more specific, we can infer a user's stance by looking at the friends that she/he follows. For example, if Bob follows Donald Trump and other republicans on twitter, it is of great possibility that Bob is in favor of Donald Trump.

Table 1. Example of User Following Information on Twitter.

User Name	**Friends**	**Stance**
Alice	{Donald Trump, Taylor Swift, Ketty Perry}	Favor
Bob	{Donald Trump and other Republicans}	Favor
Tom	{Hillary Clinton, Donald Trump and other Democrats}	Against

To address the above challenges, we proposed a unified model that jointly considers textual and friendship information. For the textual information , we utilize the standard Bi-direction LSTM model to capture the sequential features of a tweet. Since the number of tweets in training set is relatively small, we apply transfer learning to pretrain our LSTM model. To design the friendship model, we have two main insights. The first is that not all friends are relevant to a given target. If we look at irrelevant friends, they may add additional noise to the prediction model. For example, suppose we want to predict Alice's stance about Donald Trump and she has friends as listed in Tab 1. Taylor Swift and Ketty Perry are irrelevant signals that need to be treated differently from Donald Trump. For a binary classification, after the first filtering step we can get the useful information. However, stance prediction is a multi-classification problem, we should consider friends with different stances to make the final prediction. Therefore, the second insight is to fully utilize the friendship structure of the user. After we filter those irrelevant friends, we can group them by their stances towards the given target. This is because a stance from a group is more easily captured than the stance of an individual. For example if we look at the friends list of Tom in Tab 1, we can group them into three stance groups (favor,against,neutral). Then each group can be viewed as one component of the user's stance, and we can model the interactions among these groups to get the final prediction for this user. This idea is like to first get the harmonic component(different groups) of a waveform(user), and then recover the waveform(user)

as shown in Fig 1. Finally, we add a combinational layer on top of these two submodules to make stance prediction.

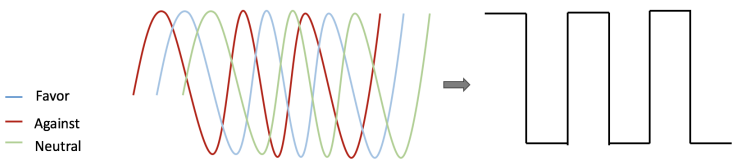

Fig. 1. Grouping Illustration.

Our main contributions are summarized as follows:

- We proposed a deep learning framework that jointly considers textual and friendship information to make stance prediction.
- We design a novel friendship model that captures the friendship structure and interactions among a user's friends.
- We construct three real datasets of stance prediction and conduct extensive experiments to verify the effectiveness of our propose models.

The rest of the paper is organized as follows. Sec. II discussed the related work. Sec. III illustrates our proposed models. Sec. IV presents the evaluation results of our methods and conclusion is drawn in Sec. V.

2 Related Work

Stance Prediction. Stance prediction is related to sentiment classification with a major difference that the target of interest may not be explicitly mentioned in the text and it may not be the target of opinion in the text. Previous work [2, 7] has shown that stance prediction is a challenging task, and would require techniques beyond traditional sentiment analysis methods that only consider textual features or sentiment lexicon. Specifically, Rui *et al.* [3] developed a weakly-guided stance modeling framework which is able to automatically predict user position on controversial issues and provide insights on the heated arguments. Ebrahimi *et al.* [4] investigated collective classification of stances on Twitter, using hinge-loss Markov random fields (HL- MRFs). However, none of them directly utilize only the following friend list of a user to make stance prediction and they are not deep learning frameworks.

Deep Learning for Stance Prediction. Recently, stance classification has attracted a significant amount of research attention from both text mining and natural language processing communities. Plenty of deep-learning frameworks were proposed to utilize textual information to make stance prediction. Isabelle *et al.* [1] proposed a target-aware LSTM model to incorporate target information

into prediction. Guido *et al.* [9] employed a recurrent neural network initialized with features learned via distant supervision. Wei *et al.* [8] develop a convolutional neural network with a "vote scheme" for prediction. However, all of these deep learning models only focus on dealing with textual information. Since friendship information is more easily obtained, our work jointly considers textual and friendship information to make stance prediction.

3 Stance Prediction on Twitter Network.

We now illustrate our proposed deep learning framework for stance prediction task. Below we first define our problem setting and then demonstrate the friend and text modules separately.

3.1 Problem Definition

In what follows, we use upper case bold letters such as $\mathbf{W}$ to denote matrices and lower case bold letter such as $\mathbf{x}$ to denote column vectors. The i-th element in vector $\mathbf{x}$ is denoted by $x(i)$.

Our problem is that given a user's tweet $s =< w_1 w_2 \cdots w_t >$ that contains t words and his/her following list $F = \{f_1, f_2 \cdots f_n\}$ that consists of n friends, predict his/her stance over a given target as "positive" (FAVOR), "negative" (AGAINST) or "neutral" (NONE).

3.2 User Text Module

Our user text module consists of two parts as shown in Fig 2: word embedding layer and tweet embedding layer.

Word Embedding Layer. An input sentence of length t is a word sequence $s =< w_1, w_2 \cdots w_t >$. Each word w in the vocabulary is described by a word vector $\mathbf{x}$. Let d be the length of each word vector $\mathbf{x}$ and n be the total number of words in the vocabulary. The trainable word embedding layer is a lookup table $\mathbf{X}^{d \times n}$ with word vectors as its columns. The word embedding layer is initialized using a skip-gram word2vec model. In more detail, we train a word2vec model on a corpus of 395,212 unlabelled tweets, collected with the Twitter Keyword Search API between November 2015 and January 2016, plus all the labelled tweets contained in our dataset. The unlabelled tweets are collected so that they contain the targets mentioned in our dataset. We combine this large unlabelled corpus with the our labelled training data and train a skip-gram word2vec model (dimensionality 100, 5 min words, context window of 5).

Tweet Embedding Layer. We use a variant of RNN called Bi-Direction Long Short Term Memory (Bi-LSTM), which is known to be effective for modeling sequences, to get the tweet representation. It improves representations of sequences by encoding a sequence from left to right and from right to left. Specifically, let r denotes the dimensionality of the LSTM hidden state, for the i-th word

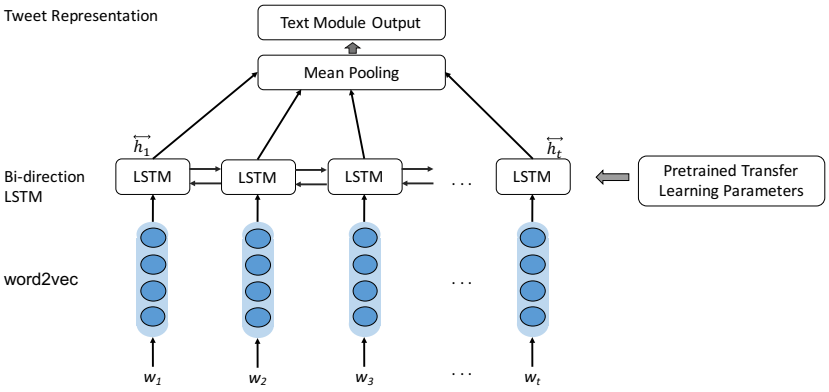

Fig. 2. User Text Module.

z

in the short-text, a Bi-LSTM takes as input $\mathbf{x_i} \in \mathbb{R}^d$, $\overrightarrow{h}_{\mathbf{i-1}} \in \mathbb{R}^r$, $\overleftarrow{h}_{\mathbf{i-1}} \in \mathbb{R}^r$ and produces $\overrightarrow{h}_{\mathbf{i}} \in \mathbb{R}^r$, $\overleftarrow{h}_{\mathbf{i}} \in \mathbb{R}^r$, $\overleftrightarrow{h}_{\mathbf{i}} \in \mathbb{R}^{2r}$ based on the following formulas

$$\begin{aligned} \overrightarrow{h}_{\mathbf{i}} &= LSTM_{fwd}(\mathbf{x_i}, \overrightarrow{h}_{\mathbf{i-1}}), \\ \overleftarrow{h}_{\mathbf{i}} &= LSTM_{bwd}(\mathbf{x_i}, \overleftarrow{h}_{\mathbf{i-1}}), \\ \overleftrightarrow{h}_{\mathbf{i}} &= \overrightarrow{h}_{\mathbf{i-1}} \oplus \overleftarrow{h}_{\mathbf{i-1}} \end{aligned} \tag{1}$$

where $\oplus$ denotes the concatenation operation.

Due to the limited number of labelled tweets in our dataset, we apply **transfer learning** to pretrain the parameters in tweet embedding layer. In this manner the network learned distributed sentence representations from a dataset containing a broad array of stance declarations rather than relying exclusively on the small sized labelled tweets. To be more specific , we first strip of all targets in the 395,212 unlabelled tweets. Then for each tweet, we use the target it contains as the label. If a tweet contains more than one targets, we select the most frequent target as its label. We trained the LSTM with gradient descent using AdaDelta and categorical cross entropy minimization and use it as the initial parameter settings of our user text module.

After getting all of the hidden states for each word $\{\overleftrightarrow{h}_{\mathbf{1}}, \overleftrightarrow{h}_{\mathbf{2}}, \cdots \overleftrightarrow{h}_{\mathbf{t}}\}$, we calculate their average to get the tweet representation $\mathbf{h_{text}} \in \mathbb{R}^{2r}$, which is also the output of the text module.

$$\mathbf{h_{text}} = \frac{1}{t} \sum_{t} \overleftrightarrow{h}_{\mathbf{t}} \tag{2}$$

3.3 User Friends Module

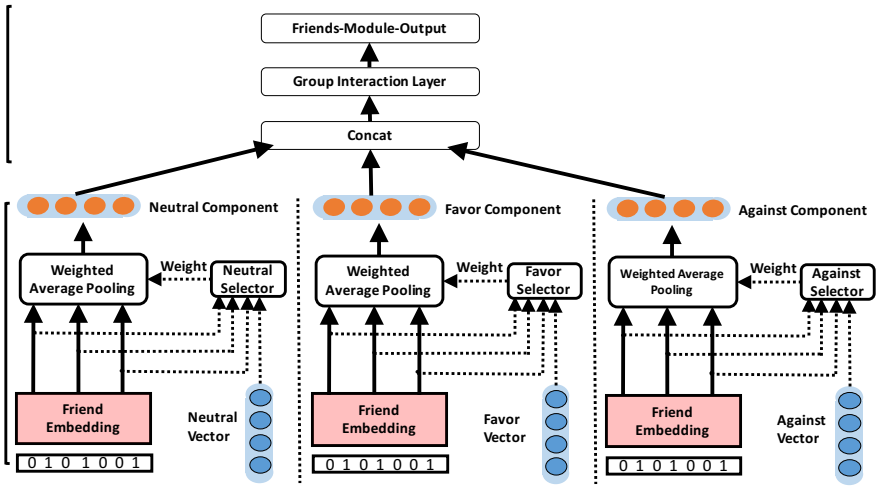

Fig. 3. User Friends Module.

As mentioned before, our user friends module consists of two parts as shown in Fig 3: friends grouping layer and group interaction layer. The main idea is to first group friends into favor, against and neutral groups (i.e. find the harmonic component of a waveform) and then send them into the group interaction layer to get the prediction results (recover the waveform from those harmonic components).

Friends Grouping Layer. Assume $\mathbf{m_j} \in \mathbb{R}^r, j = 1, 2, 3$ are latent stance vectors representing favor, against, neutral respectively and r is the dimensionality of the vector. The input friends list of length n is represented by a zero-one vector $\mathbf{u} \in \mathbb{R}^n$ where each element represents a friend id in the total friend space. $u(i)$ equals to one when the i-th friend is a friend of the user.

In order to group friends, each friend is mapped into a feature vector $\mathbf{v_i} \in \mathbb{R}^r$ via the friend embedding layer and then interact with each stance vector to get three weight $\alpha(i, j) \in \mathbb{R}, j = 1, 2, 3$. Each score represents the probability that friend f_i belongs to the group. To be more specific , we consider relating $\alpha(i, j)$ with the embedding vector $\mathbf{v_i}$ and $\mathbf{m_j}$. The rationale is that the embedding vectors are supposed to encode the information about the stance of the friends, thus they can be used to determine the contribution(weight) of friends for each stance group. We parameterize $\alpha(i, j)$ as a function with $\mathbf{v_i}$ and $\mathbf{m_j}$ as the input, where $\mathbf{W} \in \mathbb{R}^{r \times 1}$ and $b \in \mathbb{R}$ are respectively the weight matrix and bias vector that project the input into a weight score, $\odot$ denotes the element-wise multiplication.

$$\alpha(i,j) = Sigmoid((\mathbf{v_i} \odot \mathbf{m_j})\mathbf{W} + b) \tag{3}$$

The group vector is then the weighted mean pooling of the user's friend vectors.

$$\mathbf{g_j} = \frac{1}{|\{j|u(j)! = 0\}|} \sum_{j:u(j)!=0} \alpha(i,j)\mathbf{v_j}, \quad j = 1,2,3 \tag{4}$$

Group Interaction Layer. After getting the grouping vectors $\mathbf{g_1}, \mathbf{g_2}, \mathbf{g_3} \in \mathbb{R}^r$, we utilize them to form the stance vector of our predicted user. Inspired by the recent success of using neural networks to model the interaction of different components, we similarly use a Multi-Layer Perception (MLP) to parameterize the group interaction layer as follows, where $\oplus$ denotes the concatenation operation, $\mathbf{W} \in \mathbb{R}^{h,3r}, \mathbf{b} \in \mathbb{R}^h$ are respectively the weight matrix and bias vector that project the stance vector into a hidden layer.

$$\mathbf{s} = ReLU(\mathbf{W}(\mathbf{g_1} \oplus \mathbf{g_2} \oplus \mathbf{g_3}) + \mathbf{b}) \tag{5}$$

The stance vector $\mathbf{s}$ is the output of the friend module.

3.4 The Joint Model and Model Training

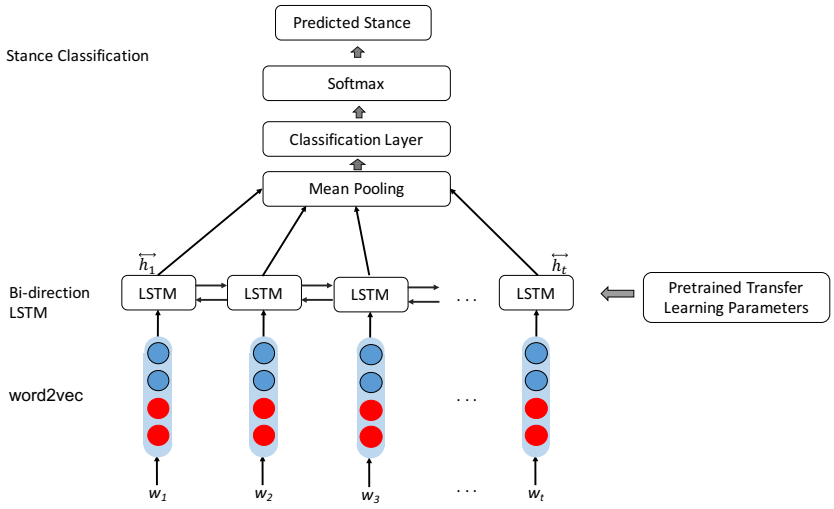

Fig. 4. Joint Framework.

According to our discussions in Sections 3.2 and 3.3, we propose a fused model that jointly consider textual and friendship information to make stance prediction.

In order to better take advantage of both textual and friends information, we append the stance vector $\mathbf{s}$ from the friend module to the embedding of each word in original tweet. The friend-augmented embedding of a word is then expressed as $\mathbf{e_t} = \mathbf{x_t} \oplus \mathbf{s}$, where $\oplus$ is the vector concatenation operation.

The joint model is illustrated in Fig 4 where the red parts is the stance vector from the friends module and the blue parts is the word embedding vector.

We use cross-entropy loss to train our model end-to-end given a set of training data $\{s_i, u_i, y_i\}$, where s_i and u_i are the textual and friendship information of the i-th user to be predicted, y_i is the one hot representation of the ground-truth stance for the $i-th$ user. We represent this model as a black-box function $f(s, u)$ whose output is a vector representing the probability of stance. The goal of training is to minimize the loss function:

$$\mathcal{L} = -\sum_i \sum_j y_j^i log f_j(s_i, u_i) + \lambda ||\theta||^2 \tag{6}$$

where i is the index of data and j is the index of class. $\lambda||\theta||^2$ is the L_2-regularization term and θ is the parameter set.

4 Evaluation

In this section, we compare the performance of our proposed methods with several strong baselines on stance detection. We firstly describe the experimental setting, then present the comparative results.

Table 2. Distribution of instances of SemEval-2016 Task 6 Dataset.

Target	# total	# total friends	# avg friends	% of instances in Train # train	favor	against	none	% of instances in Test # test	favor	against	none
Atheism	564	11986	251	451	16.2	63.9	20.1	113	20.1	62.6	17.4
Climate Change is Concern	462	12135	276	370	61.2	3.0	35.8	92	64.0	5.6	30.3
Feminist Movement	527	5689	112	422	34.2	45.2	20.5	105	34.5	45.4	20.0
Hillary Clinton	692	35778	793	689	16.6	56.7	26.6	295	16.7	59.5	23.6
Legalization of Abortion	702	23735	471	653	17.0	58.4	24.6	280	10.7	67.2	22.1
Donald Trump	707	27811	728	576	19.4	43.8	36.7	131	18.9	45.2	35.8
Crawled_Unlab*	395212	–	–	–	–	–	–	–	–	–	–

Dataset. We construct a real dateset based on Semval2016 [6] with six different targets. The statistics of the dataset is shown in Tab 3. We trained six distinct classifiers, one for each of the six targets under consideration in the evaluation. The embedding and recurrent layers of each classifier were initialized with the weights obtained from the pre-training process described above. The remainder of the weights were randomly initialized and the network was trained with stochastic gradient descent using a learning rate of 0.015 and momentum of 0.9.

Metrics. We adopt the macro-average of the F1-score for 'favor' and the F1-score for 'against' as the bottom-line evaluation metric, which is the same as

in [6]. To be more specific, the F-1 scores are calculated as follows.

$$\begin{aligned} F_{favor} &= \frac{2P_{favor}R_{favor}}{P_{favor} + R_{favor}} \\ F_{against} &= \frac{2P_{against}R_{against}}{P_{against} + R_{against}} \\ F_{average} &= \frac{F_{favor} + F_{against}}{2} \end{aligned} \tag{7}$$

Baseline Algorithms. To the best of our knowledge, no previous work has designed a deep learning framework that utilize both textual and friendship information, therefore we compare our models to several textual-only baselines.

- *Conditional Encoding* [1]: A conditional LSTM encoding framework, which builds a representation of the tweet that is dependent on the target.
- *Convolutional Neural Network* [8]: A convolutional neural network with a "vote scheme".
- *GMF* [5]: A generalized matrix factorization(GMF) that is used to model item interactions in recommendation system.
- *NCF* [5]: A neural network-based collaborative filtering(ncf) that is used to model item interactions in recommendation system.

Table 3. Results of SemEval-2016 Task 6 Dataset.

Method	Atheism F_{avg}	Climate F_{avg}	Trump F_{avg}	Feminist F_{avg}	Hillary F_{avg}	Abortion F_{avg}
Conditional Encoding	62.47	61.72	**63.54**	54.54	59.39	57.28
Convolutional Neural Network	61.18	48.24	61.17	51.33	49.92	61.04
GMF	68.57	67.41	56.04	63.78	71.28	66.72
NCF	69.73	68.81	59.07	65.01	69.50	66.36
Grouped-friendship	**71.31**	69.54	59.76	66.74	68.59	67.28
Friendship-Tweet	70.89	**70.01**	58.73	**67.04**	**70.18**	**69.59**

5 Conclusion

In this paper, we jointly consider textual and friendship information to make stance prediction. We first propose a transfer-learning-based textual module that extract the stance information from what people say, and then design a friends module that contains a grouping layer and a group interaction layer. Next, we design a joint model that incorporates the aforementioned two submodules. Finally, we have conducted extensive experiments on real datasets to verify the effectiveness of our proposed methods.

6 Appendix

6.1 Two more datasets.

Here is a summary of another two datasets that I have collected for our stance prediction problem, which may be used in the future.

Airline Dataset. The airline dataset consists of six different airline companies and users' stance about them (favor,against, neutral). The basic statistics of the dataset is shown as below.

Table 4. Distribution of instances of SemEval-2016 Task 6 Dataset.

				% of instances in Train				% of instances in Test		
# total	**# total friends**	**# avg friends**	**# train**	favor	against	none	**# test**	favor	against	none
11656	52167	257	9301	16.79	62.24	20.96	2355	17.40	62.76	19.83

Health Dataset. The Health Care Reform (HCR) dataset was built by crawling tweets containing the hashtag *#hcr* (health care reform) in March 2010. Based on this dataset, I collect all the friendship information.

Table 5. Distribution of instances of Health Dataset.

# total	**# total friends**	**# avg friends**	favor	against	none
2516	13948	319	21.50	54.89	18.68

References

1. Augenstein, I., Rocktaschel, T., Vlachos, A., Bontcheva, K.: Stance detection with bidirectional conditional encoding. In: International Conference on Empirical Methods in Natural Language Processing(EMNLP). pp. 876–885 (2016)
2. Dhanya Sridhar, James Foulds, B.H.L.G., Walker., M.: Joint models ofdisagreement and stance in online debate. In: AnnualMeeting ofthe Association for Computational Linguistics (ACL). pp. 116–124 (2010)
3. Dong, R., Sun, Y., Wang, L., Gu, Y., Zhong, Y.: Weakly-guided user stance prediction via joint modeling of content and social interaction. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1249–1258. CIKM '17 (2017)
4. Ebrahimi, J., Dou, D., Lowd, D.: Weakly supervised tweet stance classification by relational bootstrapping. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 1012–1017 (2016)
5. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web. pp. 173–182. WWW '17 (2017)

6. Saif M. Mohammad National, S.K.: Semeval-2016 task 6: Detecting stance in tweets saif. In: Proceedings ofSemEval-2016. pp. 31–41 (2016)
7. Somasundaran, S., Wiebe, J.: Recognizing stances in ideological on-line debates. In: Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text. pp. 116–124 (2010)
8. 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. pp. 384–488 (2016)
9. Zarrella, G., Marsh, A.: Mitre at semeval-2016 task 6: Transfer learning for stance detection. In: Proceedings of SemEval-2016. pp. 458–463 (2016)