Rumor Detection with Hierarchical Recurrent Convolutional Neural Network

Xiang Lin[1], Xiangwen Liao[1](✉), Tong Xu[2], Wenjing Pian[1], and Kam-Fai Wong[3]

[1] College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China
742734768@qq.com, {liaoxw,wenjingpian}@fzu.edu.cn

[2] Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China, Hefei, China
tongxu@ustc.edu.cn

[3] The Chinese University of Hong Kong, Sha Tin, Hong Kong
kfwong@se.cuhk.edu.hk

Abstract. Automatic rumor detection for events on online social media has attracted considerable attention in recent years. Usually, the events on social media are divided into several time segments, and for each segment, corresponding text will be converted as vectors for various neural network models to detect rumors. During this process, however, only sentence-level embedding has been considered, while the contextual information at the word level has been largely ignored. To address that issue, in this paper, we propose a novel rumor detection method based on a hierarchical recurrent convolutional neural network, which integrates contextual information for rumor detection. Specifically, with dividing events on social media into time segments, recurrent convolution neural network is adapted to learn the contextual representation information. Along this line, a bidirectional GRU network with attention mechanism is integrated to learn the time period information via combining event feature vectors. Experiments on real-world data sets validate that our solution could outperform several state-of-the-art methods..

Keywords: Rumor detection · Recurrent convolutional neural network

1 Introduction

A rumor is commonly defined as a statement whose truth value is unverifiable or deliberately false [1]. As the fast development of social media, the exponentially increasing rumors have caused a huge amount of loss in people's lives and properties. For instance, in 2015, a rumor about "shootouts and kidnappings by drug gangs happening near schools in Veracruz" spread through Twitter and Facebook, which caused severe chaos [9]. Therefore, automatic rumor detection techniques are urgently required to quickly identify rumor messages and dynamically monitor the propagation.

Traditionally, most of the literature regarded rumor detection as a two categories classification problem. To that end, large efforts have been made on rumor detection via machine learning methods, in which a wide variety of features manually crafted from the content are incorporated. Usually, these features included geographic location information, verification questions and corrections [2, 3], emotional polarity [4, 5], and propagation tree features [6]. Along this line, support vector machine [4], decision tree [3, 5, 8] and other classifiers were used to classify rumors. However, these methods require manual extraction of features, which takes a considerable amount of time and manpower. Moreover, the robustness of obtained features is questionable, especially for the imbalanced data set. At the same time, some other researchers attempt to detect rumors via feature representation learning methods. These methods processed the text at the sentence level, and used deep neural networks like bidirectional GRU network [9] and convolutional neural network [10], to mine key features in complicated social media sceneries. An obvious limitation of these models is that they used sentence embeddings on the text of each time period, but ignored contextual information at the word level, resulting in lower prediction performance.

To address that issue, in this paper, we proposed a novel rumor detection method based on a hierarchical recurrent convolutional neural network, which integrates contextual information for rumor detection. Specifically, with dividing events on social media into time segments, recurrent convolution neural network is adapted to learn the contextual representation information. Along this line, a bidirectional GRU network with attention mechanism is integrated to learn the time period information via combining event feature vectors. Extensive validations on real-world data sets have verified the effectiveness of our method with significant improvements compared with several state-of-the-art methods.

2 Related Work

In general, two categories of methods were widely studied for rumors detection, i.e., rumor detection based on traditional machine learning techniques, and rumor detection based on feature representation learning methods.

For the first category, usually, the prior arts manually design features for rumor detection with extracting these features from the textual content. Then, they used various classifiers to detect rumors. For instance, Castillo et al. [5] proposed four classification features (user features, message features, topic features, and propagation features) to detect rumors. Also, Qazvinian et al. [12] extracted content-based features, network-based features and Weibo-specific features to identify rumors. In 2015, Zhao et al. [2] used terms such as “unconfirmed”, “rumor” or “debunked” to find enquiries and corrections tweets. At the same time, Ma et al. [7] used dynamic time series to capture changes in a set of social environment features over time. Later, Ma et al. [6] proposed a tree-based method in 2017, which captured high-order propagation patterns on Twitter. Due to the deepening of feature design, the detection performance of rumors based on traditional machine learning has been significantly improved, but the manual design features require a lot of manpower and material resources, while the features are not robust enough. The above problems are still unresolved.

For the second category, recently, LSTM [13], RNN [14], CNN [15] and some other related techniques have been applied to text feature representation learning. These methods were also applied to feature representation learning in rumor detection problems. In 2016, Ma et al. [9] used neural network models to detect rumors for the first time. They applied RNN models to learning information changes at different time intervals associated with each event. In 2017, Yu et al. [10] converted the Weibo text to doc2vec and then applied CNN to learning the event representation. Ruchansky et al. [16] proposed a hybrid model CSI, which used singular value decomposition to obtain the user's feature vector and then applied the LSTM network to learning text features to detect rumors. In 2018, Ma et al. [18] proposed a top-down RvNN model and a bottom-up RvNN model for rumor detection. Guo et al. [19] proposed a novel hierarchical neural network combining with social information to detect rumors. Liao et al. [11] proposed a hierarchical attention network to obtain the hidden layer representations to detect rumors. Ma et al. [17] used the framework of multi-task learning to conduct rumor detection and stance classification.

In summary, the methods based on feature representation learning have achieved significant results, but these methods mainly focused the text representation of each time period at the sentence level, which ignored the contextual information at the word level. Thus, this defect may severely impact the accuracy of rumor detection.

3 Proposed Method

3.1 Problem Definition

In general, rumor detection in social media could be formulated as a binary classification problem, which will be defined as follow: Given a set of Weibo (or Twitter) events $E = \{e_1, e_2, e_3, \ldots\}$, where e_i represents an event containing a number of microblogs (or tweets). For computational efficiency, we follow previous work [19] and divide the posts in e_i into different time intervals; a set of categories $L = \{l_1, l_2\}$, where l_1, l_2 represent rumor and non-rumor, respectively. Using the model algorithm, each event E is mapped to a category, i.e., $E \rightarrow L$. The input to the question is the relevant microblog (or tweet) information for an event e_i to output the label of event: Rumor or Non-rumor.

3.2 Recurrent Convolutional Neural Networks with Feature Attention Network

As shown in Fig. 1, we first use a recurrent convolution neural network [20] to learn contextual representation information, and then utilize a bidirectional GRU network, as well as an attention mechanism which contains the event feature vector to learn the time period information.

For each social media event, our model has two inputs:

(1) A textual representation w_{ijk} of the social media event in each time period, where i represents the social media event, and j is the j^{th} time period and k indicates the k^{th} word of the social media text.

(2) The feature vector F_e of each event, including user-based features, content-based features, signal features, and so on.

Corresponding, the output of this model is the judgement whether target social media event is a rumor.

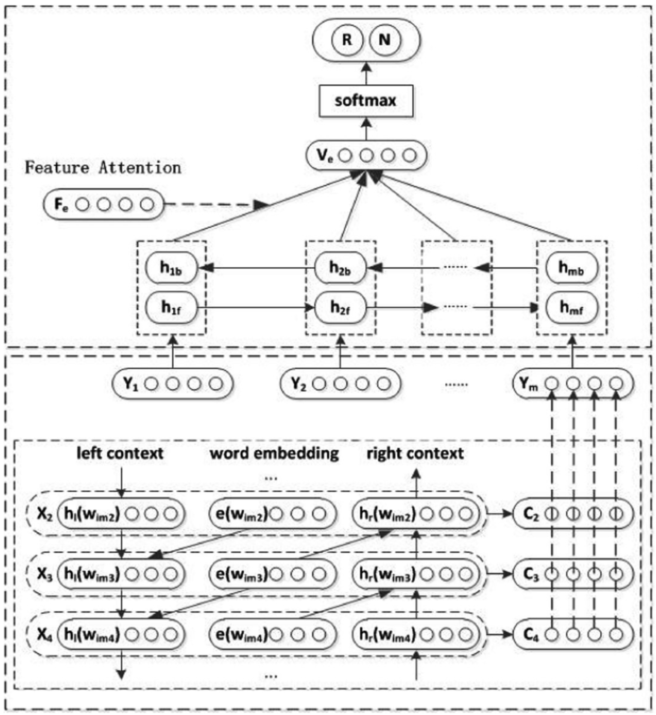

Fig. 1. Our RCNN-FAN framework for rumor detection.

To deal with this problem, our model mainly consists of two modules below:

(1) **Social media text representation based on a recurrent CNN.** This module captures contextual information with the recurrent structure and constructs the representation of text using a convolutional neural network. The module outputs the textual representation Y_t of the contextual information.
(2) **Time period representation based on BiGRU-FeatureAttention network.** This module regards the text representation Y_t learned by the previous module as the input, obtains the time hidden layer representation V_e via the BiGRU-FeatureAttention network, and then uses the fully-connected layer for event classification.

Social Media Text Representation Based on a Recurrent CNN. This module mainly includes a bidirectional recurrent neural network (bidirectional LSTM) and a convolutional neural network. Here $h_l(w_{ijk})$ is the hidden state of the previous word of word w_{ijk}, and $h_r(w_{ijk})$ is the hidden state of the next word of word w_{ijk}. $h_l(w_{ijk})$ and $h_r(w_{ijk})$ can be obtained by the following formula:

$$h_l\left(\mathrm{w}_{ijk}\right) = f\left(W_l h_l\left(w_{ijk-1}\right) + W_{el}\mathrm{e}\left(w_{ijk-1}\right)\right) \tag{1}$$

$$h_r\left(w_{ijk}\right) = f\left(W_r h_r\left(w_{ijk+1}\right) + W_{er} e\left(w_{ijk+1}\right)\right) \tag{2}$$

Where $e(w_{ijk-1})$ and $e(w_{ijk+1})$ are word embedding representations of the word w_{ijk-1} and the word w_{ijk+1}, respectively. Also, $h_l(w_{ijk-1})$ and $h_r(w_{ijk+1})$ represent the hidden states of the word w_{ijk-1} and the word w_{ijk+1}, respectively. W_l, W_{el}, W_r, and W_{er} are their respective weight matrices, and f is a nonlinear activation function tanh.

After calculating the two context hidden states $h_l(w_{ijk})$ and $h_r(w_{ijk})$ of current word w_{ijk}, two states are merged with the word embedding representation of current word as

$$X_k = \left(h_l\left(w_{ijk}\right), e\left(w_{ijk}\right), h_r\left(w_{ijk}\right)\right) \tag{3}$$

We use a convolutional layer to learn its text representation to get the result X_k, and then use a maximum pooled layer to convert text representation vectors into fixed lengths. Where C_k is the result of X_k processed by the convolutional layer Conv1D, Y_t represents the text representation vector of the t^{th} time period, and the a^{th} element of Y_t is the maximum value of the a^{th} element of $C_k (k \in [1, n])$

$$C_k = Conv1D(X_k) \tag{4}$$

$$Y_t = \max_{k \in [1,n]} C_k \tag{5}$$

Time Period Representation Based on BiGRU-FeatureAttention Network. This module contains a bidirectional GRU and a feature attention layer. The input of the bidirectional GRU is a text representation vector for each time period, and the bidirectional GRU layer can use information of the previous and subsequent time segments to learn the hidden layer representation of current time period.

$$h_{tf} = \overrightarrow{GRU}(Y_t), t \in [1, m] \tag{6}$$

$$h_{tb} = \overleftarrow{GRU}(Y_t), t \in [m, 1] \tag{7}$$

$$h_t = \left(h_{tf}, h_{tb}\right) \tag{8}$$

Where h_{tf} is the hidden layer representation of the forward GRU in time period t, h_{tb} is the hidden layer representation of the backward GRU in time period t, and m is the length of time segment, i.e., the number of time segments. Besides, h_t is the output of the bidirectional GRU in the t^{th} time period, obtained by the connection of h_{tf} and h_{tb}.

We add a feature attention layer after the bidirectional GRU, so that the model can learn key information in this time period. The input of feature attention layer is the output h_t of bidirectional GRU at each time period, and hidden layer representation vector V_e of event is generated. Formula for the feature attention layer is as follows:

$$u_t = \tanh\left(W_h h_t + W_f F_e + b_h\right) \tag{9}$$

$$\beta_t = \frac{exp\left(u_t^T u_d\right)}{\sum_m exp\left(u_m^T u_d\right)} \tag{10}$$

$$V_e = \sum_t \beta_t h_t \tag{11}$$

Where h_t is the hidden layer representation of the bidirectional GRU, F_e is the event feature vector and we obtain u_t. The content-based features and the user-based features are used in the feature vector F_e. Description of features are summarized in Table 1. The SIGNAL TEXT feature in Table 1 is a signal text that determines whether there is an enquiry or correction in the text, such as "really?", "rumor", and so on. In 2015, Zhao et al. [2] used regular expressions to match signal texts. Similarly, we also use regular expressions to match the signal text, such as "*is\s(?:that|this|it)\s true*", "*real?|really?|unconfirmed*", and so on.

Table 1. The features of the social media event.

	Feature	Description
Content-based features	AVERAGE LENGTH	The average length of twitter
	SIGNAL TEXT	The fraction of twitter with enquiries and corrections
User-based features	VERIFIED USERS	The fraction of verified users
	AUTHOR DESCRIPTION	The fraction of users that provide a personal description
	AVERAGE COUNT FOLLOWERS	The average of user count followers
	AVERAGE COUNT FRIENDS	The average of user count friends

Also, we have β_t as the result obtained by normalization of u_t via the softmax function and we obtain the hidden layer representation V_e of the event. Then, the event hidden layer representation V_e is fed into a fully connected layer, and we use the softmax function to get the label $\widehat{L_e}$ of social media event for classification.

$$\widehat{L_e} = \text{softmax}(W_e V_e + b_e) \tag{12}$$

3.3 Model Training

In the proposed model, the cross-entropy error between probability distribution of prediction and ground truth is defined as the loss function. Correspondingly, the formula for cross-entropy loss function is as follows, where y^e represents the true label of the event (1 for rumor, and 0 for non-rumor), $\widehat{L_e}$ denotes a predicted event category label, and E represents the event dataset for training.

$$\text{Loss} = -\sum\nolimits_{e \in E} \left[y^e \log\left(\widehat{L_e}\right) + (1 - y^e) \log\left(1 - \widehat{L_e}\right) \right] \tag{13}$$

In order to minimize the occurrence of over-fitting, we use the L2 regularization for GRU layer weights, and use the dropout probability for GRU layer. We also use the Adam optimization algorithm [21] to improve the convergence speed. In each iterative process of the algorithm, the model regards the event dataset as input, which is trained via RCNN-FAN to obtain the label of the event for calculating the loss.

4 Experiments and Results

4.1 Datasets

We evaluate our proposed model on a public dataset used by Ma et al. [9] in 2016. This dataset includes two parts: Twitter and Sina Weibo. This dataset is used in the literature [6, 9–11, 16, 19] and is a classic dataset for rumor detection problems.

The Sina Weibo dataset contains 4664 events, in which each contains multiple Weibo texts. The content of each Weibo text is completely provided by the original dataset. Meanwhile, the Twitter dataset contains 992 events, each with multiple tweets. However, since the dataset only gives the id of the tweet in each event, the content of the tweet needs to be downloaded via the Twitter API interface. Considering that many tweets might be unavailable as time goes by, the same for some tweets for certain events, e.g., event with ID "E354". Therefore, we removed some data to ensure the quality of experiments (Table 2).

Table 2. Statistics of the datasets.

Statistic	Sina Weibo	Twitter
Events #	4664	992
Rumors #	2313	498
Non-rumors #	2351	494
Microblogs #	3805656	1101985
Users #	2746818	491229
Avg. time length/event (hours)	2460.7	1582.6
Avg. # of posts/event	816	1111
Max # of posts/event	59318	62827
Min # of posts/event	10	10

4.2 Experimental Setup

The running environment used in this experiment is as follows: Intel(R) Xeon(R) CPU E5-2620 v4 2.10 GHz, with the operating system as Ubuntu 14.04.5 LTS, the memory as 32 GB RAM, and the GPU as Tesla K40m. Besides, we used the development platform Python 3.6.2. The learning rate is set to 0.001, the regularization coefficient λ is set to 0.01, the dropout probability is set to 0.2. The word vector of Twitter is trained using the public 100-dimensional glove word vector. Since there is no public word vector suitable for this experiment in the Sina Weibo, word vector is trained by the model. The time period segmentation method of this experiment uses the method that splits social media text of an event into parts of equal size to ensure that the number of social media texts is equal in each time period [19].

We compared our model with the following state-of-the-arts baselines:

(1) **DTC model** [5]. This model extracts four categories of features from the collected tweet data and then uses the J48 decision tree to detect rumors.
(2) **SVM-TS model** [7]. This model proposes a dynamic series time structure (DSTS) to capture the temporal characteristics, then use the SVM classifier for identifying rumors.
(3) **DT-Rank model** [2]. This model first uses some regular expressions to identify tweets with enquiry and correction signals, then clusters these tweets, and matches tweets from non-signal tweets according to the summary statement after clustering. Finally, the tweets are sorted using statistical features.
(4) **GRU-2 model** [9]. This model uses tf-idf to calculate the text representation of each time period, then uses the double-layer GRU model for training.
(5) **CAMI model** [10]. This model learns the text representation via the paragraph vector, and then uses the CNN model to train.
(6) **CSI model** [16]. This model uses the RNN model to process microblog events, then performs the singular value decomposition of the user-user association matrix to get the user score of event.
(7) **HSA-BLSTM model** [19]. This model uses the bidirectional LSTM and the attention mechanism with the social feature vector to identify rumors.
(8) **HAN-FC model** [11]. This model uses doc2vec on the text of each time period and then utilizes a hierarchical attention network to detect rumors.

4.3 Performance Comparison

In order to verify the validity of the experiment, we compared our model with several state-of-the-arts baselines. The results of these experiments are shown in Table 3, in which RCNN-FAN (Recurrent Convolutional Neural Networks with Feature Attention Network) presents our technical framework. There are two categories, i.e., R stands for rumor and N stands for non-rumor. Also, four evaluation metrics were utilized to measure the performance, namely Accuracy, Precision, Recall and F1.

Table 3. Results of comparison with different methods (R: Rumor; N: Non-rumor).

Method	C	Sina Weibo				Twitter			
		Acc.	Pre.	Rec.	F1	Acc.	Pre.	Rec.	F1
DT-Rank	R	0.755	0.711	0.832	0.767	0.614	0.618	0.584	0.601
	N		0.812	0.682	0.741		0.610	0.643	0.626
DTC	R	0.831	0.847	0.815	0.831	0.709	0.690	0.772	0.729
	N		0.815	0.847	0.830		0.733	0.643	0.685
SVM-TS	R	0.857	0.839	0.885	0.861	0.716	0.689	0.793	0.738
	N		0.878	0.830	0.857		0.754	0.639	0.692
GRU-2	R	0.910	0.876	0.956	0.914	0.723	0.712	0.743	0.727
	N		0.952	0.864	0.906		0.735	0.704	0.719
HSA-BLSTM	R	0.934	0.943	0.933	0.938	0.765	0.729	**0.838**	0.780
	N		0.924	0.936	0.930		**0.813**	0.693	0.748
CAMI	R	0.937	0.937	0.940	0.938	0.753	0.727	0.823	0.772
	N		0.938	0.934	0.936		0.789	0.682	0.732
CSI	R	0.951	0.939	0.962	0.950	0.773	0.806	0.714	0.758
	N		0.963	0.942	0.952		0.746	**0.831**	0.787
HAN-FC	R	0.964	0.955	**0.977**	0.966	0.787	0.778	0.800	0.789
	N		**0.974**	0.949	0.962		0.797	0.775	0.786
RCNN-FAN	R	**0.970**	**0.966**	**0.977**	**0.971**	**0.799**	**0.814**	0.770	**0.792**
	N		**0.974**	**0.962**	**0.968**		0.785	0.827	**0.805**

In traditional machine learning experiments, the SVM-TS model achieved the accuracy of 85.7% and 71.6% in the two datasets, respectively, and obtained the best effect in traditional machine learning. For experiments based on feature representation, the HAN-FC model achieves the accuracy of 96.4% and 78.7% in the two datasets, and the best results are obtained for all baselines. It can also be seen that the method based on feature representation learning is better than the method based on traditional machine learning. Our RCNN-FAN model achieved 97% accuracy in the Sina Weibo dataset and achieves 79.9% accuracy in the Twitter dataset, which outperformed the HAN-FC model by 0.6% and 1.2%, respectively. Moreover, it outperformed the SVM-TS model by 11.3% in the Sina Weibo dataset and 8.3% in the Twitter dataset, and it performed much better than the baselines. In addition, the Weibo dataset and Twitter datasets expressed different effects. This may because there is more noise information on the Twitter dataset [9], and some data is not available, it has a certain impact on the final experimental results.

In addition, on the Sina Weibo dataset, the accuracy, recall, and F1 values of our model reaches the highest value of the baselines, and most of the values exceed the highest values of the baselines. On the Twitter dataset, the F1 value of our model also exceeds the highest value in the baselines.

5 Conclusion

In this paper, we proposed a novel method to automatically identify rumors on social media. To be specific, RCNN-FeatureAttention network was adapted to learn the contextual representation information, as well as the bidirectional GRU network with a feature attention layer to learn the time period information. Extensive validations on real-world data sets have verified the effectiveness of our proposed solution, which significantly outperformed several state-of-the-art methods.

Acknowledgement. This research project was supported by the National Natural Science Foundation of China (No. 61772135 and No. U1605251), the Open Project of Key Laboratory of Network Data Science & Technology of Chinese Academy of Sciences (No. CASNDST201708 and No. CASNDST201606), the Open Project of National Laboratory of Pattern Recognition at the Institute of Automation of the Chinese Academy of Sciences (201900041), Fujian Provincial Natural Science Foundation Project (2017J01755), CERNET Innovation Project (NGII20160501).

References

1. DiFonzo, N., Bordia, P.: Rumor Psychology: Social and Organizational Approaches. American Psychological Association, Washington, D.C. (2007)
2. Zhao, Z., Resnick, P., Mei, Q.: Enquiring minds: early detection of rumors in social media from enquiry posts. In: 24th International Conference on World Wide Web, Florence, pp. 1395–1405 (2015)
3. Liang, G., He, W., Xu, C., et al.: Rumor identification in microblogging systems based on users' behavior. IEEE Trans. Comput. Soc. Syst. **2**, 99–108 (2015)
4. Zhang, Q., Zhang, S., Dong, J., et al.: Automatic detection of rumor on social network. In: 4th CCF Conference on Natural Language Processing and Chinese Computing, Nanchang, pp. 113–122 (2015)
5. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on twitter. In: International Conference on World Wide Web, Hyderabad, pp. 675–684 (2011)
6. Ma, J., Gao, W., Wong, K.F.: Detect rumors in microblog posts using propagation structure via kernel learning. In: 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, pp. 708–717 (2017)
7. Ma, J., Gao, W., Wei, Z., et al.: Detect rumors using time series of social context information on microblogging websites. In: 24th ACM International Conference on Information and Knowledge Management, Melbourne, pp. 1751–1754 (2015)
8. Sun, S., Liu, H., He, J., et al.: Detecting event rumors on Sina Weibo automatically. In: The Web Technologies and Applications - 15th Asia-Pacific Web Conference, Sydney, pp. 120–131 (2013)
9. Ma, J., Gao, W., Mitra, P., et al.: Detecting rumors from microblogs with recurrent neural networks. In: 25th International Joint Conference on Artificial Intelligence, New York, pp. 3818–3824 (2016)
10. Yu, F., Liu, Q., Wu, S., et al.: A convolutional approach for misinformation identification. In: 26th International Joint Conference on Artificial Intelligence, Melbourne, pp. 3901–3907 (2017)
11. Liao, X.W., Huang, Z., Yang, D.D., et al.: Rumor detection in social media based on hierarchical attention network. Sci Sin Inform **48**(11), 1558–1574 (2018). (in Chinese)

12. Qazvinian, V., Rosengren, E., Radev, D.R., et al.: Rumor has it: identifying misinformation in microblogs. In: The Conference on Empirical Methods in Natural Language Processing, Edinburgh, pp. 1589–1599 (2011)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
14. Elman, J.L.: Finding structure in time. Cogn. Sci. **14**, 179–211 (1990)
15. Lecun, Y., Boser, B., Denker, J.S., et al.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**, 541–551 (1989)
16. Ruchansky, N., Seo, S., Liu, Y.: CSI: a hybrid deep model for fake news detection. In: the 2017 ACM on Conference on Information and Knowledge Management, Singapore, pp. 797–806 (2017)
17. Ma, J., Gao, W., Wong, K.F.: Detect rumor and stance jointly by neural multi-task learning. In: The Web Conference Companion, Lyon, pp. 585–593 (2018)
18. Ma, J., Gao, W., Wong, K.-F.: Rumor detection on twitter with tree-structured recursive neural networks. In: The Meeting of the Association for Computational Linguistics, Australia, pp. 1980–1989 (2018)
19. Guo, H., Cao, J., Zhang, Y., et al.: Rumor detection with hierarchical social attention network. In: The ACM International Conference on Information and Knowledge Management, Italy, pp. 943–951 (2018)
20. Lai, S., Xu, L., Liu, K., et al.: Recurrent convolutional neural networks for text classification. In: 29th AAAI Conference on Artificial Intelligence, USA, pp. 2267–2273 (2015)
21. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv:1412.6980 (2014)