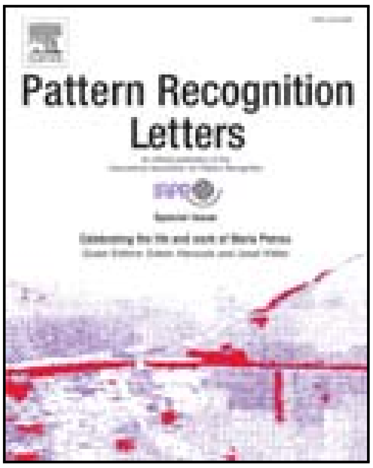


Unsupervised rumor detection based on users' behaviors using neural networks

Weiling Chen, Yan Zhang, Chai Kiat Yeo, Chiew Tong Lau,
Bu Sung Lee

PII:	S0167-8655(17)30382-3
DOI:	10.1016/j.patrec.2017.10.014
Reference:	PATREC 6968

To appear in: *Pattern Recognition Letters*

Received date:	5 January 2017
Revised date:	15 August 2017
Accepted date:	13 October 2017

Please cite this article as: Weiling Chen, Yan Zhang, Chai Kiat Yeo, Chiew Tong Lau, Bu Sung Lee, Unsupervised rumor detection based on users' behaviors using neural networks, *Pattern Recognition Letters* (2017), doi: 10.1016/j.patrec.2017.10.014

Highlights

- We propose comment based features to exploit crowd wisdom to help detect rumors.
- We use Recurrent Neural Networks to study the evolution of these features with time.
- We view rumor detection as an anomaly detection task.
- We adopt Autoencoder to detect rumors and achieve a good F1 measure of 89.16%.
- Our model is unsupervised and thus does not need labeled data.

Pattern Recognition Letters
journal homepage: www.elsevier.com

Unsupervised rumor detection based on users' behaviors using neural networks

Weiling Chen[a,**], Yan Zhang[a], Chai Kiat Yeo[a], Chiew Tong Lau[a], Bu Sung Lee[a]

[a]*Nanyang Technological University, School of Computer Science and Engineering, 50 Nanyang Avenue, Singapore 639798*

ABSTRACT

Online social networks have become the hotbeds of many rumors as information can propagate much faster than ever. In order to detect the few but potentially harmful rumors to prevent the public issues they may cause, we propose an unsupervised learning model combining Recurrent Neural Networks and Autoencoders to distinguish rumors as anomalies from other credible microblogs based on users' behaviors. Some features based on comments posted by other users are newly proposed and are then analyzed over their posting time so as to exploit the crowd wisdom to improve the detection performance. The experimental results show that our model achieves a high accuracy of 92.49% and F1 measure of 89.16%.

1. Introduction

Social psychology field defines a rumor as a controversial and fact-checkable statement (Kimmel, 2004; DiFonzo and Bordia, 2007). Rumors carrying misinformation will cause severe harm especially under some emergent situations. With the facilitation of online social networks (OSN), information propagation has been made more convenient and much more rapid, which thereby greatly amplifying the harm brought about by rumors.

A widely known example is the Fukushima Daiichi nuclear disaster occurred in Japan in March 2011. A rumor saying that iodized salt can protect people from the radiation effects spread fast on China's online social network - Sina Weibo. People rushed to buy much more iodized salt than they need which increased the salt price by almost 5 to 10 times. Similar examples of the abuse of OSN to spread rumors can be found in (Takayasu et al., 2015; DiFonzo et al., 2013). Considering the potential misunderstanding, panic and even hatred caused by rumors carrying misinformation and the infeasibility for humans to manually verify every post on OSN, an automatic mechanism for detecting rumors is of high practical value.

Most of the previous research work treats rumor detection as a classification task. However, we are the first to view it as an anomaly detection task. The theoretical support is that according to Kwon et al. (2013a), the user behaviors of posting rumors will diverge from those of posting genuine facts. In order to exploit such differences to help detect rumors, we build a user behavior model based on the recent microblogs posted by the user within a short period. Since rumors only account for a very tiny proportion of all the posts, most of the microblogs can be regarded as credible posts. Thus we can then treat rumor posts as anomalies.

Apart from using only cues from microblogs themselves, in this paper, we further include crowd wisdom (i.e. other users' comments on the suspicious microblog) to help increase the detection performance of rumors. Previous research discovered that rumor tweets were questioned much more than credible tweets (Mendoza et al., 2010; Ozturk et al., 2015). On top of that, we propose more comment based features to describe such differences in user behaviors when commenting on rumor posts and genuine posts.

In addition, the time at which these comments are posted implies the diffusion pattern of the original microblog. To take into consideration of the variation of features over time, we employ Recurrent Neural Networks (RNN) to analyze the features as inspired by previous research work (Ma et al. (2015, 2016)). Thereafter, these time dependent features are combined with the time independent features extracted from the original microblogs and then input into a variant Autoencoder (AE) for further anomaly detection. Experimental results show that our model combining RNN and the variant AE based on individual users can achieve a high accuracy of 92.49% and F1 measure

[**]Corresponding author: Tel.: +65-85751378;
e-mail: wchen015@e.ntu.edu.sg (Weiling Chen)

of 89.16%.

In summary, our main contributions are as follows:

- We exploit crowd wisdom to perform rumor detection on OSN. To be more specific, we propose new features which are extracted from the comments of the suspicious microblogs to help improve detection performance.
- We further use Recurrent Neural Networks to study the evolution of these features over time to better capture the differences between rumor posts and credible posts.
- Our model is based on individual user's behavior and we view rumors as anomalies in the recent posts by that user. In other words, we view rumor detection as an anomaly detection task.
- We, for the first time, adopt Autoencoder to detect rumors on OSN and experiments show that our model can achieve a good accuracy and F1 measure.
- Our model is unsupervised and thus does not need labeled data which makes it especially useful when rumor data is difficult to obtain for training.

The remainder of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the overview of the proposed learning model for rumor detection. Section 4 illustrates the features we have adopted for the rumor detection and provides the details of how we combine RNN and variant AE to build the detection model. Section 5 presents the experimental results as well as the comparisons with other methods. Section 6 concludes the whole paper.

2. Related Work

Many researchers in computer science field have shown interest in rumor detection on OSN in recent years. However, most of them treat it as a classification problem, hence much research focus is paid to the selection of different types of features. In the following section, we will introduce the related work on feature selection for rumor detection tasks. In addition, we also summarize some highlights of detection methods in the previous research work.

2.1. Feature Selection

The rumor detection task on OSN originated from the analysis of information credibility. One of the earliest work is Castillo et al. (2011). They proposed several features and grouped them into four classes - message, user, topic, and propagation.

Other research work extended this idea by proposing their own properties and features. Yang et al. (2015b), Ito et al. (2015), Kawabe et al. (2015) analyzed the content of the microblogs using hot topic detection, Latent Dirichlet Allocation (LDA) features, sentiment analysis in their detection work respectively. Liu et al. (2015a) further included the streaming features of OSN. Yang et al. (2015a) also analyzed the topology-related properties of the user network to facilitate rumor detection. Liu et al. (2015b), Wang and Terano (2015), Kwon et al. (2013b), Friggeri et al. (2014) identified the differences in the spread patterns of rumors and genuine facts. Zhang et al. (2015) employed implicit features and combined them with shallow features for detecting social rumors.

However, the aforementioned features are mainly extracted from the original microblog itself or the user. There is not much research on including features based on comments. Cai et al. (2014) tried to analyze the crowd responses like comments and reposts of the original microblog but they only consider the word features. Zubiaga et al. (2015) observed the rumorous conversations on OSN and proposed 4 related features but did not give detection results. In our work, we propose features based on comments to fill in the gap so as to improve the detection performance.

2.2. Detection Methods

Most of the previous work mentioned above treat rumor detection task as a classification problem. Therefore, many traditional classifiers like logistic regression, Support Vector Machine (Yang et al., 2012), Random Forest (Ma et al., 2015), Naive Bayes (Sun et al., 2013), decision tree (Liang et al., 2015), etc. are used.

Among them, what is worth mentioning is that Kwon and Ma et al. incorporated a time series fitting model into traditional classifiers in their serial research work (Kwon et al., 2013b; Kwon and Cha, 2014; Ma et al., 2015) and tried to capture the variation of these features over time. A more recent work of them (Ma et al., 2016) proposed a RNN-based model to better understand the variation of aggregated information across different time intervals. However, their RNN-based model only processed and used the words appearing in microblogs as features and did not consider other features. Inspired by this, we employ RNN to further process other additional features extracted from comments on microblogs.

Nevertheless, the aforementioned methods are all supervised methods which need labeled data. Since rumors only make up a small proportion of all the posts on OSN, it is challenging to obtain enough labeled data for training. Some researchers have tried to use unsupervised learning methods to deal with similar tasks on OSN. Miller et al. (2014); Guzman and Poblete (2013) applied anomaly detection idea to detect spam and bursty keywords respectively. For rumor detection field, only our previous work (Chen et al., 2016; Zhang et al., 2016) viewed it as an anomaly detection problem. The underlying assumption is that for an individual user, his or her posting style remains consistent during a period of time. On the other hand, when the user posts a rumor, the posting behaviors deviate from the normal ones and thus can be regarded as anomalies.

Chen et al. (2016) used Principal Component Analysis (PCA) to perform feature selection and the results are then combined with their proposed strategy (Euclidean Distance and Leave One Out) to detect anomalies but even the best accuracy is not very high (82.28%). In addition, the issue with PCA is it is premised on a linear system which may not be applicable all the time.

In our research, we apply AutoEncoder (AE) to achieve the anomaly detection purpose. AE does not have to assume a linear system and the hidden layer in an AE can be of greater

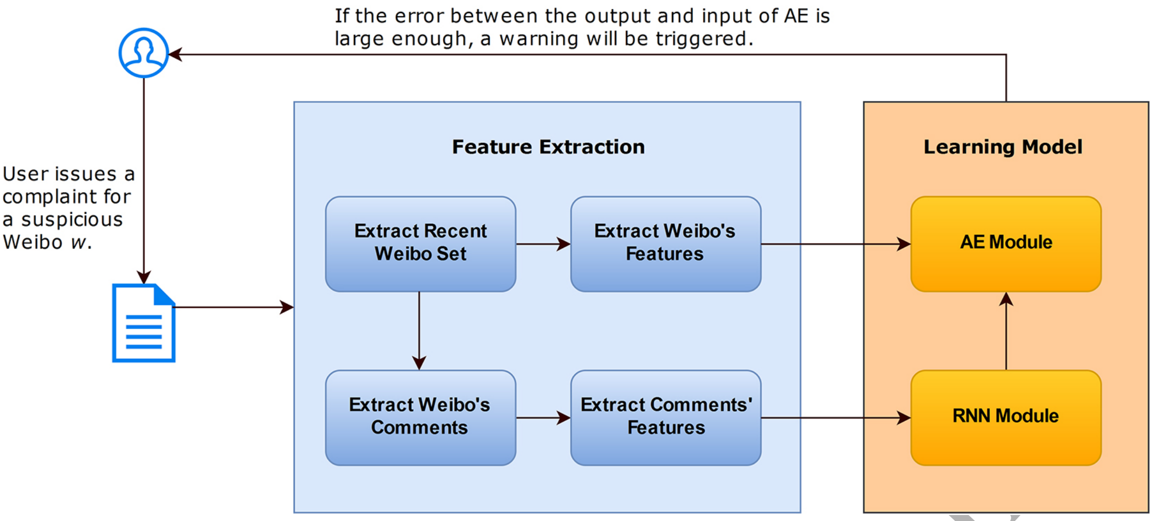

If the error between the output and input of AE is large enough, a warning will be triggered.
User issues a complaint for a suspicious Weibo w.
Feature Extraction
Extract Recent Weibo Set
Extract Weibo's Features
Extract Weibo's Comments
Extract Comments' Features
Learning Model
AE Module
RNN Module

Fig. 1. Overview of the proposed rumor detection model.

dimension than that of the input whereby the data in the new feature space can be disentangled from the hidden factors of variation. Therefore AE can usually learn more features of the original data, in other words, perform better than PCA in feature selection which contributes to a higher detection rate of rumors.

3. Overview of the Model

As we have mentioned in Section 1, in social psychology field, a rumor is a controversial and fact-checkable statement (Zhao et al., 2015). A rumor does not necessarily equate to misinformation. After a while, when external sources are being referred to debunk or verify a rumor, it can turn into an actual rumor (i.e. misinformation) or a genuine fact. Since most of the extremely negative social effects are caused by actual rumors, the focus of this research is to detect these actual rumors. In the later sections, the term ”rumor” shall be used to refer to actual rumors for convenience. On the other hand, for those rumors which turn out to be true, we refer to them as genuine facts or non-rumors.

To illustrate the procedures and performance of our model, we carry out experiments on Sina Weibo which is one of China’s most popular OSN and is often considered as an equivalence to Twitter (Koetse). Since Sina Weibo implements many common features of OSN, our model can be applied to other OSN with little refinement needed.

Fig. 1 shows the overview of our proposed learning model for rumor detection including two phases viz. feature extraction and learning model.

For a suspicious Weibo submitted to our system, we refer to the profile of its author and crawl a set of recent Weibos posted by its author. In the feature extraction phase, each Weibo in the recent Weibo set is represented by the features extracted from itself and its corresponding comments. In the learning phase, comment based features (i.e. time dependent features) of the recent Weibo set will be first put into the RNN module to be analyzed over time as comments arrive at different time. The output of the RNN module is then combined with those Weibo based features (i.e. time independent features) and forwarded to the variant AE as an input. The variant AE is then able to learn the normal behavior pattern of the author and represent them as output. Finally, the errors between the input and output of AE are calculated to perform anomaly detection and a Weibo with a larger error usually indicates a high possibility of an anomaly (i.e. rumor).

4. Methods and Models

In this section, we first introduce the data collection methods and the features we use from both original Weibos and their comments. Then we present the implementation details of RNN and variant AE in our learning model and how we combine them for rumor detection purpose.

4.1. Data Collection

In our experiments, the dataset contains n different original Weibos $\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n\}$ including both rumors and non-rumors. For each original Weibo $\mathbf{w}_i$ (i.e. microblog) submitted to our system, we crawl the recent Weibos posted by its author u_i and thus get its recent Weibo set $\mathbf{S}_{i,k} = \{\mathbf{w}_{i1}, \mathbf{w}_{i2}, ..., \mathbf{w}_{ik}\}$. Assuming the original Weibo $\mathbf{w}_i$ was posted at time t, $\mathbf{S}_{i,k}$ denotes the recent k Weibos posted by user u_i before t. Note that $\mathbf{w}_i$ is the first Weibo in $\mathbf{S}_{i,k}$ (i.e. $\mathbf{w}_i = \mathbf{w}_{i1}$).

There are already some discussions about the selection of k in Chen et al. (2016) and Zhang et al. (2016). k cannot be too large so as to avoid the gradual change in a user’s posting behaviors over time. On the other hand, k cannot be too small either, otherwise, there are too few data samples for the learning model to learn the user behaviors. We set k to 50 based on the experimental results in the aforementioned work.

We choose Sina Weibo to collect data for our dataset because it has an official Weibo Community Management Center[1] to

[1] Weibo Community Management Centre for rumor category: http://service.account.Weibo.com/?type=5

deal with the users' complaints about rumors. Users can easily send a complaint to this center indicating the url of the rumor Weibo. Then professionals employed by Sina Corporation will seek more information to verify or debunk the reported Weibo. Finally, only those regarded by professionals as rumors will be published officially at Weibo Community Management Center for everyone to access. Another reason that we create our own dataset is because the rumor datasets collected in the previous research work are not recent data. Many of them are already deleted by the online social networks and we cannot obtain the comments of these microblogs. The authors also do not provide their datasets for downloading. Therefore we cannot apply our model on their datasets.

We write a crawler to collect these rumors. As all that are published on the platform have been verified by professionals, we can be assured that they are actual rumors. Then we use APIs provided by Sina Weibo[2] to get these Weibos' recent Weibo Sets. Since rumor posts only account for a small proportion of the user's all posts, other Weibos in the recent Weibo set can be used to model the normal behaviors of a user. In this phase, we have crawled 1,257 rumors.

For the collection of non-rumors, we collect Weibos from Sina Weibo's public timeline. Similar to what we do with rumor Weibos, we also crawl their recent Weibo sets. However, we randomly choose one Weibo in the recent Weibo set as the original Weibo for the purpose of avoiding the potential bias of Sina Weibo's public timeline APIs. We then manually go through these original Weibos to make sure they are not rumors. In this phase, we have crawled 2,325 non-rumors.

Both rumors and non-rumors collected at this phase are called original Weibos and are for us to predict their labels. Each post is posted by an individual user. For each user, we crawl their recent Weibo set as well as the comments of these Weibos. Comments of all the recent Weibo sets are crawled in the next step. In total we have crawled 167,731 Weibos including both rumors and non-rumors with their recent Weibo sets and 1,501,472 comments.

4.2. Feature Selection

Each Weibo **w** in the recent Weibo set can be described by the features extracted from itself and its corresponding comments. Thus **w** is represented as $\mathbf{w} = (\mathbf{f}^w, \mathbf{f}^c)$, where $\mathbf{f}^w$ and $\mathbf{f}^c$ are two vectors containing the features extracted from **w** and its comments respectively.

In order to build the behavioral model of user $\mathbf{u}_i$ (i.e. the author of Weibo $\mathbf{w}_i$) from its recent Weibo set $\mathbf{S}_{i,k}$, two types of features are taken into consideration viz. $\mathbf{F}_i^w$ and $\mathbf{F}_i^c$, where $\mathbf{F}_i^w = \{\mathbf{f}_{ij}^w, j = 1, \ldots, k\}$ and $\mathbf{F}_i^c = \{\mathbf{f}_{ij}^c, j = 1, \ldots, k\}$. We follow Chen et al. (2016) and use their feature set (see Table 1) to build our $\mathbf{F}_i^w$ and these features are time independent. The features in $\mathbf{F}_i^c$ are extracted from comments in $\mathbf{S}_{i,k}$. They are time dependent and will be input into RNN for further analysis. Table 2 describes all these features.

The first 13 comment based features are based on the content of comments while the last two features are based on the relationship between the author of the original Weibo and the author of the comment. These features are proposed by us with reference to some previous psychology research by Kimmel (2004) and DiFonzo and Bordia (2007). People tend to express more skepticism to rumors than credible posts. Sometimes they provide figures, links for rumor busting or seek confirmation from friends. We are interested in such differences in responses to rumors and non-rumors, therefore we propose the aforementioned features based on comments.

Table 1. Weibo Based Features

Feature	Description
atti_cnt	The no. of users who liked this Weibo.
cmt_cnt	The no. of users who commented this Weibo.
repo_cnt	The no. of users who reposted this Weibo.
sent_score	Sentiment score of the Weibo.
pic_cnt	The no. of pictures posted in this Weibo.
tag_cnt	The no. of #hastags in this Weibo.
mention_cnt	The no. of @mentions in this Weibo.
smiley_cnt	The no. of smileys in this Weibo.
qm_cnt	The no. of question marks in this Weibo.
fp_cnt	The no. of first person pronouns.
length	The length of the Weibo.
is_rt	Whether the Weibo was a repost.
hour	The time the Weibo was posted.
source	How the Weibo was posted.

Table 2. Proposed Comment Based Features

Feature	Description
has_pic	Whether the comment contains a picture.
face_cnt	The no. of smileys in the comment.
atti_cnt	The no. of likes of the comment.
mention_cnt	The no. of @mentions in the comment.
tag_cnt	The no. of #tags in the comment.
url_cnt	The no. of urls in the comment.
is_reply	Whether the comment is a reply.
is_repo_cmt	Whether the comment is repostted.
positive_pb	The probability the comment is positive.
no_correct_pb	The probability the comment author agrees.
qm_cnt	The no. of question marks in the comment.
fp_cnt	The no. of first pronouns in the comment.
length	The length of the comment.
o_follow_c	Whether u_w follows u_c.
c_follow_o	Whether u_c follows u_w.

Before we input the data into the learning model, we first normalize them with natural logarithm $y(x) = ln(1 + x)$. The reason that we do not choose z-score or min-max method to normalize the data is because they tend to condense the deviation in the data which runs counter to our research purpose.

4.3. Recurrent Neural Networks

A recurrent neural network (RNN) is a feed-forward artificial neural network in which connections between units form a directed cycle. This builds an internal state of the network allowing it to exhibit dynamic temporal behavior. The reason that

[2] Sina Weibo API: http://open.weibo.com/wiki/

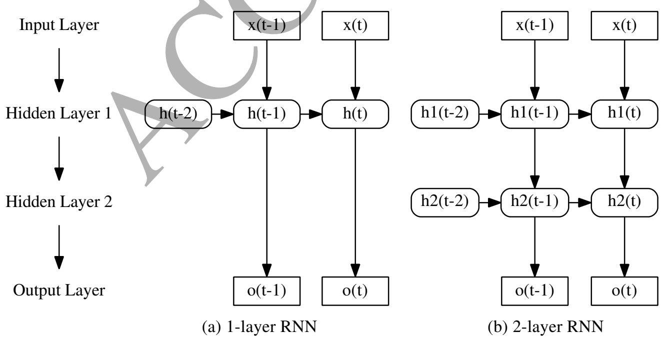
we would like to observe the temporal behaviors is because the propagation pattern of rumors and non-rumors tend to be different. In other words, other users who read the posts respond differently over time depending on whether the original post is a rumor or not.

Before we introduce the details of the RNN module, we would like to introduce the method that we use to group the incoming comments of a Weibo into different time slots. We build time slots for Weibos having at least two comments. We first calculate the time range between the first and last comment of the Weibo, then divide the time period into T time slots equally. Finally all comments are distributed into these time slots according to their posting time. In our paper, we choose $T = 7$ as the number of time slots. According to the information diffusion process presented in Finn et al. (2014) and Matsubara et al. (2012), 7 phases is enough to capture the diffusion characteristics of a piece of information regardless of whether it is a rumor or a credible post. If T is set too small, there is too few information for RNN to learn the diffusion process. On the other hand, if T is set too large, the model tend to overfit and thus not be able to learn the "common behaviors" of users.

Each Weibo $\mathbf{w}$ in the recent Weibo set has different number of comments and the comments of $\mathbf{w}$ can be divided into T sets of comments $\{\mathbf{C}_t, t = 1, \ldots, T\}$, where $\mathbf{C}_t$ denotes the set of comments falling into the tth time slot. For each comment $\mathbf{c}$, we extract m different comment based features in $\mathbf{f}^c$. In our RNN module, the input value of the tth time slot $\mathbf{x}_t = (x_{t,1}, \ldots, x_{t,m})$ is a m-vector. $x_{t,i}$ denotes the ith comment based feature in $\mathbf{f}^c$, calculated at time slot t, by applying function $g(\cdot)$ on the comments in $\mathbf{C}_t$ using Formula 1. We set $x_{t,i} = 0$ for $\mathbf{C}_t$ which contains zero comment (i.e. $|\mathbf{C}_t| = 0$). The function $g(\cdot)$ we used describe the general characteristics of all the comments falling into the same time slot. We test functions like sum and mean and they perform similarly. In our model, we choose the mean function for the following experiments.

$$x_{t,i} = \begin{cases} g\left(\cup_{c \in \mathbf{C}_t} \mathbf{f}_i^c\right), & |\mathbf{C}_t| > 0 \\ 0, & |\mathbf{C}_t| = 0 \end{cases} \tag{1}$$

Furthermore, our proposed one hidden layer RNN module is formalized as follows (see Fig. 2(a)): given an input sequence $\{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, the hidden states $\{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$ are updated at every time step so as to generate the outputs $(\mathbf{o}_1, \ldots, \mathbf{o}_T)$.



(a) 1-layer RNN (b) 2-layer RNN

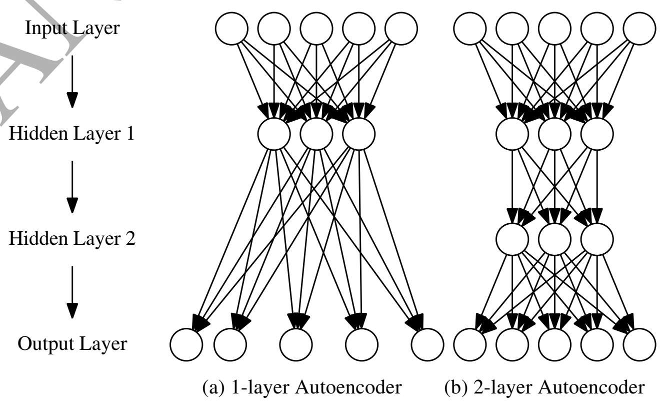
Fig. 2. Proposed RNN module.

$\mathbf{h}_t$ is the hidden state in the tth time slot and can be regarded as the "memory" of the network. $\mathbf{h}_t$ is calculated based on the previous hidden state and the input at the current step. Similarly, $\mathbf{o}_t$ is the output vector. $\mathbf{h}_t$ and $\mathbf{o}_t$ are calculated using Formula 2 in which U_R, W_R, V_R are the input-to-hidden, hidden-to-hidden, hidden-to-output parameters respectively. When training the parameters we calculate the errors between $\mathbf{o}_t$ and $\mathbf{x}_{t+1}$ since we want the output at step t to learn the influence of the comments in the previous time slots and predict the result in the actual next slot as well.

$$\begin{aligned} \mathbf{h}_t &= tanh(U_R \mathbf{x}_t + W_R \mathbf{h}_{t-1}) \\ \mathbf{o}_t &= V_R \mathbf{h}_t \end{aligned} \tag{2}$$

In order to capture higher-level feature interactions between different time slots, we further develop a multiple hidden layer structure of RNN. The output of the multiple hidden layer structure is correspondingly calculated based on the values of the last hidden layer. Fig. 2(b) shows the details of a two hidden layer RNN. In the figure, $h1(t)$ means the hidden unit for the first hidden layer while $h2(t)$ means the hidden unit for the second hidden layer. The structures of the RNN with more hidden layers are implemented in the same manner.

4.4. Autoencoder



(a) 1-layer Autoencoder (b) 2-layer Autoencoder

Fig. 3. Proposed autoencoder module.

An autoencoder neural network is an artificial neural network used for unsupervised learning of efficient codings, i.e. setting the target values to be equal to the inputs. Considering the characteristics of AE, it can be used for anomaly detection (Sakurada and Yairi, 2014). Data experience a dimension reduction process in the hidden layer of AE and in this subspace normal data and anomalies appear significantly different (Chandola et al., 2009).

The details of our proposed one hidden layer AE module can be seen in Fig 3. An Autoencoder takes an input $\mathbf{X}$ and first maps it (with an encoder) to a hidden representation $\mathbf{H}$ through a deterministic mapping. The hidden value $\mathbf{H}$ is then mapped back (with a decoder) into a reconstruction $\mathbf{O}$ of the same shape as $\mathbf{X}$. The mapping happens through a similar transformation. The input data $\mathbf{X}$ is a matrix in our experiments, where the rows of $\mathbf{X}$ denote different Weibos and the columns of $\mathbf{X}$ represent different features. The following formula concludes the calculation process in which W_A, b and V_A, c are the input-to-hidden and hidden-to-output parameters respectively.

$$\begin{aligned}\mathbf{H} =& SoftPlus(W_A\mathbf{X} + b)\\ \mathbf{O} =& SoftPlus(V_A\mathbf{H} + c),\end{aligned} \tag{3}$$

where the activation function $SoftPlus(x) = \ln(1+e^x)$. The reason we choose $SoftPlus$ as the activation function is because the range of the function corresponds to the range of the input data and we can further avoid the vanishing gradient problem.

4.5. Combination Model of RNN and AE

For each original Weibo $\mathbf{w}$, we build a behavioral model combining RNN and AE based on the recent Weibo set and its corresponding comments. The structure of our combination model is shown in Fig 4. The comment based features of the recent Weibo set, namely $\mathbf{X}^i$, will be first put into the RNN module to be analyzed along with the time and the detailed method is described in Section 4.3. We set the output of the RNN module as $\mathbf{X}^o$. $\mathbf{X}^o$ is then combined with those features in $\mathbf{f}^w$ extracted from the recent Weibos, namely $\mathbf{X}^w$, and forwarded to the Autoencoder as an input $\mathbf{X}$, i.e $\mathbf{X} = (\mathbf{X}^o, \mathbf{X}^w)$. The output of AE is $\mathbf{O}$.

To train a standard AE, $\mathbf{X}$ is set as the target value. However, in our variant AE, the target value is set as $\mathbf{Y}$ which combines the input of the RNN module $\mathbf{X}^i$ and $\mathbf{X}^w$, i.e. $\mathbf{Y} = (\mathbf{X}^i, \mathbf{X}^w)$. In a standard AE, the input itself is used as the target value because the input is already the observed real value. However in our case, the input value is not the observed real value and already has some errors because of the previous RNN model. Therefore it is more meaningful to set the target value as the real observed value $\mathbf{Y}$ which does not contain errors. A performance comparison between our variant AE as well as a standard AE is presented in Section 5.

Finally, using a proper detection method based on the errors between the input and output, we can identify the potential rumors.

In our experiments, the input dim m (the number of comment based features) of RNN module is 15 and we set the hidden dim of RNN module as 15 accordingly. In addition, the number of features in $\mathbf{f}^w$ is 14. Given that the number of time slots T is 7, the input dim of the AE module (the number of columns of $\mathbf{X}$) is 119[3] and we simply set the hidden dimension of AE module as 50.

Note that if all the recent Weibos have too few comments (less than 2), we cannot learn the social wisdom from so few comments. To speed up the model, we prune the whole RNN module and set $\mathbf{X} = \mathbf{Y} = \mathbf{X}^w$. In other words, under the situation of too few comments, we will only take into consideration of $\mathbf{f}^w$ and omit $\mathbf{f}^c$.

4.6. Rumor Detection

For an original Weibo $\mathbf{w}_i$ and its recent Weibo set $\mathbf{S}_{i,k}$, the output and target value of AE module can be represented as $\mathbf{O}_i = (\mathbf{O}_{i1}, \ldots, \mathbf{O}_{ik})^\tau, \mathbf{Y}_i = (\mathbf{Y}_{i1}, \ldots, \mathbf{Y}_{ik})^\tau$, where $\mathbf{O}_{ij}, \mathbf{Y}_{ij}$ are the output and target vector of the jth Weibo in $\mathbf{S}_{i,k}$. Using the euclidean norm, the reconstruction error for the jth Weibo can be calculated by:

$$Err_{ij} = \|\mathbf{O}_{ij} - \mathbf{Y}_{ij}\|_2. \tag{4}$$

After calculating the error between the input and output of AE, the error is compared with $threshold_i$ to determine whether it is a rumor or not. As we have mentioned before, user behaviors vary from person to person. Hence we do not set a fixed threshold for all the users. Instead, the threshold is calculated based on the user's own recent Weibo set and is defined as follows:

$$threshold_i = md_i + \max(1, sd_i) \tag{5}$$

where, md_i is the median of all the errors of the recent Weibos $\{Err_{ij}, j = 1, \ldots, k\}$ while sd_i is the standard deviation. With median and standard deviation, we can capture the overall characteristics of the behavior habit and the data. The reason that we do not use $md_i + sd_i$ as the threshold is because if a recent Weibo set does not contain a rumor (i.e. the behaviors of a user remain consistent), the reconstructed errors are stable, thus the sd_i is very small. In this case, $md_i + sd_i \approx md_i$. Therefore half of the weibos in the recent Weibo set shall be regarded as rumors which is not reasonable.

We further draw the box plot of all the standard deviations of the errors based on rumors and non-rumors in Fig 5. It turns out that for non-rumors and their recent Weibo sets, the standard deviations of the errors are smaller and most of them are less than 1. Therefore, by substituting sd_i with $\max(1, sd_i)$ the false positives of the model are reduced.

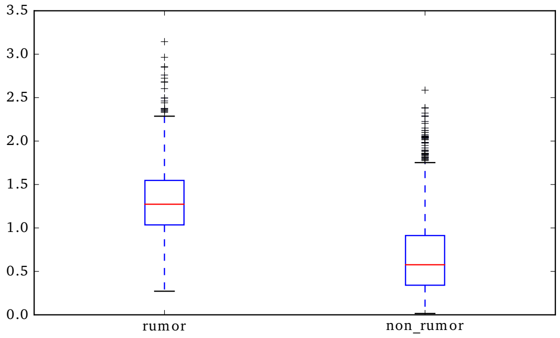


Fig. 5. Standard deviation of rumors' and non-rumors' error on recent Weibo set

Then we predict the label of the original Weibo as follows:

$$isRumor_i = \begin{cases} 1, Err_{i1} > threshold_i \\ 0, Err_{i1} \leqslant threshold_i \end{cases} \tag{6}$$

[3] 119=14+7*15. There are 14 Weibo based features and 15 comment based features while each comment based feature has 7 time slots.

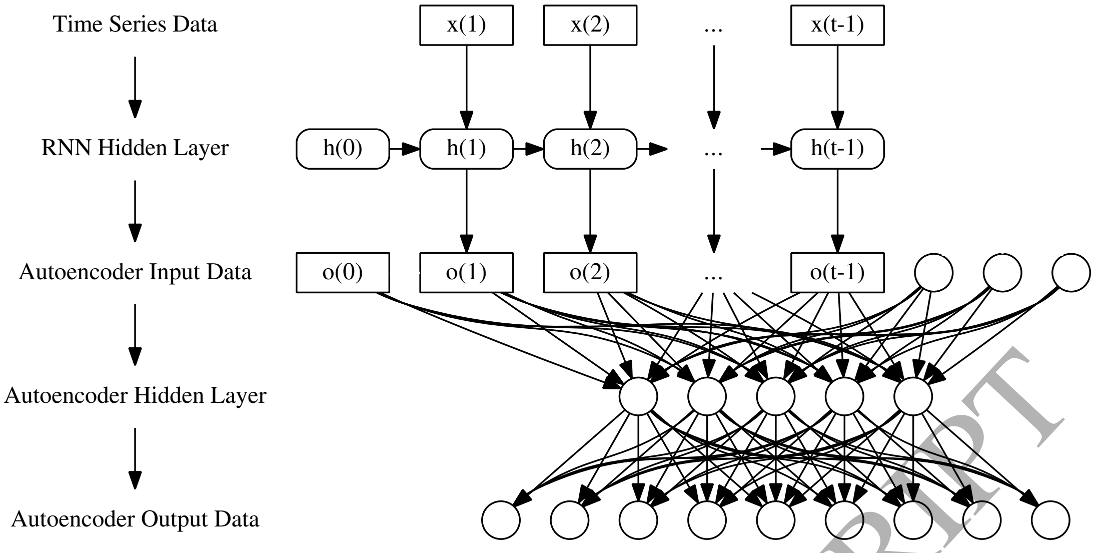


Fig. 4. The combination of RNN and AE.

5. Results and Comparisons

A series of experiments are conducted on all 167,731 Weibos and 1,501,472 comments we have obtained to evaluate the performance of our proposed rumor detection model. All the experiments are run on a Dell PowerEdge R930 Powerful 4-socket 4U Rack Server with 4 Intel Xeon E7-8890 2.5GHz processors and 512 GB of RAM[4]. In this section, we present the results of our proposed learning model as well as some discussions and comparisons. The summary of the results are shown in Table 3.

Table 3. Comparisons of Detection Performance of Different Methods

Method	**Acc** (%)	**F1** (%)	**Pre** (%)	**Rec** (%)	**FPR** (%)
Liang et al. (2015)	88.52	80.73	80.00	81.48	8.53
Chen et al. (2016)	85.68	79.00	81.37	76.77	9.51
Zhang et al. (2016)	85.73	81.53	74.70	89.74	16.43
Zhang et al. (2017)	87.07	81.23	82.81	79.71	8.94
Our-sae	92.10	88.90	87.69	90.13	6.84
Our-agg	88.72	84.58	81.29	88.15	10.97
Our-ind	92.49	89.16	90.36	87.99	5.08

5.1. One Hidden Layer vs. Multiple Hidden Layers

As introduced in Section 4, we implemented both one and multiple hidden layer learning models to compare their detection performance. The detection results with different number of hidden layers are shown in Fig 6.

The only difference among the different models are the number of hidden layers and all other settings are the same. We can see from the results that 2 hidden layer model outperforms one hidden layer model in all 5 metrics and the accuracy reaches as high as 92.49%. However, when the number of hidden layers

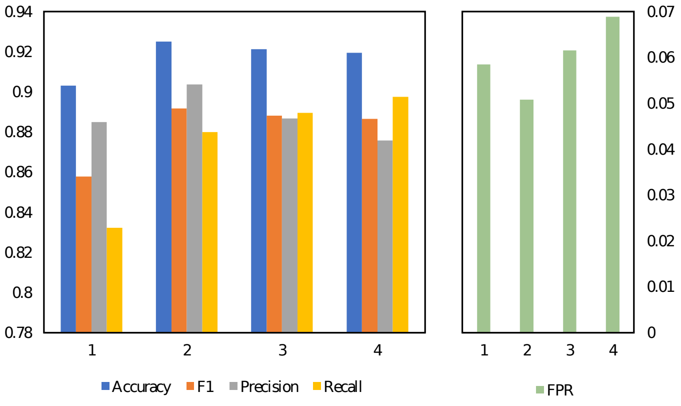


Fig. 6. Performance of learning models with different number of hidden layers.

increases, the accuracy and F1 measure slowly decreases and the false positive rate increases dramatically which can be attributed to the overfitting problem. Therefore, we will set the number of hidden layers to two for the following experiments.

5.2. Standard Autoencoder vs. Proposed Variant Autoencoder

We illustrated in Section 4.5 that the Autoencoder used in the proposed model is slightly different from a standard Autoencoder where the target value is concerned. To better support the validity of the variant AE, we compare the performance of it with a standard AE (i.e. the target value is set to the same as the input which is **X**). In Table 3, Our-sae and Our-ind show the results of a standard AE and the variant AE respectively. It is observed that the variant AE performs better than the standard AE except for the recall.

5.3. One Aggregated Model vs. Individual Models

The most significant assumption of our model is that users behave differently to rumors and non-rumors. Similarly, other

[4] Dell R930: http://www.dell.com/us/business/p/poweredge-r930/pd

users' responses to rumors and non-rumors are different. Many previous research work are based on implementing classifiers for rumor detection. However, they do not distinguish these differences among different users whereby the posting behaviors are quite varied which therefore affect the detection performance.

In an observed extreme case, most postings of a user somewhat bear the features of rumors. If we do not distinguish the user from others, most of his postings may be predicted as rumors. However, when we observe the recent microblogs posted by him, we discover his posting behaviors are quite consistent. In other words, these differences of posting behaviors in rumors and non-rumors are more meaningful when they are observed based on individual users. Chen et al. (2016) and Zhang et al. (2016) mentioned this and applied their model on individual users but did not provide the results on the whole dataset.

In our work, we carry out such a comparison to further support the opinion that individual models are more effective in detecting rumors than a single model. In Table 2, our-ind represents two hidden layer learning model on individual users while our-agg denotes two hidden layer aggregated model on all users. Other parameters are set following the description in Section 4. It is observed that only the recall is slightly smaller than that of the aggregated model, while all the other 4 metrics of the individual model are much better. The false positive rate of the aggregated model is rather high (10.97%) which is consistent with our observations mentioned above. This further proves that it is more effective to build the learning model on individual users than all users for rumor detection.

5.4. Our model vs. Other Methods

Since our method is based on user behaviors, we select 4 other recent work which also exploit similar features for comparisons. The results are shown in the first 4 rows of Table 3. We implement their methods and apply them on our collected dataset described in Section 4.

Liang's method is based on a Support Vector Machine (SVM) classifier which is a supervised learning method. Our unsupervised learning model (denoted as our-ind) performs much better than theirs in all metrics. This is because we further incorporate comments of the original Weibos. In other words, our models learn more information before making a prediction.

The other 3 compared methods are all unsupervised learning methods. For Zhang et al. (2016), our model achieves a good balance between precision and recall as well as a low FPR while their precision is less than 80% along with a rather high FPR. For the other two methods, our proposed method outperforms them in all metrics.

To conclude, our model achieves a satisfying detection performance when compared to other methods.

6. Conclusions and Future Work

Rumors and different kinds of misinformation on OSN have posed one of the most serious problems to both OSN service providers and normal users. Thus it is of great value to detect these rumors automatically to prevent the potential public issues they may cause. However, because rumors only account for a tiny proportion of all the posts on OSN, it is sometimes very difficult to obtain enough data for training. Therefore we propose an unsupervised learning model for rumor detection based on users' behaviors.

In our paper, we propose a combination of RNN and variant AE to learn the normal behaviors of individual users. The errors between outputs and inputs of the model are used to describe the deviation degree of Weibos and are compared with the self-adapting thresholds to determine whether it is a rumor or not. We implement both the one-layer and multiple-layer structure of the learning model and the two-layer model can achieve a high accuracy of 92.49% and F1 of 89.16% which is much better than the compared related methods. We also compare the performance of one aggregated model and individual models for the different users and conclude that the individual models perform better. This further verifies the assumption that the behaviors of different users vary from person to person and that our model is able to exploit this in the detection of rumors.

For future work, more rumors with a wider coverage of topics will be collected so as to further verify the adaptability of the proposed learning model.

Acknowledgments

This work is partially supported by NTU Grant No.M4081329.020, Nanyang Technological University.

References

Cai, G., Wu, H., Lv, R., 2014. Rumors detection in chinese via crowd responses, in: Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on, IEEE. pp. 912–917.

Castillo, C., Mendoza, M., Poblete, B., 2011. Information credibility on twitter, in: Proceedings of the 20th international conference on World wide web, ACM. pp. 675–684.

Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM computing surveys (CSUR) 41, 15.

Chen, W., Yeo, C.K., Lau, C.T., Lee, B.S., 2016. Behavior deviation: An anomaly detection view of rumor preemption, in: Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2016 IEEE 7th Annual, IEEE. pp. 1–7.

DiFonzo, N., Bordia, P., 2007. Rumor psychology: Social and organizational approaches. American Psychological Association.

DiFonzo, N., Bourgeois, M.J., Suls, J., Homan, C., Stupak, N., Brooks, B.P., Ross, D.S., Bordia, P., 2013. Rumor clustering, consensus, and polarization: Dynamic social impact and self-organization of hearsay. Journal of Experimental Social Psychology 49, 378–399.

Finn, S., Metaxas, P.T., Mustafaraj, E., 2014. Investigating rumor propagation with twitter trails. arXiv preprint arXiv:1411.3550 .

Friggeri, A., Adamic, L.A., Eckles, D., Cheng, J., 2014. Rumor cascades., in: ICWSM.

Guzman, J., Poblete, B., 2013. On-line relevant anomaly detection in the twitter stream: an efficient bursty keyword detection model, in: Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description, ACM. pp. 31–39.

Ito, J., Song, J., Toda, H., Koike, Y., Oyama, S., 2015. Assessment of tweet credibility with lda features, in: Proceedings of the 24th International Conference on World Wide Web, ACM. pp. 953–958.

Kawabe, T., Namihira, Y., Suzuki, K., Nara, M., Sakurai, Y., Tsuruta, S., Knauf, R., 2015. Tweet credibility analysis evaluation by improving sentiment dictionary, in: Evolutionary Computation (CEC), 2015 IEEE Congress on, IEEE. pp. 2354–2361.

Kimmel, A.J., 2004. Rumors and rumor control: A Manager's Guide to Understanding and Combatting Rumors. Taylor and Francis.

Koetse, M., . A short introduction to sina weibo: Background and status quo. http://www.whatsonweibo.com/sinaweibo/. Accessed: 2015-04-06.

Kwon, S., Cha, M., 2014. Modeling bursty temporal pattern of rumors., in: ICWSM.

Kwon, S., Cha, M., Jung, K., Chen, W., Wang, Y., 2013a. Aspects of rumor spreading on a microblog network, in: Social Informatics. Springer, pp. 299–308.

Kwon, S., Cha, M., Jung, K., Chen, W., Wang, Y., 2013b. Prominent features of rumor propagation in online social media, in: Data Mining (ICDM), 2013 IEEE 13th International Conference on, IEEE. pp. 1103–1108.

Liang, G., He, W., Xu, C., Chen, L., Zeng, J., 2015. Rumor identification in microblogging systems based on users behavior. IEEE Transactions on Computational Social Systems 2, 99–108.

Liu, X., Nourbakhsh, A., Li, Q., Fang, R., Shah, S., 2015a. Real-time rumor debunking on twitter, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, ACM. pp. 1867–1870.

Liu, Y., Xu, S., Tourassi, G., 2015b. Detecting rumors through modeling information propagation networks in a social media environment, in: International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction, Springer. pp. 121–130.

Ma, J., Gao, W., Mitra, P., Kwon, S., Jansen, B.J., Wong, K.F., Cha, M., 2016. Detecting rumors from microblogs with recurrent neural networks, in: Proceedings of IJCAI.

Ma, J., Gao, W., Wei, Z., Lu, Y., Wong, K.F., 2015. Detect rumors using time series of social context information on microblogging websites, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, ACM. pp. 1751–1754.

Matsubara, Y., Sakurai, Y., Prakash, B.A., Li, L., Faloutsos, C., 2012. Rise and fall patterns of information diffusion: model and implications, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM. pp. 6–14.

Mendoza, M., Poblete, B., Castillo, C., 2010. Twitter under crisis: can we trust what we rt?, in: Proceedings of the first workshop on social media analytics, ACM. pp. 71–79.

Miller, Z., Dickinson, B., Deitrick, W., Hu, W., Wang, A.H., 2014. Twitter spammer detection using data stream clustering. Information Sciences 260, 64–73.

Ozturk, P., Li, H., Sakamoto, Y., 2015. Combating rumor spread on social media: The effectiveness of refutation and warning, in: System Sciences (HICSS), 2015 48th Hawaii International Conference on, IEEE. pp. 2406–2414.

Sakurada, M., Yairi, T., 2014. Anomaly detection using autoencoders with nonlinear dimensionality reduction, in: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, ACM. p. 4.

Sun, S., Liu, H., He, J., Du, X., 2013. Detecting event rumors on sina weibo automatically, in: Asia-Pacific Web Conference, Springer. pp. 120–131.

Takayasu, M., Sato, K., Sano, Y., Yamada, K., Miura, W., Takayasu, H., 2015. Rumor diffusion and convergence during the 3.11 earthquake: a twitter case study. PLoS one 10, e0121443.

Wang, S., Terano, T., 2015. Detecting rumor patterns in streaming social media, in: Big Data (Big Data), 2015 IEEE International Conference on, IEEE. pp. 2709–2715.

Yang, F., Liu, Y., Yu, X., Yang, M., 2012. Automatic detection of rumor on sina weibo, in: Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, ACM. p. 13.

Yang, Y., Niu, K., He, Z., 2015a. Exploiting the topology property of social network for rumor detection, in: Computer Science and Software Engineering (JCSSE), 2015 12th International Joint Conference on, IEEE. pp. 41–46.

Yang, Z., Wang, C., Zhang, F., Zhang, Y., Zhang, H., 2015b. Emerging rumor identification for social media with hot topic detection, in: 2015 12th Web Information System and Application Conference (WISA), IEEE. pp. 53–58.

Zhang, Q., Zhang, S., Dong, J., Xiong, J., Cheng, X., 2015. Automatic detection of rumor on social network, in: National CCF Conference on Natural Language Processing and Chinese Computing, Springer. pp. 113–122.

Zhang, Y., Chen, W., Yeo, C.K., Lau, C.T., Lee, B.S., 2016. A distance-based outlier detection method for rumor detection exploiting user behaviorial differences, in: Data and Software Engineering (ICoDSE), 2016 International Conference on, IEEE. pp. 1–6.

Zhang, Y., Chen, W., Yeo, C.K., Lau, C.T., Lee, B.S., 2017. Detecting rumors on online social networks using multi-layer autoencoder, in: Technology & Engineering Management Conference (TEMSCON), 2017 IEEE, IEEE. pp. 437–441.

Zhao, Z., Resnick, P., Mei, Q., 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts, in: Proceedings of the 24th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee. pp. 1395–1405.

Zubiaga, A., Liakata, M., Procter, R., Bontcheva, K., Tolmie, P., 2015. Crowdsourcing the annotation of rumourous conversations in social media, in: Proceedings of the 24th International Conference on World Wide Web, ACM. pp. 347–353.