



# Deep Attention Model with Multiple Features for Rumor Identification

Lina Wang<sup>1,2</sup>(✉), Wenqi Wang<sup>1,2</sup>, Tong Chen<sup>1,2</sup>, Jianpeng Ke<sup>1,2</sup>,  
and Benxiao Tang<sup>1,2</sup>

<sup>1</sup> Key Laboratory of Aerospace Information Security and Trusted Computing,  
Ministry of Education, Wuhan University, Wuhan, China

{lnwang,wangwenqi\_001,chentong,kejianpeng,tangbenxiao}@whu.edu.cn

<sup>2</sup> School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China

**Abstract.** With the rapidly development of social networks and advances in natural language processing (NLP) techniques, rumors are extremely common and pose potential threats to community. In recent years, massive efforts are working on detecting rumors by using various techniques like simply investigating the content of texts, exploring the abnormality of propagation. However, these techniques are not ready to fully tackling this emerging threats due to the dynamic variations of rumors in a period of time. In this paper, we observed that the user feedback provides a clean signal for determining the trend of rumors, thus we combine the text content and the improved representation of network topology to characterize the dynamic features of rumors in a period of time. In detection, we employ a deep attention model with proposed features for spotting the minor differences between legitimate news and rumors. Experimental results show that our approach give an accuracy more than 94.7% in detecting rumors and outperforms previous approaches. Our studies also give a new insight that user interactions could be working as an important asset in rumor identification.

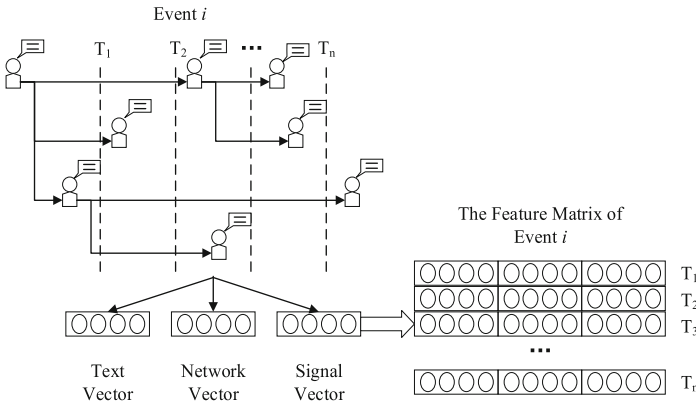
**Keywords:** Attention mechanism · Representation learning · Rumor identification · Multiple Features

## 1 Introduction

Nowadays, online social networks have greatly promoted the communication between humans, resulting in an explosive growth of various information. They have also become important ways for ordinary people to obtain external knowledge. However, this convenience also promotes the widespread of rumors. Recently, there is no standard definition of what a rumor is. DiFonzo et al. [8] define rumor as unverified and instrumentally relevant information statements in circulation that arise in contexts of ambiguity, danger or potential threat. Gupta et al. [11] regard rumor as uncertain or deliberately forged information. Matsuta et al. [19] point that a rumor spreads like an infectious disease. According to the multiple features we used, a rumor is defined here as the information with

harmful content, abnormal topology structure of networks, and disparate feedback (*i.e.*, inquiry and correction signals by users [31]). Rumors have negative influence on public security and stability of society. They not only damage the credibility of social media, but also destroy the orderly development of society. For example, someone said that high concentration alcohol could prevent coronavirus disease outbreaking in 2019 (COVID-19), leading to the death of hundreds of Iranians. Therefore, effective rumor identification methods are urgently needed to deal with this issue.

Recently, the majority of rumor identification methods extract features (*e.g.*, text content [25], propagation patterns [17], and user profiles [28]) to train machine learning models to distinguish rumors from non-rumors. However, rumors can not be described accurately by using one or a few features among them. Text content-based methods usually ignore the dynamic structure of the propagation process and highly rely on the characteristics of information content. Propagation-based methods are limited by the artificially constructed features and the learning representation ability of models. The effects are not as good as expected. User profiles-based methods require high accuracy and a large amount of training data. They ignore the role of content in propagation, so that the results may have some deviations.



**Fig. 1.** Feature representation of an event in the propagation cycle. The propagation of event  $i$  is divided into  $n$  time periods. We extract the relevant features in each time period and splice them to form a feature matrix for training.

To deal with this issue, we use text content, topology structure of networks, and user feedback signals to represent the characteristics of an event in the propagation cycle. Then, we introduce the attention mechanism, which can find the elements closest to detection task from the input sequence to help classify the text, to a deep learning model for rumor identification. The framework of our method is shown in Fig. 1, and our contributions are summarized as follows:

- We combine user feedback signals with text content and topology structure of networks to characterize the features of an event for rumor identification task.
- We propose a representation learning method for network nodes based on the space-time similarity and improve the Struct2vec method to make it more suitable for the propagation process of events on online social networks.
- Our method achieves a relatively good result with 94.2% accuracy. It also performs better in the early detection of rumors than some works.

## 2 Related Works

Rumor identification is a hot research topic in various disciplines, and there exist a lot of relevant studies. Most of them utilize different features for distinction. Hence, the selection and extraction of features are significant to rumor identification. Takahashi et al. [25] found the differences in vocabulary distribution between rumors and non-rumors and use this feature for detection. Sun et al. [24] extracted 15 features related to content, users profiles, and multimedia to identify event rumors. Ma et al. [16] recognized the deficiencies of user profiles, content and propagation characteristics, and introduced the time series of information to improve the detection efficiency. Wu et al. [27] proposed a graph-kernel based hybrid support vector machine (SVM) classifier to utilize the propagation structure of the messages, which was ignored in prior works. All these works are based on traditional machine learning methods, and the features are extracted manually which are time-consuming and labor-intensive.

Recently, deep learning (DL) has achieved great performance in various natural language processing tasks. Therefore, researchers have tried to use DL-based models to identify rumors and received better results. Ma et al. [15] first used a recurrent neural network (RNN) to learn the vector representation of information-related texts over different periods. RNN-based rumor detection is faster and more accurate than the methods at the time. Chen et al. [6] introduced a self-attention mechanism based on the work of [15] to capture contextual variations of relevant posts over time. Yu et al. [29] proposed a novel method based on convolutional neural network (CNN), which can flexibly extract key features and shape high-level interactions among them. Yuan et al. [30] extracted the global structural information and combined with the semantic information for detection. Huang et al. [12] trained a graph convolutional network with graph structural information in the user behavior, which was not considered in other works.

## 3 Preliminaries

### 3.1 Description of Rumor Identification

Rumor identification is to determine whether the information of an event transmitted on social networks is a rumor or not. It can be seen as a two-category problem. The formal definition of this task is as follows:

For a given set  $E = \{e_1, e_2, \dots, e_i, \dots, e_h\}$  with  $h$  events and a label set  $L = \{l_1, l_2\}$ ,  $e_i$  stands for a set of messages related to the  $i$ -th event in  $E$ . Each event contains  $n$  pieces of information  $m_i$ , timestamp  $t_i$  and related attributes  $f_i$ . Thus, an event is composed of some time-series information, which is symbolized as  $e_i = \{(m_{i1}, t_{i1}, f_{i1}), \dots, (m_{ij}, t_{ij}, f_{ij}), \dots, (m_{in}, t_{in}, f_{in})\}$  ( $j = 1, 2, \dots, n$ ). The labels  $l_1$  and  $l_2$  represent rumor and non-rumor respectively. The rumor identification task is to map  $e_i$  into a corresponding category by learning a pre-trained classification model  $F$ , *i.e.*,  $F: e_i \rightarrow l, l \in \{l_1, l_2\}$ . The input of the model is the event  $e_i$ , and the output is a discriminant label for the event.

### 3.2 Division of an Event

Division of the overall propagation phase can be seen as a transformation from a complete cycle into a series of different periods. For example, the messages in the set  $m_i$  of event  $e_i$  is divided into  $n$  intervals according to the timestamp  $t_{ij}$ , where  $n$  is the number of intervals, and  $j$  ranges from 1 to  $n$ . The time-series division of an event cycle is divided below.

We assume that the earliest timestamp in the event  $e_i$  is  $startTime_i$ , and the latest timestamp is  $endTime_i$ .  $e_i$  is divided into  $n$  intervals at equal length as shown in (1).

$$T_{ij} = \frac{j}{n} * endTime_i + \frac{n-j}{n} * startTime_i \quad (1)$$

where  $T_{ij}$  is the end time of the  $j$ -th period in the linear division of  $e_i$ .

But the distribution of information in an event has a significant long-tail phenomenon. It indicates that most of the information is concentrated in the early stage of propagation. Then the number of information about the event drops sharply with time. To eliminate the long-tail phenomenon, we obtain the logarithmic intervals by nonlinear division instead of linear division. Hence, the interval of the latter period becomes larger and larger. The distribution of the data in each period has a good consistency. The calculation process of nonlinear division is shown in (2).

$$t_{ij} = (endTime_i - startTime_i + 1)^{\frac{T_{ij} - startTime_i}{endTime_i - startTime_i}} + startTime_i - 1 \quad (2)$$

where  $t_{ij}$  is the end time of the  $j$ -th period in the logarithmic interval division of  $e_i$ .

### 3.3 Propagation of an Event

Due to the complexity of network topology, the representations of propagation process in existing methods have two main problems. One is the incomplete use of propagation features. Existing researches tend to focus on one aspect of the propagation process, so that the information disseminated throughout its cycle

is not fully learned. For example, researchers only use the time-series features [15, 16] or content features [9, 18]. They intentionally ignore the complexity of network topology. Thus, the dynamic variability of online social networks is lack of consideration. The other one is a lack of quantitative expression for user opinion [14, 26]. Because the semantic representation of potential topics is limited by short text corpus, the feedback information and user opinions of events (i.e. user feedback signals) in the networks are not well quantified. In response to the two problems, our method takes better use of the dynamic feature and user feedback signals to represent the propagation process of an event on online social networks.

## 4 The Multiple Features

The multiple features consist of text content, network topology, and user feedback signals. In this section, we introduce the constructions of network topology and user feedback signals.

### 4.1 Text Content

On online social network, text content includes two forms: one is the content of the message published by the original publisher, and the other is the reposting of the former by some users, *i.e.*, a copy of the original message. There is no difference in their contents, so that researchers usually regard them as the same one to study the characteristics of the words, symbols and uncertainties they contain. In this paper, we convert the text content into a vector according to word2vec, and then combine the features of topology feedback signals to construct a feature matrix for rumor identification.

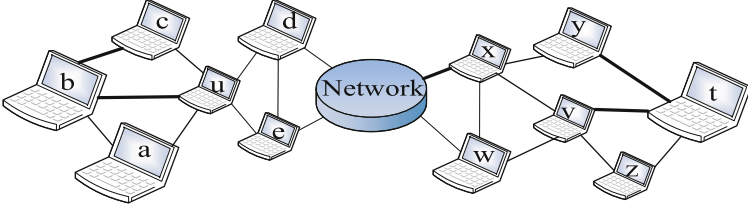
### 4.2 Network Topology

Struct2vec [21] is a framework for learning latent representations of network nodes. Inspired by it, we propose a representation learning method for network nodes based on the space-time similarity. We have done some improvements on the Struct2vec. The attributes of node feature, the similarity of ordered degree sequence and the average passive response time which are not considered in Struct2vec are used in our method.

### 4.3 Definition of the Similarity Between Nodes

Figure 2 is a diagram of node similarity on online social networks. We assume that the thickness of the edge between two nodes in Fig. 2 represents the degree of influence. The size of a node represents its characteristic properties, such as the number of fans. Generally, node  $u$  and node  $v$  are not similar in an open social network according to the similarity assumption of neighbors. Because they are not directly connected and do not share any neighbor nodes. In that case, the

probability that  $u$  and  $v$  appear in the same walk sequence by the traditional representation algorithm is almost zero. But  $u$  and  $v$  will have some similarities from the perspectives of spatial structure, degree of influence and node attributes. The degrees of  $u$  and  $v$  are 5 and 4 respectively. They are connected to other parts of the network through 2 nodes from spatial structure. Both  $u$  and  $v$  only have high influence on individual nodes in the view of influence. From the view of node attributes, the sizes of node  $u$  and node  $v$  are similar. The distributions of their neighbor nodes also have some similarities. Hence, it can be said that  $u$  and  $v$  are similar.



**Fig. 2.** Diagram of node similarity on social network. Each icon represents a node and they are connected by the networks.

Therefore, principles of the similarity between nodes in a directed graph with node attributes are defined as follows. First, the more similar the feature attributes of nodes are, the higher the similarity is. If their neighbor nodes also have similar distributions, the similarity of them will increase. Second, the closer the degrees of two nodes are, the higher the similarity is. If their neighbor nodes also have similar degrees, the similarity will increase. Third, the closer the average passive response time between two nodes is, the higher the similarity is. The passive response time of a node is better to describe its influence or similarity on social networks than the response time.

**Structural Similarity of Loop Nodes.** Given a directed graph  $G = \langle V, E \rangle$  with node attributes, the directed edge  $e_{ij}$  represents the forwarding behavior of node  $i$  for node  $j$ . Let  $R_k(u)$  denote the set of nodes with the shortest distance  $k$  to node  $u$ , where  $R_0(u)$  represents the node  $u$  itself.  $R_1(u)$  represents the directly connected neighbor set of  $u$ . Let  $D_i$  denote the indegree of node  $i$ . The loop structure refers to the part with a loop in the directed cycle graph(DCG). The definition of  $k$ -hop loop structure similarity  $f_k(u, v)$  for given nodes  $u$  and  $v$  is in (3).

$$f_k(u, v) = f_{k-1}(u, v) + g(S(R_k(u), S(R_k(v)))) \cdot \sum_c \alpha_c g(F_c(R_k(u)), F_c(R_k(v))) \quad (3)$$

where  $k \geq 0$  and  $f_{-1} = 0$ .  $\alpha_c$  is the weight of feature  $c$ .  $g(S_1, S_2)$  is the sequence similarity calculated by the dynamic time warping algorithm [4] for sequence  $S_1$  and sequence  $S_2$ . The sequence  $S(V)$  is calculated in (4).

$$S(V) = \text{sort}([D_1 \cdot e^{-\Delta t_1}, \dots, D_n \cdot e^{-\Delta t_n}]) \quad (4)$$

where  $e$  is the natural constant.  $\Delta t_i$  is the average passive response time of node  $i$ .  $\text{sort}$  is the function for ascending order according to their values. The ordered sequence  $F_c(N)$  of the feature  $c$  in node set  $N$  is calculated in (5).

$$F_c(N) = \text{sort}([c_1, \dots, c_n]) \quad (5)$$

**Construction of Hierarchically Weighted Graph.** The distance of the local loop structure between two nodes is calculated by (3). Then we construct a hierarchically weighted graph  $M$  based on random walk sampling by the method in [21].  $M$  is made up of  $(k^* + 1)$  layers. Each layer in  $M$  is composed of all the nodes in  $G = \langle V, E \rangle$ . Nodes in the set  $V$  are pairwise connected without directions. The current layer represents the jump probability based on the nearest  $k$ -hop loop distance of the nodes, where  $k = 0, \dots, k^*$ . The weights  $w_k(u, v)$  of the edge between  $u$  and  $v$  in the  $k$ -th layer are calculated in (6).

$$w_k(u, v) = e^{-f_k(u, v)} \quad (6)$$

Nodes belonging to different levels are connected by directed edges. Hence, each node is connected with its corresponding upper node and lower node. The definition of weight on a directed edge connected to the upper node is shown in (7).

$$w(u_k, u_{k+1}) = \log(\Gamma_k(u) + e) \quad (7)$$

The definition of weight connected to the lower node is shown in (8).

$$w(u_k, u_{k-1}) = 1. \quad (8)$$

where  $\Gamma_k(u)$  is the number of edges in  $k$ -th layer that are connected to the node  $u$  with weights greater than the average. The calculation process is shown in (9):

$$\Gamma_k(u) = \sum_{v \in V} 1 \cdot \text{bool}((w_k(u, v) > \bar{w}_k)) \quad (9)$$

$\text{bool}$  is the boolean function, and  $\bar{w}_k$  is the average weight of all edges for the  $k$ -th layer.

**Node Sampling Based on the Biased Random Walk.** To catch the node sequences, we sample on the hierarchically weighted graph  $M$  using a biased random walk. Each sampling is decided according to the pre-set probability walking at the current, upper or lower layer. Each time the sampled nodes are more inclined to select similar nodes to current ones in local structure, influence, and other features.

If it walks at the current layer, we assume that it is the  $k$ -th layer. The probability from node  $u$  to node  $v$  is shown in (10).

$$p_k(u, v) = \frac{e^{-f_k(u, v)}}{Z_k(u)} \quad (10)$$

where  $Z_k(u) = \sum_{v \in V, v \neq u} e^{-f_k(u,v)}$ . It is the normalization factor for node  $u$  in  $k$ -th layer.

If it walks at the upper layer, the  $(k + 1)$ -th layer is selected with the probability shown in (11).

$$p_k(u_k, u_{k+1}) = \frac{w(u_k, u_{k+1})}{w(u_k, u_{k+1}) + w(u_k, u_{k-1})} \quad (11)$$

If it walks at the lower layer, the  $(k - 1)$ -th layer is selected with the probability shown in (12).

$$p_k(u_k, u_{k-1}) = 1 - p_k(u_k, u_{k+1}) \quad (12)$$

**Representation of Network Topology.** The sequence  $H_S$  for global user nodes is obtained by biased random walk sampling on the probabilistic jump of  $M$ . The following construction of network topology is shown in Fig. 3. There are  $T$  nodes in the graph  $G = \langle V, E \rangle$ , where  $T \in \{u_1, u_2, \dots, u_T\}$ . This model is used to predict the central nodes for a given node sequence, and then these central nodes are linked to form the network topology. The optimization objective is shown in (13).

$$\frac{1}{T} \sum_{t=i}^{T-i} \log p(u_t | u_{t-i}, \dots, u_{t+i}) \quad (13)$$

where  $i$  represents the size of a sliding window.  $p(u_t | u_{t-i}, \dots, u_{t+i})$  is the output of the model. The prediction result is shown in (14).

$$p(u_t | u_{t-i}, \dots, u_{t+i}, \mathbf{S}) = \frac{e^{y_{u_t}}}{\sum_j e^{y_j}} \quad (14)$$

$y_j$  is the non-normalized likelihood probability of node  $j$  calculated in (15).

$$y = b + Uh(u_{t-i}, \dots, u_{t+i}; H) \quad (15)$$

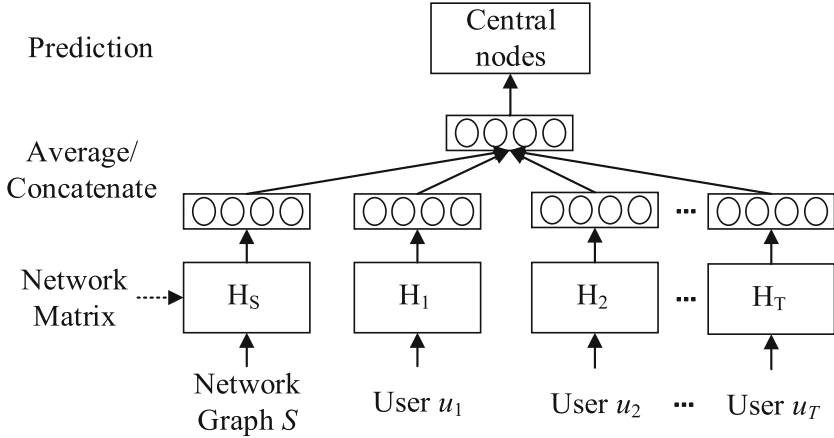
$U$  and  $b$  are parameters of the output layer.  $h$  is averaged by the node vectors in  $H$ .

#### 4.4 Feedback Signals

Feedback signals are the feedback information of users for an event, and they have an obvious time effect. The information about the same event will lead to different sub-events which stand for various point of views. The changes of views are related to the mode of propagation [17]. Hence, feedback information of users can help us to have a better understanding of the event.

The user feedback signals have achieved significant results on rumor identification [2, 24]. But the usage of this feature often has two problems, which are incomplete collection [31] and poor scalability. To overcome these problems, we efficiently use this feature and propose a mining method based on seed words.





**Fig. 3.** Training model of network structure vector.  $H_i$  represents the node sequence obtained by sampling the  $i$ -th node, where  $i$  ranges from 1 to  $T$ .

Our approach utilizes the word vectors [23] pre-trained in the external corpus (e.g., Wikipedia). It extends the scope of semantic sources and ensures the accuracy of feedback signals in the overall semantic space. The corresponding algorithm is shown in Algorithm 1.

**Generation of Seed Words:** The regular expression of inquiry and correction signals [31] is used as the initial generation method. The information content of datasets is regularly matched to obtain the initial seed words.

**Matching Calculation of Inquiry and Correction Signals:** In the process of finding a match, each seed word is separately calculated to find a word whose cosine similarity in the pre-trained word vectors  $P$  is greater than a threshold.

**Construction of Feedback Signals:** All the seed words obtained in the previous steps are seen as the word set  $Q$  of feedback signals. The information sequence of event  $e_i$  is divided into  $n$  periods by the nonlinear segmentation method. The information text sequence  $d_{i1}, d_{i2}, \dots, d_{in}$  from  $e_i$  in each nonlinear period is obtained. Then matching process between the text sequence  $d_{ij}$  of the event  $e_i$  in the  $j$ -th period and the word set of feedback signals is carried out by Aho-Corasick [1].

## 5 Model

In this section, we analyze the influence of multiple features on online social networks and design an attention model for rumor identification. The basic architecture of our model is shown in Fig. 4.

**Input Layer.** The network topology  $(w_{i1}, w_{i2}, \dots, w_{in})$ , user feedback signal  $(s_{i1}, s_{i2}, \dots, s_{in})$  and text content  $(d_{i1}, d_{i2}, \dots, d_{in})$  of event  $e_i$  in each non-linear period are used as the input data.

**Algorithm 1.** Mining algorithm of user feedback signals

---

**Input:** event set  $E = e_1, e_2, \dots, e_m$ , number of iterations  $\gamma$ , number of event periods  $n$ , similarity threshold  $t$ , pre-training word vector set  $P$

**Output:** feedback signal matrix  $S$  for each period of the event

```

1:  $Q \leftarrow \text{RexMatch}(E)$ 
2: for iter  $\leftarrow 1, \dots, \gamma$ : do
3:    $L \leftarrow |Q|$ 
4:   for  $j \leftarrow 1, \dots, L$ : do
5:      $U_j \leftarrow \text{GetSimilarWords}(Q[j], t, P)$ 
6:      $Q \leftarrow \text{AddWord}(Q, U_j)$ 
7:   end for
8: end for
9: for  $i \leftarrow 1, \dots, m$ : do
10:   $d_{i1}, d_{i2}, \dots, d_{in} \leftarrow \text{Split}(e_i, n)$ 
11:  for  $j \leftarrow 1, \dots, n$ : do
12:     $S_{ij} \leftarrow \text{AhoCorasick}(d_{ij}, Q)$ 
13:  end for
14: end for
15: return  $S$ 

```

---

**Embedding Layer.** The multiple features are structured as input to the embedding layer. Then the heterogeneous features are normalized.

**Encoder Layer.** The output of the embedding layer is processed by the encoding layer to obtain time-series information of the event.

**Attention Layer.** The attention layer weights the obtained information from the encoding layer to get related content of the event in the propagation process.

**Output Layer.** The output of the attention layer is used as an input of a fully connected layer for predicting the category of event.

## 5.1 Embedding Layer

Due to the multiple features of the event come from different feature domains, they are not suitable as the direct inputs to the model. Therefore, the primary role of the embedding layer in the network architecture is to transform them into vectors. Then, these vectors are standardized and aggregated as inputs for an encoding layer. The input of an embedding layer is the original feature data  $x_t$  shown in (16) of the event at period  $t$ , including the network topology vector  $w_t$ , the user feedback signal vector  $s_t$  and the text vector  $d_t$ .

$$x_t = \text{aggregate}(d_t, w_t, s_t) \quad (16)$$

where *aggregate* is the function to connect the vectors.

The output of an embedding layer acts as the input  $c_t$  for the encoding layer at period  $t$ . The calculation process is shown in (17).

$$c_t = \tanh(W_a x_t + b_a) \quad (17)$$

where  $W_a$  is the weight matrix and  $b_a$  is the bias. *tanh* is the activation function.

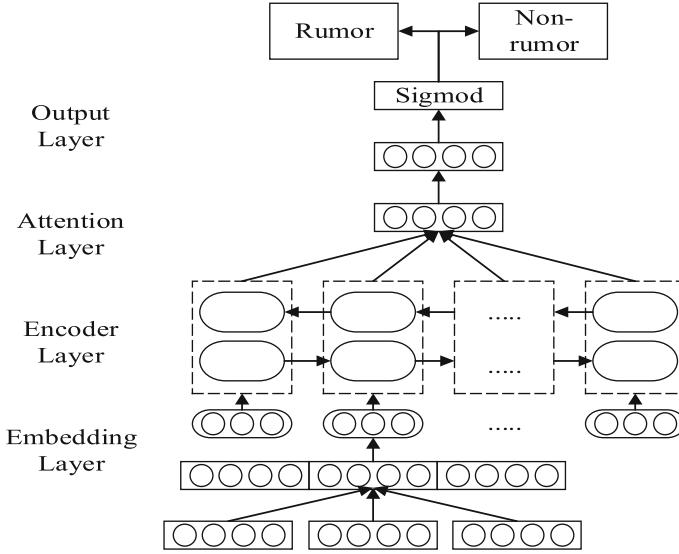


Fig. 4. Basic architecture of the model

## 5.2 Encoding Layer

To obtain the relationship of response time between the rumor and the event in general propagation process, the Bi-directional Gated Recurrent Unit [7] (Bi-GRU) model is adopted in this layer. Bi-GRU can gain long-term dependence on the data. It also has the flexibility to handle variable-length inputs. Hence, it is suitable for capturing information about event propagation in time-series relation. The calculation of the encoding layer is shown in (18):

$$\begin{cases} h_t = (\vec{h}_t, \overleftarrow{h}_t) \\ \vec{h}_t = \overrightarrow{GRU}_1(c_t), t = 1, \dots, n \\ \overleftarrow{h}_t = \overleftarrow{GRU}_1(c_t), t = n, \dots, 1 \end{cases} \quad (18)$$

where  $\vec{h}_t$  represents the output value of the forward part at period  $t$  and  $\overleftarrow{h}_t$  represents that of the backward part.  $h_t$  is used as an input to the successor network.  $\overrightarrow{GRU}_1$  and  $\overleftarrow{GRU}_1$  are the forward and backward propagation process respectively.

## 5.3 Attention Layer

This layer introduces the attention mechanism [3] by adding different weights to the output sequence  $(h_1, h_2, \dots, h_n)$ . By weighting the output of the encoding layer, more context information of the original data can be utilized while aligning

the input and output. The input of attention layer is  $h_t$  and the output  $v_e$  is calculated in (19):

$$\begin{cases} u_t = \tanh(W_c h_t + b_c) \\ a_t = \frac{e^{u_t^T u_c}}{\sum_j e^{u_j^T u_c}} \\ v_e = \sum_j a_j h_j \end{cases} \quad (19)$$

where  $W_c$  is the weight matrix and  $b_c$  is the bias.  $u_c$  represents the initial random weight of the attention layer.  $a_t$  represents the attention weight of the potential information in Bi-GRU. The attention module in the network structure is used for automatically learning the attention weight  $a_t$ . It can capture the correlation between  $h_t$  and the hidden state of the decoder. Then, these attention weights are used to construct the content vector  $v_e$ . The output of an attention layer is used as the input of a fully connected layer for the final rumor discrimination. In addition, it can also be used as an independent analysis of events.

#### 5.4 Output Layer

The final output discrimination of the output layer is shown in (20).

$$\hat{L}_e = \sigma(W_e v_e + b_e) \quad (20)$$

where  $\hat{L}_e$  represents the predicted category of the event.  $\sigma$  is the activation function.  $W_e$  is the weight matrix and  $b_e$  is the bias. We use the cross-entropy loss function to measure the difference between the predicted category and the ground-truth category. The Adam optimization algorithm [13] is used for training iteration as shown in (21).

$$Loss = -\frac{1}{N} \sum_{j=1}^N [L_j \ln \hat{L}_j + (1 - L_j) \ln(1 - \hat{L}_j)] + \frac{\lambda}{2} \|\theta\|_2^2 \quad (21)$$

where  $N$  represents the total number of samples in the training set and  $L_j$  represents the ground-truth category of the  $j$ -th event.  $\theta$  is a set of parameters for the model.  $\lambda$  is a hyper-parameter.

## 6 Experiments

### 6.1 Dataset

We use the classic dataset which has been applied in the work of [15, 17, 22, 29] on the rumor detection. It is divided into Weibo data and Twitter data. But the Twitter data only provides the ID of each event, and the contents need to be obtained through the official interface. Some of them can not be obtained due to access rights or non-existence. After statistics, about fifteen percent of Twitter

data has lost. To carry out a complete experimental comparison on the same dataset, we use Weibo data to conduct experiments. The entire dataset has a total of 2313 rumors and 2351 non-rumor samples, and each of them contains a number of information.

## 6.2 Baselines

In order to verify the effectiveness of our method, we compare with some benchmark models on the same dataset. They are described as follows:

**DT-Rank** [31]: This method obtains feature information by regular matching and text clustering. The clusters of samples are sorted by various statistical features.

**SVM-TS** [16]: This method uses the support vector machine (SVM) as a classification model. It utilizes time-dependent propagation features which are connected with event features as inputs of the model.

**DTC** [5]: This method uses the decision tree as the classification model. The characteristics they used are the number of microblog posts, the average number of followers, and so on.

**GRU-2** [15]: This method uses term frequency-inverse document frequency (TF-IDF) to represent text information of each period. The classification model is the Bi-GRU.

**CAMI** [29]: This method is different from GRU-2. It uses segment vectors to represent text information. The model it used is a convolutional neural network (CNN).

## 6.3 Experiment of Representation Learning Method for Network Nodes

To verify the validity of the representation learning method for network nodes proposed in this paper, we analyze the existing methods and compare the performance of them with the same training model. We extract the vectors of network nodes by representation learning methods as the training data. Then we train an extra DNN model to evaluate the effectiveness of these methods. The experimental results are shown in Table 1.

From Table 1, we can see that the Struc2vec model performs better than Deepwork and Node2vec. Struc2vec is not based on the hypothesis of neighbor similarity but the spatial structure similarity. However, it is still not as effective as our representation learning method of network nodes on the training model. Besides, the models [10, 20, 21] are not well applied to directed graphs with node attributes and edge weights. They are designed without considering the characteristic attributes of user nodes and the time attributes of events on online social networks.

**Table 1.** Experiments on representation methods of network nodes

Method	Category	Accuracy	Precision	Recall rate	F1 score
Struct2vec [21]	R	0.785	0.762	0.823	0.791
	N		0.811	0.747	0.777
Deepwalk [20]	R	0.618	0.671	0.699	0.685
	N		0.691	0.662	0.676
Node2vec [10]	R	0.723	0.702	0.766	0.733
	N		0.747	0.679	0.712
<b>Ours</b>	R	<b>0.828</b>	<b>0.802</b>	<b>0.865</b>	<b>0.833</b>
	N		<b>0.857</b>	<b>0.790</b>	<b>0.822</b>

#### 6.4 Experiment of Rumor Identification

In order to verify the effectiveness of our method, we compare with some benchmark models on the same dataset. The experimental results are shown in Table 2. From Table 2, we can see that our method perform better than these state-of-art works in rumor identification task. We also compare the effect of feedback signals, which show a difference of nearly 3% points.

The rumor identification task not only needs to have the judgment after overall propagation cycle of an event, but also needs to verify the authenticity of the information in the early stage. Accurate discrimination during the early period of an event plays an extremely important role in warning, containment, and blocking of rumors on social networks. Hence, experiments are conducted to verify if our method works in the early propagation of rumors.

The average time for the official media of Sina Weibo to refute rumors is 72 h [15, 16, 29]. Hence, we select 9 time points within 72 h and train the discriminant models respectively. The experimental results of our method are shown in Table 3. We can see that our method achieves a high accuracy in the early detection of rumors. As the propagation of events, the accuracy of rumor identification is gradually increased and it tends to be stable around the point of 24 h.

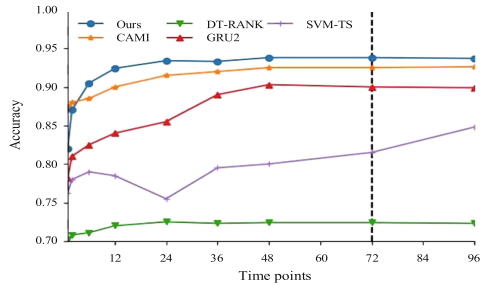
Figure 5 is a performance comparison between the proposed method and the existing methods at different time points in the early detection of rumors. The experimental results show that our method is better than others in the early detection of rumors. Even though the CAMI has a high detection rate in the initial stage of the event, but its effect is not as good as our method with the time changes.

**Table 2.** Experiments on rumor identification by different methods

Method	Category	Accuracy	Precision	Recall rate	F1 score
DT-Rank [31]	R	0.732	0.738	0.715	0.726
	N		0.726	0.749	0.737
SVM-TS [16]	R	0.857	0.839	0.885	0.861
	N		0.878	0.830	0.857
DTC [5]	R	0.831	0.847	0.815	0.831
	N		0.815	0.847	0.830
GRU-2 [15]	R	0.910	0.876	0.956	0.914
	N		0.952	0.864	0.906
CAMI [29]	R	0.933	0.921	<b>0.945</b>	0.933
	N		<b>0.945</b>	0.921	0.932
<b>Ours (except feedback)</b>	R	0.915	0.902	0.913	0.922
	N		0.887	0.907	0.936
<b>Ours (all features)</b>	R	<b>0.947</b>	0.944	0.927	0.935
	N		0.930	<b>0.945</b>	<b>0.938</b>

**Table 3.** Early detection of rumors

Deadline	Category	Accuracy	Precision	Recall rate	F1 score
1 h	R	0.821	0.816	0.823	0.820
	N		0.825	0.818	0.821
3 h	R	0.873	0.882	0.857	0.869
	N		0.863	0.887	0.875
6 h	R	0.905	0.880	0.934	0.906
	N		0.931	0.875	0.902
12 h	R	0.924	0.908	0.941	0.924
	N		0.940	0.906	0.923
24 h	R	0.934	0.928	0.939	0.933
	N		0.939	0.928	0.934
36 h	R	0.933	0.936	0.927	0.932
	N		0.929	0.937	0.933
48 h	R	0.938	0.929	0.946	0.937
	N		0.946	0.929	0.937
72 h	R	0.940	0.933	0.941	0.937
	N		0.941	0.934	0.938
96 h	R	0.938	0.953	0.917	0.935
	N		0.922	0.955	0.940



**Fig. 5.** Comparisons in the early detection of rumors

## 7 Conclusion and Future Work

In this paper, we creatively propose a new point of view based on the multiple features for rumor identification task and achieve a relatively good result. Our method also performs better in the early detection of rumors than some works. Besides, we propose a representation learning method for network nodes based on the space-time similarity. This method can also be used for other time-series tasks on social networks like public opinion monitoring and influence analysis of user. But there exists some shortcomings. Our method is slightly better than the best one for rumor identification in the propagation cycle of an event. The time for training and testing needs several days. In the future, we will do some improvements to the model for optimizing the performance of our model and reducing training time. At the same time, the loss of spatial information in network topology and the representation method of network nodes can also be optimized.

**Acknowledgement.** This work was partly supported by the National Natural Science Foundation of China under No. U1836112, the National Key R&D Program of China under No. 2016YFB0801100, the National Natural Science Foundation of China under No. 61876134 and U1536204.

## References

1. Aho, A.V., Corasick, M.J.: Efficient string matching: an aid to bibliographic search. *Commun. ACM* **18**(6), 333–340 (1975)
2. Al-Khalifa, H.S., Al-Eidan, R.M.: An experimental system for measuring the credibility of news content in Twitter. *Int. J. Web Inf. Syst.* **7**(2), 130–151 (2011)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: *Proceedings of the International Conference on Learning Representations* (2015)
4. Bemdt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: *Proceedings of AAAI-94 Workshop on Knowledge Discovery in Databases*, vol. 10, pp. 359–370 (1994)



5. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on Twitter. In: Proceedings of the 20th International Conference on World Wide Web, pp. 675–684 (2011)
6. Chen, T., Wu, L., Li, X., Zhang, J., Yin, H., Wang, Y.: Call attention to rumors: deep attention based recurrent neural networks for early rumor detection. In: Ganji, M., Rashidi, L., Fung, B., Wang, C. (eds.) PAKDD 2018, pp. 40–52. Springer, Heidelberg (2018). [https://doi.org/10.1007/978-3-030-04503-6\\_4](https://doi.org/10.1007/978-3-030-04503-6_4)
7. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of the Conference on Empirical Methods on Natural Language Processing (2014)
8. DiFonzo, N., Bordia, P.: Rumor, gossip and urban legends. *Diogenes* **54**(1), 19–35 (2007)
9. Friggeri, A., Eckles, D., Adamic, L.: Rumor cascades in social networks. In: Proceedings of the International AAAI Conference on Weblogs and Social Media (2014)
10. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
11. Gupta, M., Zhao, P., Han, J.: Evaluating event credibility on Twitter. In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp. 153–164 (2012)
12. Huang, Q., Zhou, C., Wu, J., Mingwen, W.: Deep structure learning for rumor detection on Twitter. In: Proceedings of the International Joint Conference on Neural Networks (2019)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference for Learning Representations (2015)
14. Liu, X., Nourbakhsh, A., Li, Q., Fang, R.: Real-time rumor debunking on Twitter. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management (2015)
15. Ma, J., et al.: Detecting rumors from microblogs with recurrent neural networks. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, pp. 3818–3824 (2016)
16. Ma, J., Gao, W., Wei, Z., Lu, Y., Wong, K.F.: Detect rumors using time series of social context information on microblogging websites. In: Proceedings of the 24th ACM International Conference on Information and Knowledge Management, pp. 1751–1754 (2015)
17. Ma, J., Gao, W., Wong, K.F.: Detect rumors in microblog posts using propagation structure via kernel learning. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 708–717 (2017)
18. Margolin, D., Keegan, B., Hannak, A., Weber, I.: Get back! you don’t know me like that: the social mediation of fact checking interventions in Twitter conversations. In: Proceedings of the International AAAI Conference on Weblogs and Social Media (2014)
19. Matsuta, T., Uyematsu, T.: On the distance between the rumor source and its optimal estimate in a regular tree. In: arXiv preprint [arXiv:1901.03039](https://arxiv.org/abs/1901.03039) (2019)
20. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
21. Ribeiro, L.F.R., Savarese, P.H.P., Figueiredo, D.R.: struc2vec: Learning node representations from structural identity. In: ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2017)

22. Ruchansky, N., Seo, S., Liu, Y.: CSI: a hybrid deep model for fake news detection. In: Proceedings of the 26th ACM International Conference on Information and Knowledge Management (2017)
23. Song, Y., Shi, S., Li, J., Zhang, H.: Directional skip-gram: explicitly distinguishing left and right context for word embeddings. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 2, pp. 175–180 (2018)
24. Sun, S., Liu, H., He, J., Du, X.: Detecting event rumors on sina weibo automatically. In: Ishikawa, Y., Li, J., Wang, W., Zhang, R., Zhang, W. (eds.) APWeb 2013. LNCS, vol. 7808, pp. 120–131. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-37401-2\\_14](https://doi.org/10.1007/978-3-642-37401-2_14)
25. Takahashi, T., Igata, N.: Rumor detection on Twitter. In: The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence Systems. IEEE (2012)
26. Wang, S., Terano, T.: Detecting rumor patterns in streaming social media. In: Proceedings of the 2015 IEEE International Conference on Big Data (2015)
27. Wu, K., Yang, S., Zhu, K.Q.: False rumors detection on Sina Weibo by propagation structures. In: 2015 IEEE 31st International Conference on Data Engineering, pp. 651–662 (2015)
28. Yang, F., Liu, Y., Yu, X., Yang, M.: Automatic detection of rumor on Sina Weibo. In: Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, pp. 1–7 (2012)
29. Yu, F., Liu, Q., Wu, S., Wang, L., Tan, T.: A convolutional approach for misinformation identification. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, pp. 3901–3907 (2017)
30. Yuan, C., Ma, Q., Zhou, W., Han, J., Hu, S.: Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In: Proceedings of the IEEE International Conference on Data Mining (2019)
31. Zhao, Z., Resnick, P., Mei, Q.: Enquiring minds: early detection of rumors in social media from enquiry posts. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1395–1405 (2015)