CED: Credible Early Detection of Social Media Rumors

Changhe Song, Cheng Yang, Huimin Chen, Cunchao Tu, Zhiyuan Liu, Maosong Sun

Abstract—Rumors spread dramatically fast through online social media services, and people are exploring methods to detect rumors automatically. Existing methods typically learn semantic representations of all reposts to a rumor candidate for prediction. However, it is crucial to efficiently detect rumors as early as possible before they cause severe social disruption, which has not been well addressed by previous works. In this paper, we present a novel early rumor detection model, Credible Early Detection (CED). By regarding all reposts to a rumor candidate as a sequence, the proposed model will seek an early point-in-time for making a credible prediction. We conduct experiments on three real-world datasets, and the results demonstrate that our proposed model can remarkably reduce the time span for prediction by more than 85%, with better accuracy performance than all state-of-the-art baselines.

Index Terms—Rumor, Early Detection, Deep Neural Network, Social Media.

1 Introduction

Rumor is an important phenomenon in social science and has witnessed many interests of researchers in social psychology field for many decades [1], [2]. According to the explanation of wikipedia[1] and sociologists [3], a rumor usually involves some concerned public statements whose integrity cannot be quickly or ever verified.

With the rapid growth of large-scale social media platforms, such as Facebook, Twitter, and Sina Weibo, rumor is becoming a more and more serious social problem than ever before. Due to the convenience of accessing information on these social media platforms, rumors can spread explosively within a short time before contradicted or detected. The widespread of rumors in social media severely prevents people achieving reliable information and may result in enormous economic loss or public panic when some emergencies happen.

How to detect rumors at an early stage of spread is extremely critical to prevent these damages. However, it is complicated for ordinary people to distinguish rumors from massive amounts of online information, due to the limitation of professional knowledge, time or space [4]. Therefore, many news organizations and social media service providers pay great efforts to construct rumor reporting and collecting websites (e.g.,snopes[2] and factcheck[3]) or platforms (e.g., Sina Rumor Reporting Center). Nevertheless, such practice needs a lot of human effort to collect and verify rumors, and also faces issues of coverage and time delay.

To address these issues, researchers propose to detect online rumors automatically using machine learning techniques. Most existing models treat rumor detection as a binary classification task and tend to design effective features from various information sources, including text content, publisher's profiles and propagation patterns [5], [6], [7], [8], [9], [10]. However, these feature engineering-based methods are biased and time-consuming, and can not ensure the flexibility and generalization when applied to other rumor detection scenarios.

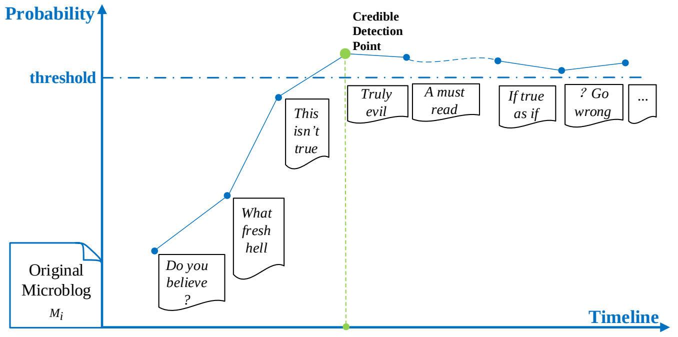


Fig. 1. An example of early rumor detection with Credible Detection Point.

Compared with feature engineering methods, the effectiveness of deep neural networks on automatic feature learning has been verified in many NLP tasks, such as parsing, text classification, machine translation and question answering. Motivated by the successful utilization of deep neural networks, Ma et al. [11] employed recurrent neural network (RNN) to learn a dynamic temporal representation for each microblog based on its reposts over time and make the prediction according to the representation of the entire repost sequence. It is the first attempt to introduce deep neural networks into repost-based rumor detection and achieves considerable performance on real-world datasets. Besides, Yu et al. [12] employed paragraph vector [13] and convolutional neural networks on the repost sequence to detect rumors.

However, to achieve early rumor detection, the model needs to make a reliable prediction as early as possible before it's widespread. It is worth pointing out that existing neural network

• Changhe Song, Huimin Chen, Cunchao Tu, Zhiyuan Liu (corresponding author), and Maosong Sun are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.
E-mail: {sch19, chm15}@mails.tsinghua.edu.cn, tucunchao@gmail.com, {liuzy, sms}@mail.tsinghua.edu.cn
• Cheng Yang is with Beijing University of Posts and Telecommunications and also with the Department of Computer Science and Technology, Tsinghua University.
E-mail: albertyang33@gmail.com

Manuscript received April 19, 2005; revised August 26, 2015.

1. https://en.wikipedia.org/wiki/Rumor
2. http://www.snopes.com/
3. http://www.factcheck.org/

based rumor detection methods can only detect rumors by considering the entire or fixed proportions of repost information, which is not effective in practice. In this article, we propose a novel model to implement online real-time detection of social media rumors.

As shown in Fig. 1, we present the temporal repost sequence of a specific rumor and the corresponding prediction probability curve. For this example, we can make a credible prediction at an early time stamp, as there appears quite a lot doubts and refutation to the original microblog. Based on this observation, we introduce "Credible Detection Point" and propose a novel early rumor detection model, Credible Early Detection (CED). Specifically, CED learns to determine the "Credible Detection Point" for each repost sequence during the training stage and ensure the credibility of the prediction result at this time point, i.e., there is no plot reversal after credible detection point. In this way, we can also make reliable detection without using the information after the "Credible Detection Point" in real applications, so that to detect rumors as soon as possible in social media platforms.

To verify the effectiveness of our model, we conduct rumor detection experiments on three representative real-world datasets, and experimental results demonstrate that our proposed model can remarkably reduce the time span for prediction by more than 85%, even with much better detection accuracy.

We make the following noteworthy contributions:

(1) To solve the problem that current rumor detection methods can only detect rumors using the entire or fixed proportions of repost information, we proposed the concept of "Credible Detection Point", making it possible to dynamically obtain the minimum amount of repost information needed to make a reliable detection during detection process for each microblog.

(2) Based on "Credible Detection Point", we propose a novel repost-based early rumor detection model, Credible Early Detection (CED). By regarding all reposts of a microblog as a sequence, CED will seek a microblog-specific "Credible Detection Point" for each microblog to make a credible prediction as early as possible.

(3) We conduct a series of experiments on three real-world datasets. Experimental results demonstrate that our model can remarkably reduce the detection time by more than 85%, with even better detection accuracy than all state-of-the-art baselines. Moreover, more than 60% of microblogs can be detected with less than 10% repost information. This performance makes real-time monitoring of rumors on social media possible.

(4) We expand the relevant Weibo rumors dataset [11] about twice the original size. We believe that the expanded dataset can better benefit further researches on rumor detection.

We will release all source codes and datasets of this work on Github [4] for further research explorations.

2 Related Work

The study of rumors can be traced back to 1940s [1]. With the rise of social networking media in recent decades, more and more methods of rumor detection have received a lot of attention. Most rumors detecting methods have been included in the comprehensive survey of Kumar et al [14].

Rumor detection in social media aims to distinguish whether a message posted on social media platforms is a rumor or not according to its relevant information, such as text content, comments, repost patterns, publisher's profiles and so on. Based on the types of information and methods used, we divide the existing rumor detection methods into three categories: (1) Traditional classification methods using artificially designed feature; (2) Deep neural networks related methods; (3) Propagation mode based methods.

2.1 Traditional Classification Methods

Most traditional proposed rumor detection models highly depend on manually designed features. These features are mainly concentrated in text content and users' information.

For example, Castillo et al. [5] design various types of features (e.g., sentence length, number of sentiment words, user's age, number of followers and friends) to evaluate the credibility of a message on specific topics. Yang et al. [6] introduced additional client-based and location-based features to identify rumors in Sina Weibo. Kwon et al. [7] employed temporal, structural and linguistic features to improve the performance of rumor detection. Liu et al. [8] proposed verification features by treating the crowd's contradictory beliefs as their debates on veracity. Ma et al. [9] integrated the temporal characteristics of topical features and sentiment features into rumor detection. Wu et al. [10] proposed to capture the high-order propagation patterns to improve rumor detection.

Most of these feature-based methods are biased, time-consuming and limited. They are usually designed for specific scenarios and hence cannot be easily generalized for other applications.

2.2 Deep Neural Networks Related Methods

To address above issues of traditional feature-based methods, researchers employed deep neural networks to learn efficient features automatically for rumor detection.

Ma et al. [11] utilized various recurrent neural networks (RNN) to model the repost sequence. Yu et al. [12] employed convolutional neural networks (CNN) on the repost sequence to capture the interactions among high-level features.

There is also a technical trend mentioned earlier that combines neural network with manual design features roundly. Bhatt et al. [15] combined the neural, statistical and external features using deep MLP. Ruchansky et al. [16] proposed a model that combines three characteristics: the text of an article, the users' response, and the source users' information promoting.

But this method of combining different information is only an increase in the types of information that can be used, not paying enough attention to early detection. Moreover, all these neural network related models can only detect rumors with the consideration of the entire or fixed proportions of repost information, which is not able to detect rumors as early as possible in practice.

2.3 Propagation Mode Related Method

Unlike the previous two kinds of methods which focus on the use of the text content information, the propagation mode related methods focus on the differences in the characteristics of real and false information transmission.

Vosoughi et al. [17] verified that there is a substantial difference in the actual transmission of false information and real information by studying a large number of Twitter data experiments. Falsehood diffused significantly farther, faster, more in-depth, and more broadly than the truth in all categories of information. Shao

4. https://github.com/

et al. [18] found that the sharing of fact-checking content typically lags that of misinformation by 10 $\sim$ 20 hours. Moreover, fake news is dominated by very active users, while fact checking is a more grassroots activity. Ma et al. [19] proposed a kernel-based method, which captures high-order patterns differentiating different types of rumors by evaluating the similarities between their propagation tree structures.

However, the rumor detection methods based on propagation model have not yet been fully developed. So far no studies have shown whether the difference in the mode of transmission in the early dissemination of rumors helps the detection of rumors. It is necessary to explore the characteristics of essential propagation modes to achieve more reliable rumor early detection.

There have been some early rumor detection methods in recent years [20], [21], [22]. Nevertheless, similar to feature-based methods, most existing models heavily depend on well-designed features or signals in repost or comment information.

Thus in this paper, inspired by the mode of propagation, we take advantage of deep neural networks on feature learning and propose our early rumor detection model CED. The effectiveness of CED has been verified through experiments on real-world datasets.

3 Methodology

In this section, we will introduce how to classify a microblog into rumors or facts according to its repost sequence. First, we will formalize the early rumor detection problem and give essential definitions. Then we will introduce basic concepts of recurrent neural networks. Finally, we will describe the details of our proposed model for early rumor detection.

3.1 Problem Formalization

Microblogs in social media platforms are usually conditioned to a certain length and thus contain limited information for rumor detection. Therefore, we utilize the relevant repost information for identification as the previous work [11] did.

Assume we have a set of microblogs as $\mathbf{M} = \{M\}$. Each original microblog message M has a relevant repost sequence, denoted as $M = \{(m_i, t_i)\}$. Here, each repost m_i represents the text content, and t_i denotes the corresponding time stamp. Generally, rumor detection aims to predict the label $y \in \{0, 1\}$ for each microblog M, according to the repost sequence $\{(m_i, t_i)\}$. Note that, we set $y = 1$ for rumors and $y = 0$ otherwise.

3.2 Repost Sequence Partition

Following the practice in [11], we need to convert original repost sequences into more informative neural network inputs because of two major reasons. On the one hand, most reposts reserve no critical information and will not benefit rumor prediction. It is very common that people repost a microblog without expressing any comments or opinions. On the other hand, a large number of microblogs own thousands of reposts, which can not be well addressed by existing neural networks and will also introduce computational efficiency issues.

Therefore, we convert the original repost sequence of a microblog by merging constant reposts into a single unit. Specifically, we batch a certain number (N) of consecutive reposts together. To ensure each batch of reposts is informative and the time granularity is small enough, we set $N = 10$ practically. After this processing, the repost sequence of $M = \{(m_i, t_i)\}$ is converted to $F = \{f_i\}$, where $f_i = \{m_{(i-1)*N+1}, \cdots, m_{i*N}\}$.

3.3 Feature Acquisition of Parted Repost Sequence

Based on the converted sequence, we need to transform the text information in each interval into a feature vector and feed this vector into neural networks. We have two ways to get feature vectors $\mathbf{F}$ for every converted repost sequence F as follows:

3.3.1 TF-IDF

Following [11], we employ the simple and efficient text representation method TF-IDF [23] for interval representation. Assuming the vocabulary size is V, we can obtain a variable-length matrix $\mathbf{F} = \{x_i\} \in \mathbb{R}^{V \times |F|}$ for each converted repost sequence F.

3.3.2 Convolutional Neural Network

Convolutional Neural Network(CNN) has been successfully applied in sentence semantic analysis [24], click-through rate prediction [25], text classification [26] and reinforcement learning tasks [27], which is made up of stacked convolutional and pooling layers, the architectures of which help model significant semantic features and achieve much improvement in respective fields. CNN is usually trained through stochastic gradient descent (SGD), with back-propagation to compute gradients.

To get the feature vector of each departed repost $f_i = \{w_1, \ldots, w_k\}$ of converted repost sequence $F = \{f_i\}$, we first represent each word w_i with its word embeddings vector $\mathbf{w}_i \in \mathbb{R}^d$, where d is the dimension of word embeddings. Then each departed repost f_i of length n (padded when necessary) is represented as instance matrix:

$$\mathbf{w}_{1:n} = \mathbf{w}_1 \oplus \mathbf{w}_2 \oplus \cdots \oplus \mathbf{w}_n \tag{1}$$

where $\oplus$ is the concatenation operator, so $\mathbf{w}_{1:n} \in \mathbb{R}^{d \times n}$. A convolutional layer is obtained by convolution operations of a weight matrix $G \in \mathbb{R}^{d \times h}$ in a row-wise way, where h is the length of words window applied to produce a new feature. Followed by a nonlinearity function $ReLU$ [28] applied to the convolution result, an element of a feature map can be obtained as:

$$c_i = ReLU(< G, \mathbf{w}_{i:i+h-1} >_{\mathcal{F}} + b) \tag{2}$$

note that $c_i \in \mathbb{R}^{n-h+1}$, where $\mathbf{w}_{i:i+h-1}$ is the i-th to $(i+h-1)$-th columns of $\mathbf{w}_{1:n}$, b is bias term and the subscript $\mathcal{F}$ is the Frobenius inner product, i.e., the summation of products of corresponding elements of both matrices. We then apply a max-pooling operation over the feature map c and take the maximum value $\hat{c} = max(c)$ as the most significant feature corresponding to the weight matrix G. Now we have described the process of how the feature is extracted from one filter now.

The CNN model uses multiple filters (with varying window sizes and different values in weight matrix) to obtain multiple features. Assume that the length of the feature vector obtained from CNN for each departed repost f_i is E, we can get a variable-length matrix $\mathbf{F} = \{x_i\} \in \mathbb{R}^{E \times |F|}$ for each converted repost sequence F. Moreover, sometimes the above-mentioned layer could be repeated to yield deeper layers to attain high-level interactions features. These features can be passed to a fully connected softmax layer whose output is the probability distribution over labels, but in our model feature matrix, $\mathbf{F}$ of repost sequence will be sent into RNN as input.

3.4 Rumor Detection with RNNs

Recurrent neural networks (RNNs) are a typical class of feed-forward neural networks for sequence modeling and have achieved great success in natural language processing tasks, such as text classification and machine translation. Generally, RNNs deal with variable-length sequences through a recurrent unit. For the i-th step, recurrent unit updates its hidden state h_i based on previous hidden state h_{i-1} and current input x_i.

Specifically, Cho et al. [29] propose Gated Recurrent Unit (GRU), which involves two gates, i.e., the reset gate r and update gate z. The calculation of the hidden state in GRU is as follows:

$$\begin{aligned} r_i &= \sigma(U_R \cdot x_i + W_R \cdot h_{i-1}), \\ z_i &= \sigma(U_Z \cdot x_i + W_Z \cdot h_{i-1}), \\ \widetilde{h}_i &= \tanh(U_H \cdot x_i + W_H \cdot (h_{i-1} \odot r_i)), \\ h_i &= (1 - z_i) \cdot h_{i-1} + z_i \cdot \widetilde{h}_i. \end{aligned} \tag{3}$$

Here, $\odot$ indicates the element-wise multiplication. U and W are the weight matrices.

After feeding the feature matrix $\mathbf{F}$ of a repost sequence into various RNNs, we can get every hidden state of each interval and take the final hidden state $h_{|F|}$ as the representative vector of repost sequence.

The objective function of rumor detection is the log-likelihood of predicting the label $y \in \{0, 1\}$ given the representation vector as follows:

$$\mathcal{O}(|F|) = \log p(y|h_{|F|}), \tag{4}$$

where $p(1|h_i) = \sigma(h_i \cdot s)$ and $p(0|h_i) = 1 - p(1|h_i)$. Here, s is weight vector and will be learnt during training.

3.5 Early Rumor Detection

Early rumor detection aims to make a credible prediction for a microblog at an advanced time point, while existing rumor detection methods can only predict considering the entire repost information.

3.5.1 *Credible Early Detection (CED)*

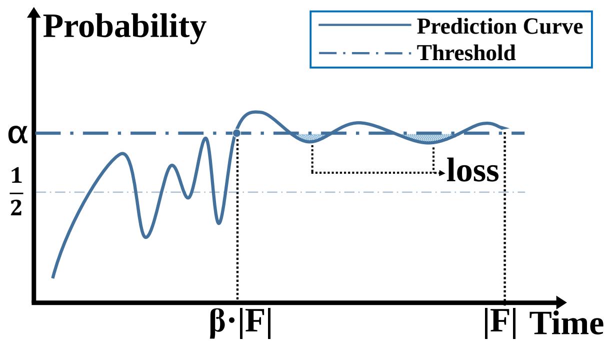


Fig. 2. An illustration of the prediction curve. Credible Detection Point shows at $\beta|\mathbf{F}|$ time point.

According to our observation over the repost information, we assume that there exists a "Credible Detection Point" for each microblog. As illustrated in Fig. 2, before this time point, there usually exist conflict reposts that support or oppose the original microblog, which will confuse people. During this period, the rumors are difficult to distinguish for both people and rumor detection models. The predictions of rumor detection models are unstable which may disturb the performance of early rumor detection models. However, after this time stamp, the debate about the original microblog reaches an agreement, and the following prediction should be stable and credible.

Based on this assumption, we introduce a parameter $\beta \in [0, 1]$ for each microblog to represent the "Credible Detection Point". This parameter β is determined by the time point when the prediction threshold is reached for the first time:

$$\beta = \frac{n_f}{|F|} \tag{5}$$

where n_f is the time point when prediction probability reaches the threshold α, i.e., $p(1|h_{n_f}) \geq \alpha$ or $p(0|h_{n_f}) \geq \alpha$ for the first time.

Thus, the first part of our objective aims to maximize the prediction probability after this time point as follows:

$$\mathcal{O}_{pred} = \frac{1}{(1-\beta)|F|} \sum_{\beta|F| \leq i \leq |F|} \mathcal{O}(i). \tag{6}$$

To ensure the property of early detection, the second part of our objective aims to minimize the prediction time β as follows:

$$\mathcal{O}_{time} = -\log \beta. \tag{7}$$

Besides, we also want to guarantee the reliability of the detection point. In other words, the prediction probabilities after this detection point should be stable and scale out the threshold. As shown in Fig. 2, for $y = 1$, we aim to minimize the difference value under the upper threshold, and for $y = 0$, we want to minimize the difference exceed the lower threshold. The objective function is as follows:

$$\begin{aligned} \mathcal{O}_{diff} = -\frac{1}{(1-\beta)|F|} \sum_{\beta|F| \leq i \leq |F|} (y \cdot \max(0, \log \alpha \\ -\mathcal{O}(i)) + (1-y) \cdot \max(0, \mathcal{O}(i) - \log(1-\alpha))). \end{aligned} \tag{8}$$

At last, we introduce two hyper-parameters λ_0 and λ_1 to combine these three objectives as follows:

$$\mathcal{O}_{CED} = \mathcal{O}_{pred} + \lambda_0 \cdot \mathcal{O}_{diff} + \lambda_1 \cdot \mathcal{O}_{time}. \tag{9}$$

3.5.2 *CED with Original Microblog*

Original microblog messages play an important role in rumor detection, which is usually regarded as a specific repost by existing repost-based rumor detection models. In this part, we extend CED to integrate original microblog information in a simple way, denoted as **CED-OM**.

In CED-OM, we apply Convolutional Neural Network(CNN) with commonly used architecture as in [26] to get the feature vector of original microblogs. Specifically, for each original microblog $\mathbf{m} = \{w_1, \ldots, w_k\}$, we first convert each word w_i into its word embeddings $\mathbf{w}_i \in \mathbb{R}^d$, where d is the dimension of word embeddings. Afterwards, we apply convolution operation to the i-th sliding window of length h through $\mathbf{c}_i = G \cdot \mathbf{w}_{i:i+h-1} + b$, where G and b are convolution matrix and bias vector respectively. At last, we apply a max-pooling operation over $\mathbf{c}_i$ to obtain the feature r. Then we get the final feature vector $\mathbf{r}$ by controlling the number of convolution operations with varying window sizes h and different values in weight matrix.

We simply concatenate the feature vector $\mathbf{r}$ with each hidden state h_i to get the final hidden state as $\hat{h}_i = h_i \oplus \mathbf{r}$ getting larger vector dimension, which is further used to replace $\mathcal{O}(i) = \log p(y|h_i)$ in Eq. 9. Note that, we employ a simple way

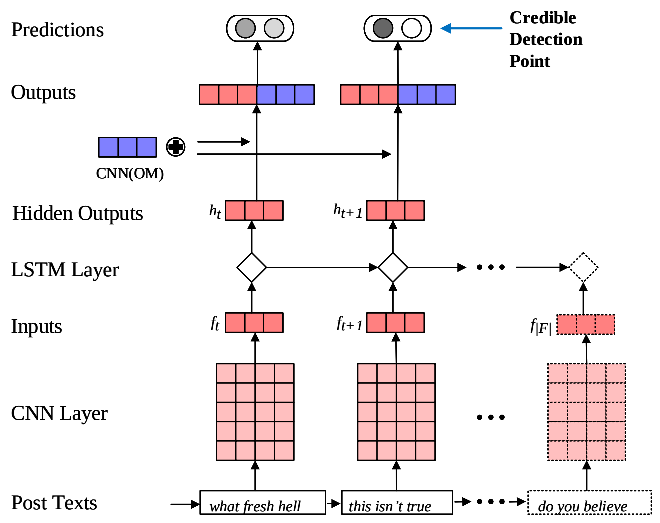


Fig. 3. Model structure of CED-CNN. $CNN(\cdot)$ refers to the CNN method to obtain feature vector as mentioned in section 3.3. Based on this, $CNN(OM)$ means the feature vector got from original microblog. As the repost information is continuously sent to the model in chronological order, the prediction probability gap is getting larger. Once Credible Detection Point is obtained, the subsequent information is no longer needed, which is indicated by dotted lines.

to incorporate original microblog into rumor detection. More complex and effective approaches, such as attention mechanism [30], [31], can be introduced to enhance the microblog representations, which will be explored in future work.

To optimize the objective in Eq. 9, we employ an efficient optimization algorithm Adam [32]. Note that, the microblog-specific parameter β will also be determined by Eq. 5 during training.

3.5.3 ***CED using CNN to deal with reposts***

On the basis of the previous method using original microblog CED-OM, we proposed a more effective method to handle the reposts sequence as mentioned in section 3.3, that is, using CNN instead of TF-IDF. In this part, we propose **CED-CNN** to obtain feature vectors of repost sequence more efficiently, achieving much more timely **Early detection**.

Analogous to CED, we segment the reposts sequence. At the same time, we set the maximum length L of the repost sequence $M = \{(m_i, t_i)\}$ by the observing statistical result. For each microblog, repost sequence gets the same length L through padding, while real length $|F| \leq L$. Each repost $m_i = \{w_1, \dots, w_k\}$ of every microblog consists of indefinite-length words. We need to convert each word w_i into word vector and pad each repost into same sentence length. Then we get word vector representation $\mathbf{w}_{1:n}$ of each repost, where n is the padding length of the repost sentence.

$$\mathbf{w}_{1:n} = \mathbf{w}_1 \oplus \mathbf{w}_2 \oplus \cdots \oplus \mathbf{w}_n \tag{10}$$

We note the CNN method to obtain feature vectors of reposts sequence, which is mentioned in section 3.3, as $CNN(\cdot)$, the final output feature map is obtained as fellows:

$$\mathbf{F} = \{x_i\} = CNN(\mathbf{w}_{1:n}) \tag{11}$$

where the length of the feature vector obtained from CNN for each departed repost f_i is E, and $\mathbf{F} \in \mathbb{R}^{E \times |F|}$ for each converted repost sequence F.

Similar to CED and CED-OM, we send this feature sequence $\mathbf{F}$ to RNN. At the same time, we apply CNN with commonly used architecture to get the feature vector of original microblog, noted as $\mathbf{r}$. Then We concatenate the feature vector $\mathbf{r}$ with each hidden state h_i to get the final hidden state as $\hat{h}_i = h_i \oplus \mathbf{r}$, which is further used to replace $\mathcal{O}(i) = \log p(y|h_i)$ in Eq. 9.

To sum up, we hire CNN to obtain feature vector of original microblog $\mathbf{r}$ and feature vectors of repost sequence $\mathbf{F}$ in the CED-CNN method, which is shown in Fig. 3. Departed repost feature map $\mathbf{F}$ is set to RNN as input, one column each time. After getting each hidden state, we use the original microblog feature vector $\mathbf{r}$ to expand the dimension of each hidden state and get prediction probability, where the objective function is same as that in Eq. 9. The early detection point is determined also as shown in Eq. 5. During the training process, we check the prediction probabilities at every step of report sequence. When it reaches the threshold for the first time, we get parameter β of current repost sequence and calculate the objective function based on β. It is worth pointing out that the parameter β represents the "Credible Detection Point" during testing.

3.5.4 *Detection Strategy for Testing*

To realize the early detection of rumors during testing, we propose a threshold based rumor detection strategy. Sequentially, we calculate the prediction probabilities $p_i = p(1|h_i)$ at each step of the repost sequence and make a prediction as follows:

$$\widetilde{y} = \begin{cases} 1, & \text{if } p_i \geq \alpha, \\ 0, & \text{if } p_i < 1 - \alpha, \\ \phi, & \text{otherwise.} \end{cases} \tag{12}$$

Here, $\alpha \in [0.5, 1]$ is a pre-defined threshold as in Eq. 8. if $\widetilde{y} \in \{0, 1\}$, it means the prediction probability reaches the threshold and we can make a credible prediction only using i intervals rather than $|F|$. If $\widetilde{y} = \phi$, we will look for the prediction result of next step.

4 Experiments

To verify the effectiveness of our proposed early detection models, we perform rumor detection experiments on three representative real-world datasets including Weibo and Twitter. Besides, we also conduct detailed analysis on detection accuracy, early detection, and parameter sensitivity.

4.1 Datasets

For evaluation, we employ two standard real-world rumor datasets from Sina Weibo[5] and Twitter[6] as in [12], which are separately labeled as "Weibo-stan" and "Twitter". We get the microblog ID lists of the two datasets and collect all the repost information of each microblog. Note that, some microblogs are unavailable as they are deleted or their belonging accounts are closed.

Also, in order to obtain a larger dataset to verify the effectiveness of our model, we also model on Ma et al. [11] method to crawl more Weibo data to get a larger dataset, which is noted as "Weibo-all". For more Weibo data, we obtain a set of known rumors from the Sina community management center[7], which reports various misinformation. The Weibo API can capture the

5. https://www.weibo.com/
6. https://twitter.com
7. http://service.account.weibo.com

TABLE 1
Statistics of the datasets. (Here, *Rep.* indicates reposts.)

Weibo-all	Rumors	Non-rumors	All
Numbers	3,851	4,199	8,050
Reposts	2,572,047	2,450,821	5,022,868
Min. Rep.	3	5	3
Max Rep.	59,317	52,156	59,317
Ave. Rep.	668	584	624
Weibo-stan	**Rumors**	**Non-rumors**	**All**
Numbers	2,313	2,350	4,663
Reposts	2,088,430	1,659,258	3,747,688
Min. Rep.	3	5	3
Max Rep.	59,317	52,156	59,317
Ave. Rep.	903	706	804
Twitter	**Rumors**	**Non-rumors**	**All**
Numbers	495	493	988
Reposts	148,594	397,373	545,967
Min. Rep.	4	5	4
Max Rep.	7,071	37,087	37,087
Ave. Rep.	300	806	553

original messages and all their repost/reply messages given. We also gather a similar number of non-rumor events by crawling the posts of general threads that are not reported as rumors on Sina community management center. The detailed statistics of these three datasets are listed in Table 1.

4.2 Baselines

To detect rumors according to the repost information, we employ four representative baselines as follows:

CNN-OM [26] CNN based models have achieved promising results in text classification tasks. Here, we employ CNN in [26] to deal with just the original microblogs for rumor classification.

TF-IDF [23] We simply merge all the text information of the repost sequence into the bag of words and calculate the TF-IDF representation vector of this sequence. With these TF-IDF representations, we train an SVM [33] classifier for rumor classification.

GRU-2 [11] Ma et al. propose to utilize RNNs to learn representations of repost sequences. Here, we follow this idea and employ a 2-layer GRU [29] to train rumor classifiers.

CAMI [12] Yu et al. 2017 employ paragraph vector to represent each repost internal and utilize CNN to capture high-level interactions for misinformation identification[8].

4.3 Evaluation Metrics and Parameter Settings

To evaluate the performance of various methods on rumor detection, we adopt *accuracy* (Acc.) metric as follows:

$$accuracy = \frac{a}{T}, \tag{13}$$

where a is the number of correctly predicted rumors and non-rumors in testing set, and T is the size of the testing set. Meanwhile, we also evaluate the performance of these models using macro *precision*, *recall* and *F-measure* metrics.

8. As the source code of this work is not available, we reproduce this model with the help of the authors.

Besides, we also want to demonstrate the effectiveness of our proposed early detection model. Thus, we propose *Early Rate* (ER) to evaluate the utilization ratio of repost information as follows:

$$ER = \frac{1}{T} \sum_{i \in Testing} \frac{t_i}{|F_i|}, \tag{14}$$

where t_i indicates that the model makes a prediction at t_i time stamp for the i-th microblog, i.e, prediction probability reaches the threshold α for the first time, and $|F_i|$ is the total length of intervals of the i-th repost sequence. Lower *ER* value means the model can detect rumors earlier, with less repost information used.

Because the performance improvement is not obvious when N is less than 10, which will also introduce computational efficiency issues, we select 10% instances as the validation dataset. Then following the settings in [11], we randomly split the rest for training and testing with a ratio of $3 : 1$ in all three datasets (Weibo-stan, Weibo-all and Twitter). We set the dimension of TF-IDF vector to 1000, the vocabulary size of original microblogs to $3,000$ when using CNN-OM and CED-CNN and the collection vocabulary size of original microblogs and repost sequences to $20,000$.

We show the results of CED under two settings of prediction threshold α, including 0.875 and 0.975. The weights λ_0 and λ_1 are set to 0.01 and 0.2 respectively. Besides, we employ the same neural network settings as [11] and utilize a 2-layer GRU as sequence encoder.

For a fair comparison, we set the hidden size to 200 for CED, GRU-2 and CAMI, and 100 for CED-OM and CED-CNN. For CNN part of original microblog in CED-OM and CED-CNN, we set the filter widths to $[4, 5]$, with each filter size to 50. As a result, the final hidden state in CED-OM and CED-CNN is 200 as well.

4.4 Results and Analysis

As shown in Table 2 and Table 3, we list the detailed results of different methods under various evaluation metrics. We also bold the best result of each column in both tables. Note that, the original microblogs in the Twitter dataset are not released by [11], [12]. Therefore, we omit the results of CNN-OM and CED-OM on this dataset. In addition, baselines' Early Rate results are all N/A because we send all reposts data as input to the baseline methods, which can not automatically use part of the data. From the experimental results, we have the following observations:

(1) Our proposed early rumor detection models: CED, CED-OM, and CED-CNN, achieve significant and consistent improvements compared with all baseline methods, with higher accuracy and even much less repost sequence information. It demonstrates the reasonability and effectiveness of our proposed early detection models and strategy.

(2) Comparing with all the baselines, our model can considerably reduce the detection time by around 86% in Weibo and 77% in Twitter, with even better performance on accuracy. The reason is that CED aims to make the representation of each single repost intervals predictable. On the contrary, other methods only make the final representation predictable. Such practice makes our model capable of detecting early with partial repost information. Especially, CED-CNN achieves a very high detection accuracy using only 13.2% of the repost sequence information, making it possible to detect rumors in real-time on social media.

(3) By incorporating original microblogs, CED-OM achieves significant improvement on early rate under both thresholds. It is

TABLE 2
Weibo experimental results. (0.875 and 0.975 are prediction threshold of CED, CED-OM and CED-CNN.)

	Weibo-stan					Weibo-all				
Methods	Acc.	Precision	Recall	F1	ER	Acc.	Precision	Recall	F1	ER
CNN-OM	0.809	0.829	0.781	0.804	N/A	0.887	0.883	0.884	0.883	N/A
TF-IDF	0.859	0.799	0.951	0.868	N/A	0.819	0.914	0.678	0.779	N/A
GRU-2	0.920	0.926	0.900	0.913	N/A	0.906	0.923	0.878	0.901	N/A
CAMI	0.896	0.890	0.876	0.883	N/A	0.893	0.866	0.914	0.889	N/A
CED(0.875)	0.938	0.930	0.946	0.938	32.7%	0.913	0.926	0.892	0.908	31.3%
CED(0.975)	**0.946**	**0.946**	0.944	**0.945**	41.4%	0.921	0.934	0.899	0.916	45.6%
CED-OM(0.875)	0.916	0.896	0.941	0.918	28.1%	0.920	0.930	0.902	0.916	19.6%
CED-OM(0.975)	0.942	0.923	**0.964**	**0.943**	32.0%	0.913	0.942	0.874	0.906	19.5%
CED-CNN(0.875)	0.900	0.920	0.889	0.904	**15.1%**	0.941	0.947	0.929	0.938	**13.2%**
CED-CNN(0.975)	0.912	0.920	0.912	0.916	19.3%	**0.947**	**0.954**	**0.934**	**0.944**	17.9%

TABLE 3
Twitter experimental results.

Methods	Acc.	Precision	Recall	F1	ER
TF-IDF	0.587	0.569	0.941	0.710	N/A
GRU-2	0.672	0.626	0.779	0.694	N/A
CAMI	0.595	0.667	0.525	0.588	N/A
CED(0.875)	0.717	0.692	0.733	0.712	53.0%
CED(0.975)	**0.744**	**0.708**	0.791	0.747	52.5%
CED-CNN(0.875)	0.721	0.642	0.929	0.760	**32.1%**
CED-CNN(0.975)	0.704	0.616	**1.000**	**0.762**	43.8%

mainly because that original microblogs can provide additional information when there are only a few of reposts. The lack of reposts raises difficulties for CED to detect rumors, which can be addressed by considering original microblogs to some extent.

(4) The Early Rate of CED-CNN method achieves the best performance in all three data sets. At the same time, it also has excellent performance in Accuracy, especially after being fully trained in a large dataset like Weibo-all. Compared with CED, CED-CNN has better performance in Early Rate performance, which shows that the method using CNN to extract the characteristics of forwarding sequence information is more efficient than TF-IDF.

(5) The detection threshold α plays an important role in balancing the detection accuracy and early rate. As shown in Table 2, a higher threshold means that our model will give more confident prediction result, while delaying the detection time and resulting in a higher early rate. The threshold provides a flexible selection according to the actual scenarios.

Observations above demonstrate that our model is able to make a more accurate prediction with limited repost information. It is robust and flexible to various datasets and parameter settings.

In the following sections, We have made a comprehensive performance analysis of the proposed model. Because the number of samples in the Twitter dataset is too small, the experiments are conducted mainly in the Weibo dataset.

4.5 Early Detection.

In CED, the "Credible Detection Point" for each microblog is determined by Eq. 5 and constantly ahead during training and inferred during testing based on threshold-based detection strategy. In other words, CED can learn and infer an appropriate detection point for various microblogs. As β keeps changing during the optimization and is different for each sample, we plot the mean value curve of β (which is ER) in the Weibo verification dataset as the training step increases in Fig.4. The decrease of ER during training shows that the objective function of our CED-CNN model converges well and fast.

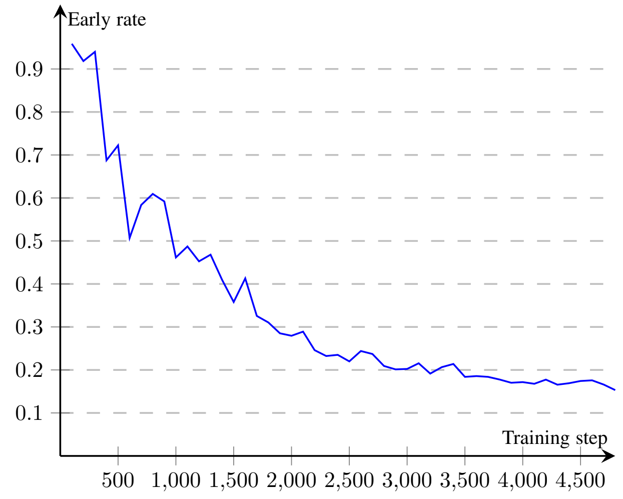


Fig. 4. Convergence curve of β during training. Early rate actually reflects the average β value in the verification dataset.(Batch size=8)

In Fig. 5, we show the distribution of "Credible Detection Point" in testing set. In order to reflect the specific performance of different models in different datasets, we have separately calculated the detection point distribution of the three methods (CED, CED-OM, CED-CNN) in Weibo-all datasets. From this figure, we find that:

(1) About 30% microblogs can be detected with less than 10% repost information. More than $40/60\%$ microblogs can be detected with less than $10/20\%$ repost information respectively using CED-OM. It reflects the applicability of our model to handle different rumors.

(2) With the consideration of original information and repost sequence information using CNN, CED-CNN can detect more rumors than both CED and CED-OM with 10% repost information, that is, more than 60% of samples in Weibo-all dataset. This verifies the advantage of incorporating original microblogs and CNN-forwarding-processing method again.

(3) A peak occurs when using the whole repost sequence to make detections (shown as using 100% repost information). It can be seen that this situation occurs much less in CED-CNN, which is less than 10%, indicating that CED-CNN needs less repost information to make credible detection and has higher utilization of forwarding information.

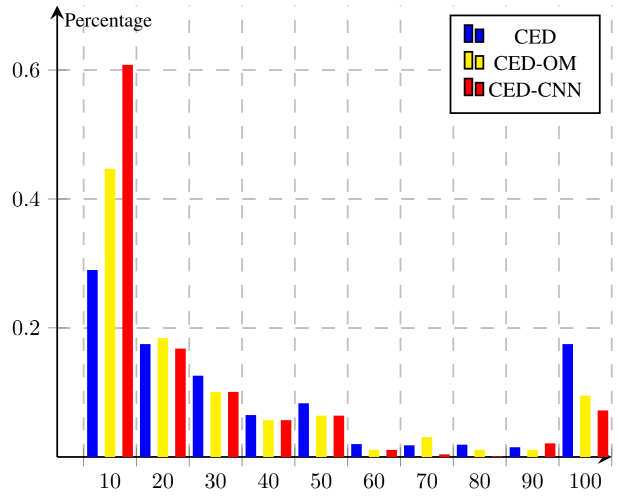


Fig. 5. Early rate distribution in Weibo-all dataset. The abscissa represents the percentage of repost sequence information used to reach Credible Detection Point(%).

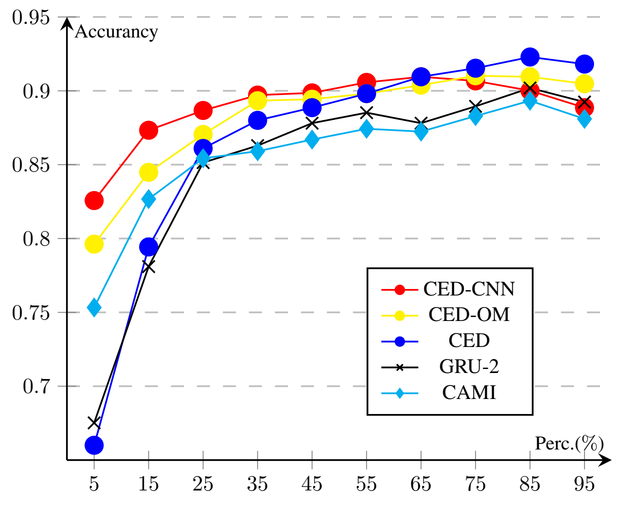


Fig. 6. Model comparison using different fixed percentages of repost information in Weibo. (Here, Perc. means percentage of used repost information)

In order to verify the performance of various methods with limited repost information, we set different percentages of repost information and compare our models with two best-performed baselines, including GRU-2 and CAMI. Note that we do not only just limit the testing repost information, but also limit the repost information for training. As shown in Fig. 6, we observe that: CED-OM and CED-CNN achieve better performance when the percentage is less than 55%, and CED performs best when the percentage is more than 55%. This also proves the necessity of original microblog information and CNN-forwarding-processing method in the early stage.

It is worth pointing out that our CED method can tell the difference of repost information every microblog needs for credible detection, which is more reasonable than a fixed percentage for all microblogs. So this fixed percentage of repost information experiment is actually not suitable for our method. This also explains the phenomena that CED-OM and CED-CNN performs more poorly as applied fixed percentage arises.

4.6 Ablation Study

In section 3.5, we explain the idea of “Credible Detection Point” applied to objective functions. To verify the role of various parts in the objective function, we perform experiments using different objective functions.

Taking $\mathcal{O}_{time}$ as an example, in order to explore the role of $\mathcal{O}_{time}$ in the model, we design an objective function that lacks $\mathcal{O}_{time}$:

$$\mathcal{O}_1 = \mathcal{O}_{pred} + \lambda_0 \cdot \mathcal{O}_{diff}. \tag{15}$$

We compare this result using $\mathcal{O}_1$ with the result of using $\mathcal{O}_{CED}$, which contains the whole objective function parts as Eq. 9. In this way, we can learn about the contribution of $\mathcal{O}_{time}$ to the overall approach.

Similarly, we set up another objective function $\mathcal{O}_2$ to verify the effectiveness of $\mathcal{O}_{diff}$:

$$\mathcal{O}_2 = \mathcal{O}_{pred} + \lambda_1 \cdot \mathcal{O}_{time}. \tag{16}$$

We choose CED-CNN method and Weibo-all dataset to finish this experiment. The relevant results are shown in Table 4.

TABLE 4
Experimental results of ablation studies on Weibo-stan dataset.

Methods	Acc.	Precision	Recall	F1	ER
O1-CED-CNN(0.975)	0.949	0.961	0.930	0.946	21.1%
O2-CED-CNN(0.975)	0.934	0.916	0.948	0.932	20.8%
CED-CNN(0.975)	0.947	0.954	0.934	0.944	17.9%

By comparing the results of $\mathcal{O}_1$ and $\mathcal{O}_2$ experiments with the overall results $\mathcal{O}_{CED}$, we get the following conclusions:

(1) Compared with the CED-CNN method we proposed, method $\mathcal{O}_2$ gets almost the same Accuracy as CED-CNN, but Early Rate decreases by 3%, indicating that $\mathcal{O}_{time}$ can enhance model early detection performance without loss of Accuracy.

(2) The Accuracy of $\mathcal{O}_2$ method is reduced significantly, and there is also a loss in Early Rate performance, which is compared to the CED-CNN method. This shows that $\mathcal{O}_{diff}$ plays an important role both in promoting Early Rate and Accuracy.

The above experiments effectively validate the specific role of $\mathcal{O}_{time}$ and $\mathcal{O}_{diff}$ in model training, both of which reflect that the purpose of the design is effectively achieved.

TABLE 5
Case study of ablation studies on Weibo. We bold the prediction probability when it reaches CDP. $\mathcal{O}_1$ actually gets the wrong prediction result.

No.	Repost Translation	$\mathcal{O}_1$	$\mathcal{O}_2$	$\mathcal{O}_{CED}$
1	'Ridiculous','Really'	0.6734	0.9661	**0.9777**
2	'Bye','Prophecy'	0.3340	0.9509	0.9788
3	'My God','False'	0.9220	**0.9948**	0.9873
4	'Li Bai','Dizziness'	0.9222	0.9942	0.9874
5	'Poetry','Ha ha ha'	0.9141	0.9944	0.9880
6	'Too false','Prophet'	0.9218	0.9951	0.9882
7	'Really fake','Believe'	0.9253	0.9953	0.9884
......	......	......	......	......
90	'Really','Vulgar'	0.1027	0.8062	0.9820
91	'God','Respect'	**0.0248**	0.5816	0.9823
92	'Really fake','Intensely'	0.0815	0.8732	0.9828
93	'Funny,'Mighty'	0.0806	0.8572	0.9825

Especially, the designed purpose of $\mathcal{O}_{diff}$ is to minimize the difference value under the upper threshold of rumors, and to minimize the difference exceed the lower threshold of non-rumors after Credible Detection Point(CDP), so that the CDP of the model will advance overall. This explains why the Early Rate performance of $\mathcal{O}_2$ is weaker than that of the CED-CNN. Besides, $\mathcal{O}_{diff}$ guarantee the credibility of the detection point. In other words, the prediction probabilities after this detection point should be stable and scale out the threshold, which explains why the CED-CNN's accuracy is significantly higher than that of $\mathcal{O}_2$.

In addition, we show a case study for ablation test. We train CED-CNN without $\mathcal{O}_{diff}$ and without $\mathcal{O}_{time}$, respectively, and compared these with the original loss function trained model, while result shown in Table 5. This rumor case study suggests that without $\mathcal{O}_{diff}$, the prediction probability curve would fluctuate, especially after reaching CDP. For example, using the model trained with $\mathcal{O}_2$, the prediction probability decreases after reaching CDP(which is 0.9948). While without O-time, the number of reposts required to reach CDP increases significantly, and the prediction curve becomes more unstable, and there may be flipping leading to prediction errors.

From this conclusion analysis, we confirm the validity of introducing "Credible Detection Point" thought into objective functions.

4.7 Parameter Sensitivity

There are three critical hyper-parameters in CED, CED-OM, and CED-CNN, i.e., α, λ_0 and λ_1. To verify the stability of our model, we investigate the parameter sensitivities here. In Fig. 7, we show the accuracy and early rate results of CED (left three) and CED-CNN (right three) under various parameter settings in Weibo-stan dataset, which is smaller to show parameter sensitivity more intuitively. From this figure, we observe that:

(1) The hyper-parameter α controls the threshold for prediction. From the top two figures in Fig. 7, we observe that increasing α can benefit prediction accuracy but lengthen the detection time. Therefore, we need to find a tradeoff between accuracy and early rate according to the real-world scenarios. Based on our experience, when $\alpha = 0.975$, our model performs better than baselines and reduces the detection time by about 87%.

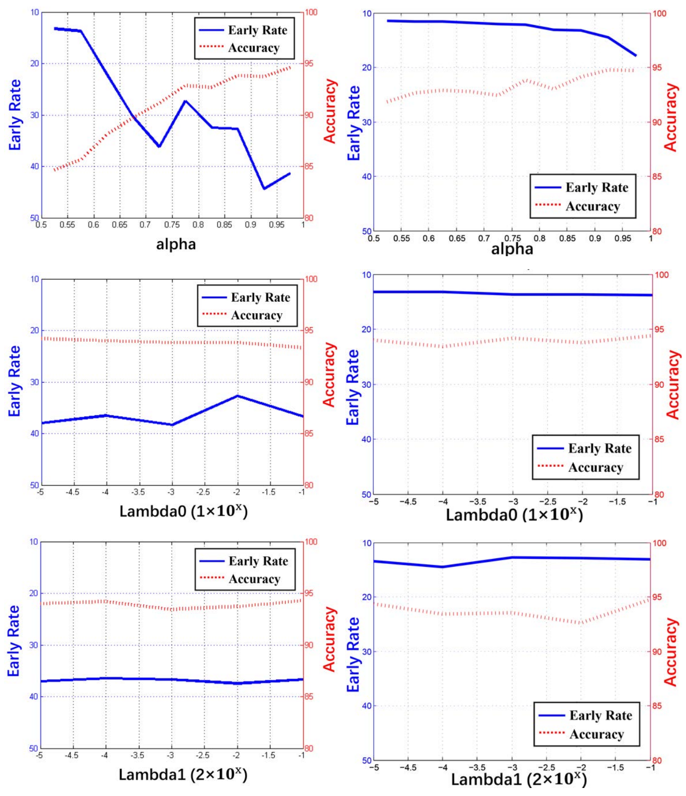


Fig. 7. Parameter Sensitivities of CED (top: α; middle: λ_0; below: λ_1; left three figures: CED, right three figures: CED-CNN.)

(2) In CED-CNN, the hyper-parameter α shows consistently good results and excellent stability. This shows that the threshold setting, which directly determines the judgment of early detection point, has an insufficient effect on the performance of the model. CED-CNN perform well in both Early Rate and Accuracy with different α, so it is very reliable when balancing Early Rate and Accuracy performance.

(3) The hyper-parameter λ_0 controls the weight of the difference loss, and the hyper-parameter λ_1 controls the weight of the prediction time as in Eq. 9. From the middle and below figures in Fig. 7, we find that CED has a stable performance when λ_0 and λ_1 range in a large scope (from 10^{-5} to 10^{-1}), for both CED and CED-CNN. It demonstrates the stability and robustness of CED and CED-CNN method.

All the observations above indicates that our model is flexible to different parameter settings and can be easily trained in practice. In addition, we see that the stability of CED-CNN is very excellent and it is much more stable than CED.

4.8 Cold-Start Experiment

Previous experiments have shown that our method is designed to work well when similar rumors have been employed in training set. However, many times the rumors are not correlated to historical rumors, which is called cold-start cases.

If words in repost content do not or rarely appear in the training set, we consider it as a cold-start case, because our model mainly uses repost information to detect rumors. We calculate the word repetition number of each sample compared with all the remaining

TABLE 6
Selected case and their repost information. (CDP indicates Credible Detection Point.)

Original Microblog: "The famous anti-fraud activist Fang Zhouzi announced that he would stop updating in Sina Weibo and move to another platform." Everyone who reposts this microblog will receive an iPhone.		
2012-08-14 13:09:18	"<u>*Good*</u>! Waiting for the iPhone!" "You will **break** your promise!" "<u>*Reposted*</u>"	0.247
2012-08-14 13:15:28	"<u>*Followed*</u>! Waiting for the phone!" "<u>*Followed*</u>!" "Will I get two phones if I repost twice?"	0.265
2012-08-14 13:29:03	"<u>*Okay*</u>∼ Waiting for your <u>*gift*</u>∼" "**Really**?" "**No one** will **believe** it!"	0.563
2012-08-14 13:40:25	"@Wenzhou Entertainment News. It's only a **micro-marketing event**!"	**0.977 (CDP)**
2012-08-14 13:45:03	"Only **word games**! Your account should be **closed** directly!"	0.958
2012-08-14 13:51:22	"<u>*Reposted*</u>! Wonder if it could be realized!'	0.917

TABLE 7
Selected cases and their repost information. (We bold the negative signals and underline the positive ones.)

Correct Case		Incorrect Case	
Repost Time	Content	Repost Time	Content
2012.11.17 09:45:30	It's always a **trap**!	2013.12.06 23:40:48	Save in address book, the latest news!
2012.11.17 12:33:43	Spreading, **vigilance**	2013.12.08 14:54:49	@Changsha City Small Animal Protection Association
2012.11.17 12:35:06	What are the reasons for so many **liars**?	2013.12.08 18:24:37	<u>Finaly</u>!
2012.11.17 16:56:18	Near the year off, all kinds of **trick** are likely to happen, be sure to be **careful**	2013.12.09 00:23:21	<u>Great</u>
2012.11.17 19:08:55	[surprise][**dizzy**]	2013.12.09 02:10:21	<u>Really</u>? <u>Really</u>? <u>good</u>!
2012.11.18 08:14:16	**False** news	2013.12.09 12:41:05	<u>Really</u> stunned
2012.11.21 00:01:36	Be **careful**!	2013.12.09 13:50:44	<u>[applaud]</u><u>[applaud]</u>
2012.11.24 14:31:26	**Not true**, but onlookers	2013.12.09 19:26:14	**Useless**
2012.12.12 22:06:21	<u>True</u> or **false**	2013.12.09 19:59:51	Remember this phone!
2013.01.24 16:13:59	I received a similar phone call, but fortunately i did **not believe**.	2013.12.09 22:06:57	It is said that the message is **false**

TABLE 8
Cold-start experimental results (CS Rate, which is short for cold-start rate, represents the word repetition rate of reposts in Weibo-all testing set).

CS Rate	Acc.	Precision	Recall	F1	ER
50%	0.931	0.913	0.933	0.923	34.6%
40%	0.923	0.930	0.904	0.917	43.5%
30%	0.942	0.939	0.902	0.920	50.2%
20%	0.946	0.952	0.872	0.910	66.4%
10%	0.939	0.966	0.864	0.912	67.5%
5%	0.940	0.950	0.868	0.907	79.2%

samples in the Weibo-all dataset. We find that the Average Word Repetition Number is 306.5. Then we select all the samples with word repetition numbers lower than $306.5 *$ Cold-Start Rate to build a new testing set. We randomly select a fixed number of training examples from the rest samples. The model we use here is CED-CNN(0.975). We find that:

(1) Taking the 5% Cold-Start Rate as an example, the size of test sets is only 338, and the accuracy rate is still above 0.90.

(2) We also found a large increase in ER, which suggests that more repost information is needed to determine whether a cold-start Weibo is a rumor because the words are unfamiliar. But note that the accuracy is guaranteed.

In conclusion, when in the face of cold start rumors, our model adaptively uses more repost information to ensure that the accuracy rate does not decrease much. We add section 4.8 to show this result in our manuscript.

4.9 Case Study and Error Analysis

To give an intuitive illustration of CED, we select a representative case from Weibo, which is correctly predicted as a rumor by CED, and show its repost list in Table 6. In this table, we bold the negative signals and underline the positive ones manually for an intuitive understanding. We also show the prediction probability of each interval in the last column.

From this table, we observe the probabilities of the first several intervals vary a lot due to the conflict information in these intervals. Then, CED makes a correct prediction at the "Credible Detection Point", and the rest probabilities are quite consistent. That conforms to the assumption of CED that the prediction curve after "Credible Prediction Point" should be stable.

For the incorrectly predicted ones, their reposts are quite unsatisfactory, which leads to difficult to find a credible prediction

TABLE 9
High frequency word statistic of Credible Detection Point.

No.	Content	Num.	No.	Content	Num.
1	Angry	398	11	Praise	109
2	Love	342	12	Onlookers	106
3	Surprise	338	13	Death	101
4	Say	257	14	Really fake	98
5	Truly	211	15	False	83
6	Laugh	210	16	Awesome	81
7	Overly	196	17	Refute a rumor	72
8	Think	147	18	Happy	63
9	Like	130	19	Truth	54
10	True	116	20	Start a rumor	52

point and make a credible prediction. We summarize three reasons for errors: (1) Too few reposts; (2) Reposts are quite conflicting; (3) Reposts are not relevant to the original microblogs.

Actually, (1) and (3) reasons can be solved as time goes by. The important microblog texts that need to be identified as rumors or not will inevitably accumulate enough comments to be used by the CED. Then to demonstrate the conflicting reposts problem of CED, we select two representative cases and show their repost lists in Table 7. From this table, we observe that CED can make a correct prediction at an early stage because the rest reposts are quite consistent. That conforms to the assumption of CED that the prediction curve after "Credible Prediction Point" should be stable. However, for the incorrect one, its reposts are quite conflicting, which result in a problem that CED cannot find a credible prediction point and make a credible prediction.

Therefore, we conclude that CED is not good at addressing the conflicting repost sequence well, which will be our future work.

4.10 Credible Detection Point Analysis

In section 3.5.1, we make a strong assumption that there exists a single "Credible Detection Point" to predict, which drives our proposed methodology and experiments. Finally in addition to our observation, we now show a data analysis of the CDP to prove the existence of this point.

In order to prove the existence of "Credible Detection Point", we extract the repost content of 1810 samples in Weibo-all testing set when model CED-CNN(0.975) reaches the Credible Detection Point in the experiments of Table 2, and count the frequency of each word. After excluding the effect of Weibo format words (for examples: "PAD", "Weibo" and "Repost") and emojis, we list the most frequently used words which lead to CDP in Table 9. The Num. means how many times this word appears in CDP content of these 1810 test samples. In addition, in order to facilitate the display, we have carried out the closest translation of the original Chinese vocabulary.

We can observe that most of these words have strong emotional attitudes and a discussion of truth and falsehood. This is very consistent with our observation over the repost information. Intuitively the original microblog can be classified with a high confidence with these words in the reposts. Thus the rest of the repost information is no longer needed after the CDP.

As proved, our model can detect these Credible Detection Points, making it possible to dynamically obtain the minimum amount of repost information needed to make a reliable rumor detection.

5 Conclusion and Future Work

In this paper, we focus on the task of early detection of social media rumors. This task aims to distinguish a rumor as early as possible based on its repost information. While existing works can only make a prediction with the entire or fixed proportions of repost sequence, we assume there exists a "Credible Detection Point" for each microblog. Moreover, we propose Credible Early Detection (CED) model to remove the effect of interference repost information and make a credible prediction at a specific credible detection point. Experimental results on real-world datasets demonstrate that our model can significantly reduce the time span for predictions by more than 85%, with even better accuracy.

For future works, we can incorporate other important information into early rumor detection, such as publisher's profiles and propagation structure besides the repost information and original microblogs. This additional information is expected to solve the conflict issue in repost sequence.

Another direction is to distinguish the critical signals or reposts from the repost sequence would be helpful for early rumor detection. We will try to utilize attention mechanism to select important reposts and improve the early rumor detection.

Acknowledgments

This work is supported by the Major Project of the National Social Science Foundation of China under Grant No. $13\&ZD190$. This research is also part of the NExT++ project, supported by the National Research Foundation, Prime Ministers Office, Singapore under its IRC@Singapore Funding Initiative.

References

[1] G. W. Allport and L. Postman, "The psychology of rumor." 1947.
[2] J.-N. Kapferer, *Rumeurs: le plus vieux média du monde*. Editions du seuil, 1987.
[3] W. A. Peterson and N. P. Gist, "Rumor and public opinion," *American Journal of Sociology*, vol. 57, no. 2, pp. 159–167, 1951.
[4] Z. Liu, L. Zhang, C. Tu, and M. Sun, "Statistical and semantic analysis of rumors in chinese social media," *Science China: Informationis*, 2015.
[5] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on twitter," in *Proceedings of WWW*, 2011, pp. 675–684.
[6] F. Yang, Y. Liu, X. Yu, and M. Yang, "Automatic detection of rumor on sina weibo," in *Proceedings of KDD*, 2012, p. 13.
[7] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in *Proceedings of ICDM*, 2013, pp. 1103–1108.
[8] X. Liu, A. Nourbakhsh, Q. Li, R. Fang, and S. Shah, "Real-time rumor debunking on twitter," in *Proceedings of CIKM*, 2015, pp. 1867–1870.
[9] J. Ma, W. Gao, Z. Wei, Y. Lu, and K.-F. Wong, "Detect rumors using time series of social context information on microblogging websites," in *Proceedings of CIKM*, 2015, pp. 1751–1754.
[10] K. Wu, S. Yang, and K. Q. Zhu, "False rumors detection on sina weibo by propagation structures," in *Proceedings of ICDE*. IEEE, 2015, pp. 651–662.
[11] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, "Detecting rumors from microblogs with recurrent neural networks." in *Proceedings of IJCAI*, 2016, pp. 3818–3824.
[12] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "A convolutional approach for misinformation identification," in *Proceedings of IJCAI*, 2017.
[13] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning*, 2014, pp. 1188–1196.
[14] S. Kumar and N. Shah, "False information on web and social media: A survey," *arXiv preprint arXiv:1804.08559*, 2018.

[15] G. Bhatt, A. Sharma, S. Sharma, A. Nagpal, B. Raman, and A. Mittal, "Combining neural, statistical and external features for fake news stance identification," in *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 2018, pp. 1353–1357.

[16] N. Ruchansky, S. Seo, and Y. Liu, "Csi: A hybrid deep model for fake news detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 797–806.

[17] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[18] C. Shao, G. L. Ciampaglia, A. Flammini, and F. Menczer, "Hoaxy: A platform for tracking online misinformation," in *Proceedings of the 25th international conference companion on world wide web*. International World Wide Web Conferences Steering Committee, 2016, pp. 745–750.

[19] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors in microblog posts using propagation structure via kernel learning," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2017, pp. 708–717.

[20] Z. Zhao, P. Resnick, and Q. Mei, "Enquiring minds: Early detection of rumors in social media from enquiry posts," in *Proceedings of WWW*, 2015, pp. 1395–1405.

[21] T. N. Nguyen, C. Li, and C. Niederée, "On early-stage debunking rumors on twitter: Leveraging the wisdom of weak learners," in *Proceedings of ICSI*, 2017, pp. 141–158.

[22] L. Wu, J. Li, X. Hu, and H. Liu, "Gleaning wisdom from the past: Early detection of emerging rumors in social media," in *Proceedings of ICDM*, 2017, pp. 99–107.

[23] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[24] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[25] Q. Liu, F. Yu, S. Wu, and L. Wang, "A convolutional click prediction model," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 1743–1746.

[26] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[27] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 2154–2162.

[28] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[29] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *Proceedings of SSST-8*, 2014.

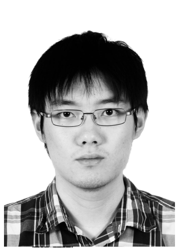
[30] C. N. dos Santos, M. Tan, B. Xiang, and B. Zhou, "Attentive pooling networks," *CoRR, abs/1602.03609*, vol. 2, no. 3, p. 4, 2016.

[31] C. Tu, H. Liu, Z. Liu, and M. Sun, "Cane: Context-aware network embedding for relation modeling," in *Proceedings of ACL*, 2017.

[32] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of ICLR*, 2015.

[33] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

Changhe Song is a master student of the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China. His research interests are natural language processing, social computation, text-to-speech and voice conversion.



Cheng Yang is an assistant professor in Beijing University of Posts and Telecommunications. He received his B.E. degree and Ph.D. degree from Tsinghua University in 2014 and 2019, respectively. His research interests include natural language processing and network representation learning. He has published several top-level papers in international journals and conferences including ACM TOIS, EMNLP, IJCAI and AAAI.

Huimin Chen Huimin Chen is a PhD student of the Department of Computer Science and Technology, Tsinghua University. Her research interests include natural language processing and social computing. She has published several papers in international conferences including ACL, EMNLP, IJCAI.

Cunchao Tu is a PhD student of the Department of Computer Science and Technology, Tsinghua University. He got his BEng degree in 2013 from the Department of Computer Science and Technology, Tsinghua University. His research interests are natural language processing and social computing. He has published several papers in international conferences and journals including ACL, IJCAI, EMNLP, COLING, and ACM TIST.

Zhiyuan Liu is an associate professor of the Department of Computer Science and Technology, Tsinghua University. He got his BEng degree in 2006 and his Ph.D. in 2011 from the Department of Computer Science and Technology, Tsinghua University. His research interests are natural language processing and social computation. He has published over 40 papers in international journals and conferences including ACM Transactions, IJCAI, AAAI, ACL and EMNLP.

Maosong Sun is a professor of the Department of Computer Science and Technology, Tsinghua University. He got his BEng degree in 1986 and MEng degree in 1988 from Department of Computer Science and Technology, Tsinghua University, and got his Ph.D. degree in 2004 from Department of Chinese, Translation, and Linguistics, City University of Hong Kong. His research interests include natural language processing, Chinese computing, Web intelligence, and computational social sciences. He has published over 150 papers in academic journals and international conferences in the above fields. He serves as a vice president of the Chinese Information Processing Society, the council member of China Computer Federation, the director of Massive Online Education Research Center of Tsinghua University, and the Editor-in-Chief of the Journal of Chinese Information Processing.