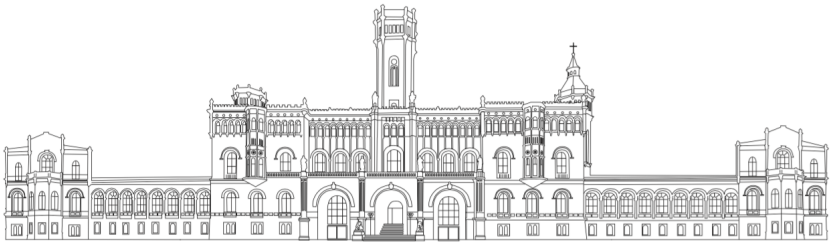


LEIBNIZ UNIVERSITÄT HANNOVER

FAKULTÄT FÜR ELEKTROTECHNIK UND INFORMATIK
INSTITUT FÜR VERTEILTE SYSTEME

Rumor Detection On Twitter

Masterarbeit

eingereicht von

CHENG LI

am 13.11.2016

Erstprüfer :	Prof. Dr. techn. Wolfgang Nejdl
Zweitprüfer :	Prof. Dr. Eirini Ntoutsi
Betreuer :	M.Sc. Tu Nguyen

Abstract

In this thesis, we extensively study the task of rumor detection in Twitter. Twitter is a microblogging service which is already used by millions of users. Users can publish and exchange information with short tweets through all kinds of Twitter clients: website, mobile phone or email. These advantages make Twitter an ideal media for spreading breaking news and false rumors.Therefore, automatic detecting rumors in social media has recently become a trending topic. Early research mostly focused on the rumors during one or several specific events like earthquake or terrorist attack. But in our work, we study on the general types of impact rumors. Additionally most previous works for rumor detection are modeled with static features, like the content of tweets or propagation features, and they ignored the fact which those features can change with the development of the event by time.

We build up a system with Random Forest and the dynamic model proposed by Ma et al. [29] called Dynamic Series-Time Structure (DSTS) with the temporal features and their varieties over time which is named TS-RF. We adapt a series of mathematical epidemiological models (i.e., SpikeM, SIS and SEIZ) to model the propagation patterns of rumors and add them as temporal features for TS-RF. To improve the time series model's performance at early stage of the events, we develop a single tweet's credibility scoring model with CNN+LSTM, which can predict a single tweet as rumor related or news related, with 81.20% accuracy. The output of the single tweet's credibility scoring model which we called CreditScore, is a novel and then is shown to be the best feature for time series model TS-RF in our extensive experiments. Overall, our time series rumor detecting model (TS-RF) reaches over 90% accuracy and out-performs state-of-the-art models (e.g., DSTS) in the study of 48 hours. Last but not least, to the best of our knowledge, we are the first work to evaluate the temporal fluctuations performance of rumor features over time.

Zusammenfassung

In dieser Arbeit, wir umfassend untersuchen die Aufgabe der Gerüchtekorrektur in Twitter. Twitter ist ein Microblogging-Dienst, der bereits von Millionen von Nutzern verwendet wird. Benutzer können veröffentlichen und tauschen Informationen mit kurzen Tweets durch alle Arten von Twitter-Clients: Website, Handy oder E-Mail. Diese Vorteile machen Twitter zu einem idealen Medium für die Verbreitung von Nachrichten und falschen Gerüchten. Daher ist die automatische Erkennung von Gerüchten in Social Media vor kurzem Thema geworden. Die frühe Forschung konzentrierte sich hauptsächlich auf die Gerüchte während eines oder mehrerer spezifischer Ereignisse wie Erdbeben oder Terroranschläge. Aber in unserer Arbeit, wir untersuchen die allgemeinen Arten von Auswirkungen Gerüchte. Darüber hinaus sind die meisten früheren Arbeiten für die Erkennung von Gerüchten mit statischen Funktionen, wie der Inhalt von Tweets oder Propagation Funktionen modelliert, und sie ignoriert die Tatsache, dass diese Funktionen können mit der Entwicklung des Ereignisses von Zeit zu ändern.

Wir bauen ein System mit Random Forest auf und das dynamische Modell, das von Ma et al. [29]. Als "DSTS" bezeichnet wird, mit den zeitlichen Merkmalen und deren Varietäten, die mit dem Namen TS-RF bezeichnet werden. Wir adaptieren eine Reihe von mathematischen epidemiologischen Modellen (d.h. SpikeM, SIS und SEIZ), um die Ausbreitungsmuster von Gerüchten zu modellieren und sie als zeitliche Merkmale für TS-RF hinzuzufügen. Um die Leistung des Zeitreihenmodells im frühen Stadium der Ereignisse zu verbessern, entwickeln wir mit CNN + LSTM ein einziges Tweets Glaubwürdigkeitsbewertungsmodell, das mit einem Genauigkeitsindex von 81,20% einen einzelnen Tweet als Gerücht oder Neuigkeiten prognostizieren kann. Die Ausgabe des Credibility-Scoring-Modells des einzelnen Tweet, das wir CreditScore nannten, ist ein Roman und wird dann als das beste Feature für das Zeitreihenmodell TS-RF in unseren umfangreichen Experimenten

gezeigt. Insgesamt erreicht unser Zeitreihen-Gerücht-Erkennungsmodell (TS-RF) in der Studie von 48 Stunden eine über 90% ige Genauigkeit und übertrifft modernste Modelle (z. B. DSTS). Zu guter Letzt, nach unserem besten Wissen, sind wir die erste Arbeit, um die zeitlichen Schwankungen Leistung von Gerüchen Features im Laufe der Zeit zu bewerten.

Acknowledgment

Firstly, I want to thank my parents. Without their support, I can’t finish my master study in a foreign country.

I am very grateful to my mentor M.Sc. Tu Nguyen who gave me lots of help and inspiration in master thesis.

I also want to thank Dr. Tuan Anh Hoang who gave me lots of useful advises.

Last but not least, I want to thank my girlfriend for accompanying me through all the hard and good time.

Without you all my master study would not have been a success.

Contents

Abstract iii

Acknowledgment vii

1 Introduction 1
- 1.0.1 Contributions . . . 2
- 1.0.2 Thesis Outline . . . 3

2 Background and Related Work 5
- 2.1 Twitter . . . 5
  - 2.1.1 Retweet . . . 5
  - 2.1.2 Mentions . . . 5
  - 2.1.3 Hashtags . . . 5
  - 2.1.4 Favorite . . . 5
  - 2.1.5 Verified User . . . 6
  - 2.1.6 Followers . . . 6
  - 2.1.7 Following . . . 6
  - 2.1.8 Twitter API . . . 6
- 2.2 Credibility of Tweets . . . 6
- 2.3 Definition of Rumor . . . 6
- 2.4 Machine Learning . . . 7
  - 2.4.1 Machine learning Overview . . . 7
  - 2.4.2 Decision Tree . . . 7
  - 2.4.3 Random Forest (RF) . . . 8
- 2.5 Neural Network . . . 9
  - 2.5.1 Convolutional Neural Network . . . 10
  - 2.5.2 Recurrent Neural Network . . . 10

2.5.3 Long Short-Term Memory . . . 12
2.5.4 GRU Cell . . . 13
2.5.5 Dropout . . . 13
2.6 K Fold Cross Validation . . . 15
2.7 Levenberg-Marquardt Method . . . 15
2.8 Tensorflow . . . 15
2.9 Related Work . . . 16

3 Single Tweet's Creditability Scoring Model 19
3.1 Introduction . . . 19
3.2 Motivation and Related Work . . . 19
3.3 Features . . . 20
3.3.1 Text Features . . . 20
3.3.2 User Features . . . 20
3.3.3 Twitter Features . . . 21
3.4 Experimental Evaluation . . . 22
3.4.1 Datasets . . . 22
3.4.2 Classification Models . . . 24
3.4.3 Experiment Setting . . . 25
3.4.4 Experiment Results . . . 27
3.4.5 Discussion . . . 28

4 Time Series Rumor Detection Model 33
4.1 Introduction . . . 33
4.2 Dynamic Series-Time Structure (DSTS) . . . 34
4.2.1 Time Stamps Generation . . . 34
4.2.2 Dynamic Series-Time Structure (DSTS) . . . 35
4.3 Features . . . 35
4.3.1 Text Features . . . 36
4.3.2 Twitter Features . . . 36
4.3.3 User Features . . . 36
4.3.4 Epidemiological Modeling Features . . . 36
4.3.5 SpikeM model Features . . . 42
4.3.6 CrowdWisdom Features . . . 47
4.3.7 CreditScore Features . . . 47
4.3.8 Feature Selection . . . 47
4.4 Experimental Evaluation . . . 50
4.4.1 Datasets . . . 50
4.4.2 Experiment Setting . . . 50
4.4.3 Experiment Result . . . 51
4.4.4 Discussion . . . 61

5 Outlook **67**
5.1 Conclusion 67
5.2 Future Work 67

Bibliography **69**

List of Figures **79**

List of Tables **81**

1 Introduction

Twitter is a microblogging service which is used by millions of users. Users can publish and exchange information with short tweets within 140 characters. It is cheap and can be accessed through several kinds of clients like website, email or mobile phone. A study by Pew Research Center shows that the people in USA under age of 30 consider Internet as the major source of news and in all ages crowds, Internet becomes the second important media [20]. These advantages make Twitter be one of the most important platform for publishing breaking news, but at the same time it also becomes an ideal media for spreading unverified information. On Twitter, everyone can be a journalist and publish news or rumors without verifying which must be done by traditional journalists before publishing any news. So people are more likely to believe traditional media or news blogger rather than Twitter [46].

Rumor could be defined as a statement whose truth value is unverified or deliberately false [38]. And they could be harmful to the government, market and society. One case is some hacked Twitter accounts spread a rumor about Obama injured in white house. As the consequence, the S&P crashed and wiped off 130 Billion dollars of stock value [31]. So a method of detecting rumors on Twitter can be very useful and it could be better if it can detect rumors as soon as possible before it widely spreads.

The structure of our system is shown in Figure 1.1.

- Crawl the data from Twitter interface
- Extract features from the tweets
- Extract time series features with the dynamic series-time structure
- Train single tweet's credibility scoring model with neural network and merge the output to the time series features
- Train the time series model TS-RF with time series feature

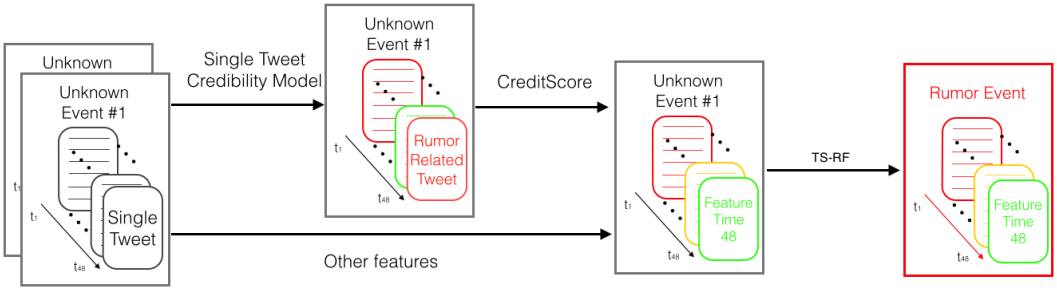


Figure 1.1: pipeline of the rumor detecting system

1.0.1 Contributions

In this thesis, we make the following contributions:

- Our work is the first research which clearly defines the time period of rumor events. Other works either use natural time period (week or month) or do not explain their definition. We develop a novel approach for definition of rumor's time period in Section 3.4.1.
- We develop a model for classifying single tweet with high credibility or low credibility. We call it single tweet's creditability scoring model. It is trained only with features which can be extracted from single tweet. We test 2 different kinds of models: classifier with handcrafted features and network models. Single tweet's creditability scoring model gets 81% accuracy.
- We develop a time series model for detecting rumor events TS-RF. We used Dynamic Series-Time Structure (DSTS)[27] to capture the temporal features. And we use the parameters of 3 epidemiological models as features: modified SpikeM Model [23], SIS model and SEIZ model [16]. Within 48 hours after the event's spreading, TS-RF model can detect the rumor events with 90% accuracy.
- We add the outputs of single tweet's creditability scoring model as a feature into TS-RF, in order to improve the performance in the early stage. We call it CreditScore. And we approve that CreditScore is the best feature in the rumor detection task and it can significantly improve accuracy of rumor detection in first 24 hours.
- We study how the performances of features change over time during the spreading of rumors. For example the performance of features about external URLs

gets better after 24 hours and the features of sentiments are useless after 25 hours. And within 25 hours which is the average time for human editors detecting rumors, we get 87% accuracy for rumor detection.

1.0.2 Thesis Outline

The rest of this thesis is organized as follows: In Chapter 2 we explain some terminology of Twitter and some techniques which we use for extracting feature and modeling. In Chapter 3, we introduce our single tweet's credibility model. We show the performance of models with handcrafted features are worse than the neutral network model. In Chapter 4, we introduce the time series model TS-RF and the time series features. We compare the time series model and static model and we discuss the performance of features changing over time. Finally, in Chapter 5 we add some concluding remarks and describe future work.

2 Background and Related Work

2.1 Twitter

Twitter is a microblogging service. Now there are more than 140 million active users[1]. User can publish short messages within 140 characters as known as tweets.

2.1.1 Retweet

A retweet is re-posting of a tweet by other users. One way of retweet is using "RT" keyword at the beginning of another tweet which means it is a retweet. Another way is using "retweet button" which is officially launched by Twitter after 2015. The difference between these two retweet is tweets are retweeted by "retweet button" can't be searched by Twitter's searching interface, but manually retweets with RT keyword can. So in our work retweet means the tweets are retweeted manually. The number of how many times of this tweet has been retweeted is showed behind it.

2.1.2 Mentions

Mentions are in form like "@username" which are added in the text of tweet. The users are mentioned will receive the notification of this tweet on their homepage.

2.1.3 Hashtags

Mentions are in form like "#topic". It means this tweet belongs to some topics.

2.1.4 Favorite

Favorites means how many users like this tweet. It is showed on the interface of Twitter

[1] https://blog.Twitter.com/2012/Twitter-turns-six

2.1.5 Verified User

Verified User means this account of public interest is authentic by Twitter. It is showed by a blue icon behind the name of the poster.

2.1.6 Followers

The followers of a user are accounts who receive this user's posting. The total number of followers can been seen in the profile of poster.

2.1.7 Following

The following are other accounts who follow this user. The total number of following can been seen in the profile of poster.

2.1.8 Twitter API

Twitter API is provided by Twitter[2] for developer. But the search API only return a sampling of recent Tweets published in the past 7 days[3]. We need the full stories of the events, so we crawled the data directly from the searching interface[4].

2.2 Credibility of Tweets

Twitter has been used for reporting breaking news when emergency events happen like disaster [21]. But the people are likely to trust the news which post on traditional news website more than the news with same headline but posted on Twitter [15]. And the work of Thomson et al. shows that different crowds of people trust tweets basing on different kinds of sources [46].

2.3 Definition of Rumor

The definition of rumor in our work is unverified information spreading on Twitter over time. It is a set of tweets including the the sources, spreaders, retweets, doubting tweets and the denying tweets. If a rumor didn't widely spread and it could be harmless. Therefore, we would like to focus on the rumors which widely spread and contain one or more bursty pikes during propagation. We call it "rumor event".

[2] https://dev.twitter.com/overview/api
[3] https://dev.twitter.com/rest/public/search
[4] https://twitter.com/search-home

2.4 Machine Learning

2.4.1 Machine learning Overview

Machine learning covers vast numbers of algorithms and has been successful applied many different fields. The challenges of ML are finding the best model which is suitable for this task, optimizing the parameters and selecting the features.

Normally we split the ML methods into Supervised learning, Unsupervised learning and Reinforcement learning [40]. The supervised learning is the most popular method of ML. It needs a set of inputs and a set of desired outputs. And the algorithm will learn to produce the correct output based on the new input. The unsupervised learning needs a set of inputs but no outputs. The algorithm will generate the outputs like clusters or patterns. The unsupervised learning task can be used for example when people can't label the outputs.

2.4.2 Decision Tree

Classification is a supervised data mining technique. Our work can be considered as a classification task. Decision tree is a model we use for base line in the after work. A decision tree uses a tree model of decisions and their possible consequences as shown in Figure 2.1. DT is an non-parametric supervised learning method used for classification and regression. Because the trees structure is very simple to visualize, so the result of DT is easy to understand for human. But DTs are unstable [7]. DT is the basic unit of random forest.

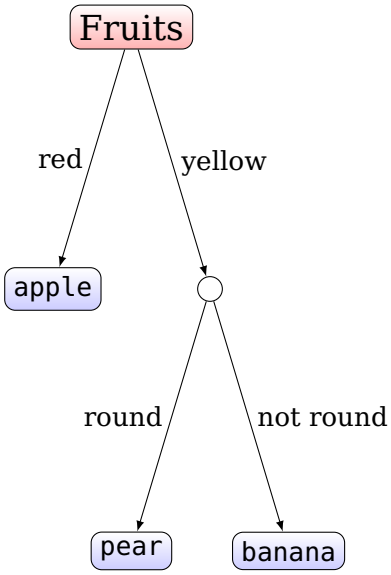


Figure 2.1: Decision Tree

2.4.3 Random Forest (RF)

Another model we use for testing is random forest. RF is an algorithm of supervised learning which developed by Leo Breiman [8]. It's built by a set of classification trees [9]. Each tree is trained by a small bootstrap sample of training set and while prediction each tree votes one single candidate. RF generates the result of prediction By taking the majority vote.

For example we got task to classify pears and apples. We have the features are whether the fruit round, whether the fruit has seeds, whether the fruit red and whether the fruit is juicy. We build up a 3 trees random forest as the graph 2.2. Tree 1 randomly takes the subset of the features red and seeds and votes to apple. Tree 2 randomly takes the subset of the features red and seeds and votes to pear. Tree 3 randomly takes the subset of the features round and red and votes to apple. The most vote is apple, so the output of RF is apple.

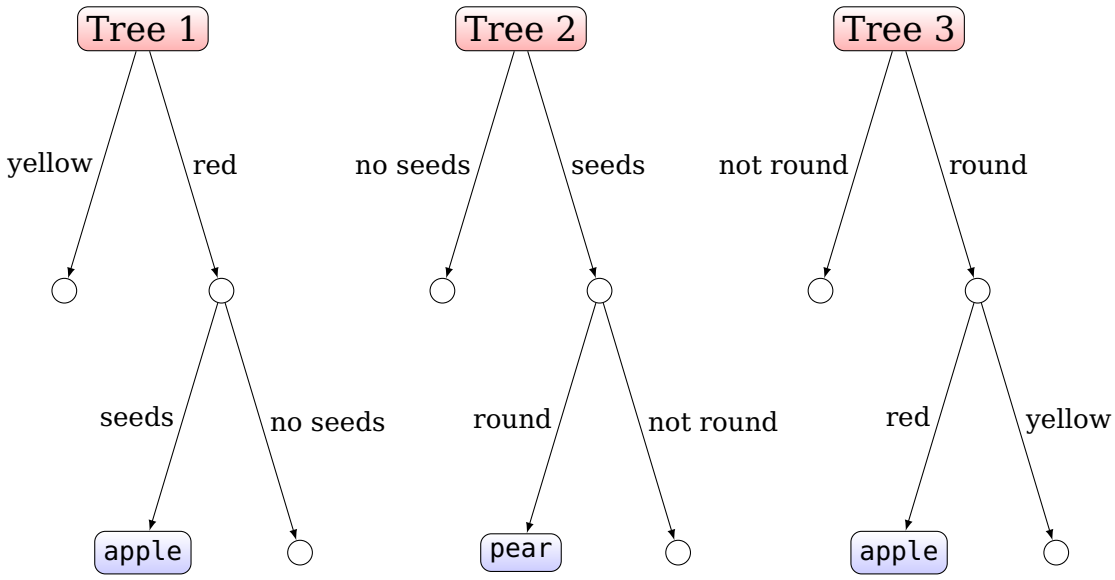


Figure 2.2: An Example of Random Forest with 3 Trees

Because RF uses a random subset of features instead of the best features in every node, so it can avoid the overfitting problem [8]. Another benefit of RF is that it can generate the **features' importance** which is an important index for feature seletion. RF is built up by a subset of training data. If there are N trees, RF will take N times bootstrap sampling. So there must be some features which we didn't select for training the model, we call them out-of-bag (OOB) data. In the above example the feature whether the fruit is juicy didn't join the construction the Trees, so it is a OOB data. These data can be use for validating the model and the output is OBB error $E_{oob}(G) = \frac{1}{N}\sum_{n=1}^{N} err(y_n, G_n^-(X_n))$ with X_n are features only in OOB. At last we get the importance of feature by permuting one feature to random numbers.

$importance(i) = E_{oob}(G) - E^{p}_{oob}(G)$; where $E^{p}_{oob}(G)$ is the OBB error after permuting a feature. The more performance drop down, that means the more important this feature is. We use this method to measure the performance of features and evolute the models later. One more important benefit of RF is it can easily visualize, so it can convince people not only by the test accuracy but also by the giving us a good explanation, for example rumors contain more negative sentiment or the poster of real news more likely live in large city.

2.5 Neural Network

Figure 2.3 is a simple model of forwarding neural network which is a computing systems that processes information by their dynamic state response to external inputs [39]. In the Figure 2.3 the round nodes are called neurons. Each neuron has an active function like sigmod or tanh 2.4. Neurons are connected each others with weight. The process of training the model is actually the process learning the weights of the connections of the network. With the back-propagation algorithm we 1) compute the loss from the network output and 2) update the weights by passing the error backwards through the network [39].

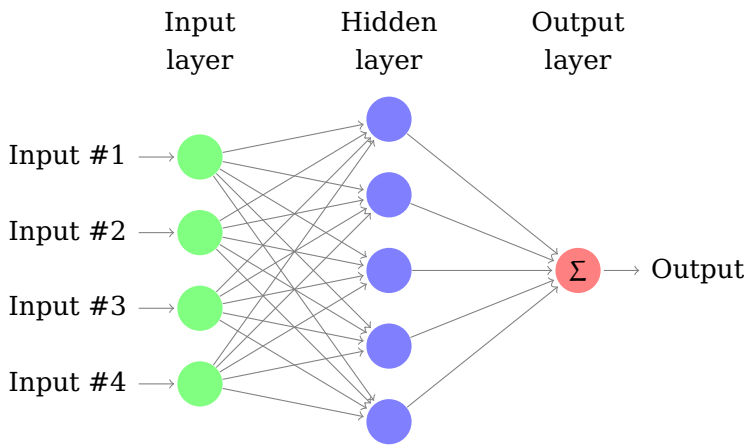


Figure 2.3: An Example of 1 Layer Neural Network

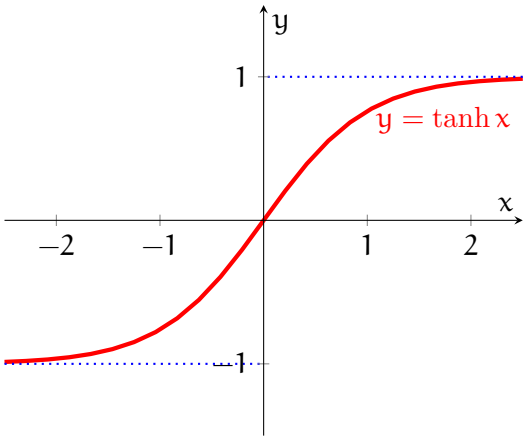


Figure 2.4: tanh Function

2.5.1 Convolutional Neural Network

CNN is a kind of neural network which contains a convolutional layers. First CNN is developed by LeCun et al. [25] for computer vision to resolve the problem of imagines' shift or rotation. With the convolutional layer CNN make that is possible to learn from the local features. Figure 2.5 is an example p of CNN in application of text classification. The word "wait" and its following word "for" are mapped together into different classes through different convolutional filters. That makes convolutional neural network does not learn from the single word form sentence but from the word and its context. That makes sense because one word can in different context have different meaning. For example *"Apple Tree"* and *"Apple Company"*, the same apple but contains different meaning. Pooling layer often follows after the convolutional layers in CNN models whose mainly task is to down-sample from the output of convolutional layers. We use it for for single tweet's creditability score model in the later 3.4.2

2.5.2 Recurrent Neural Network

A Recurrent neural network (RNN) is one of of class of neural network. The different between RNN and other neural network is there are feedback connections between the hidden layers showing as Figure 2.6, that makes the input from the past also can influence the network. The activation status of the hidden layer neurons presents as an internal state which can be considered as a kind of memory. That makes recurrent neural network has the ability to process the sequence input. Human understand the words of a sentence often need the help of the previous words, but the feed forwarding neural network will ignore the previous inputs.

There are several types of applications of RNN. First one is multi-input and single-

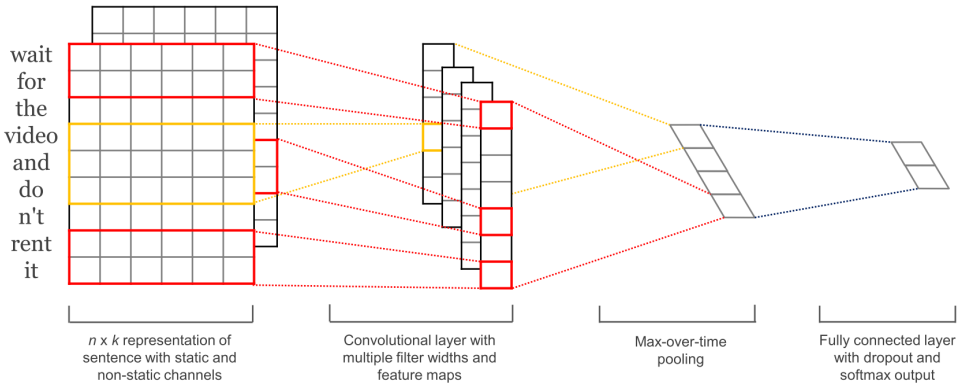


Figure 2.5: CNN for Text Classification (source from[18])

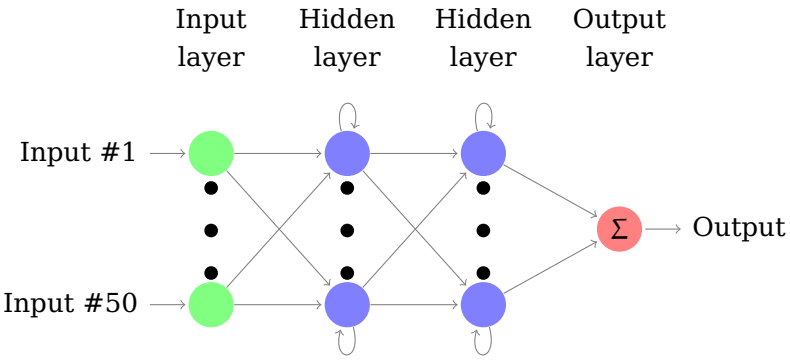


Figure 2.6: 2 Layer Recurrent Neural Network

output 2.7, application in for example text classification. Second is single-input and multi-output 2.8, application in for example input an imagine and generate a sentence for description. Third is multi-input and multi-output 2.9, application in for example the translation system.

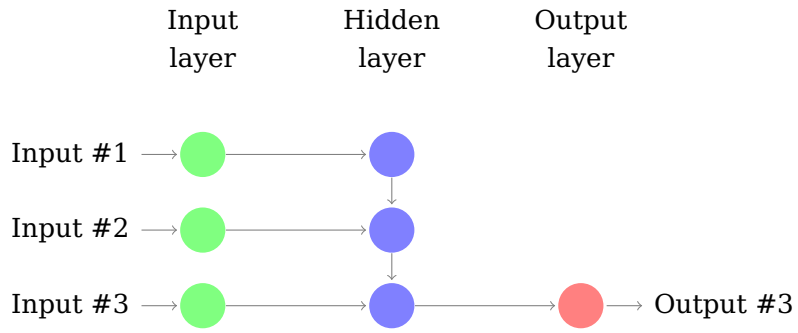


Figure 2.7: Multi-input Single-output Recurrent Neural Network

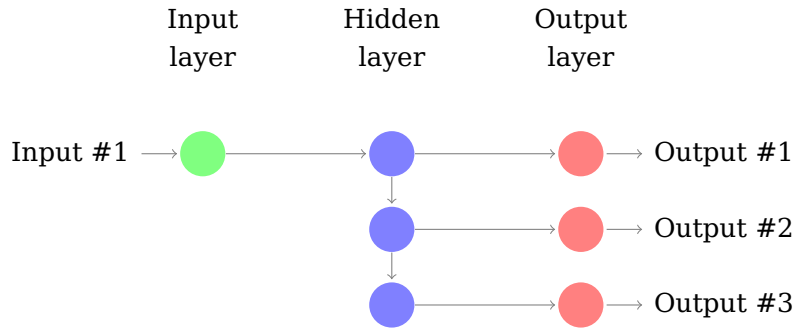


Figure 2.8: Single-input Multi-output Recurrent Neural Network

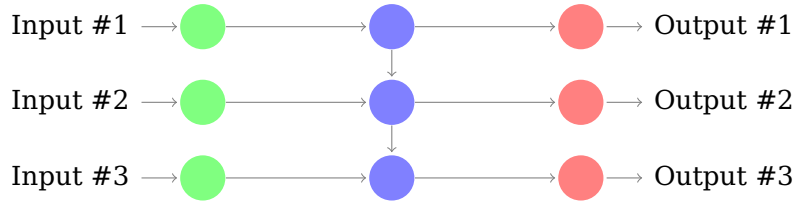


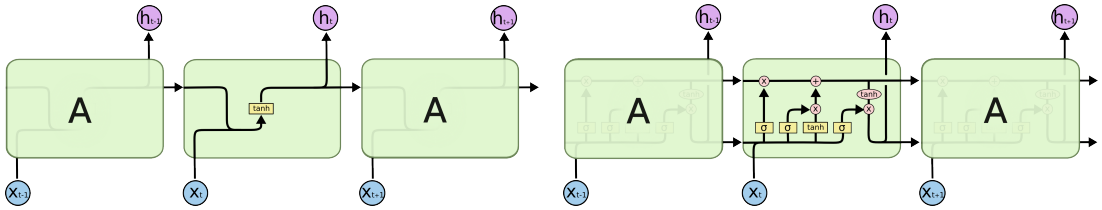
Figure 2.9: Multi-input Multi-output Recurrent Neural Network

2.5.3 Long Short-Term Memory

But recurrent neural networks are difficult to train for a long sequence input, because there are two problems **"gradient explode"** and **"gradient vanish"** [37]. Gradient vanish means that the gradient of the earlier inputs goes exponentially fast towards zero. That make recurrent neural networks can't learn the long-term dependencies in sequences. Gradient explode means the gradients become exponentially large while training.

People can change the active function like ReLU to avoid the vanishing gradient problem, but more popular solution is using Long Short-Term Memory which is first invented by Hochreiter and Schmidhuber [14]. An LSTM memory cell contains self-

connected memory cell with its a memory cell, an input gate, an output gate and a forget gate units. LSTM uses a series gate units to control the cell receiving data flow, extracting data flow or forgetting the current state at each time step.



(a) Simple RNN model (b) LSTM RNN model

Figure 2.10: Two Models of RNNs (a) Normal Model of RNN with tanh Unit (b) RNN with LSTM Cells (source [36])

2.5.4 GRU Cell

GRU Cell is a special case of LSTM which is invented by Cho, et al. [11]. The forget and input gates are merged into a update gate. The cell state and hidden state are combined as one state. These changes make the structure of GRU simpler than LSTM.

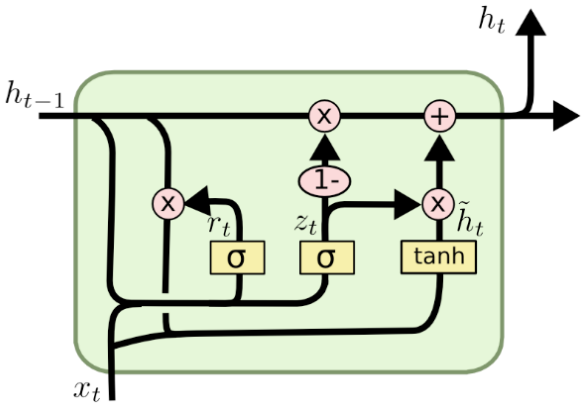
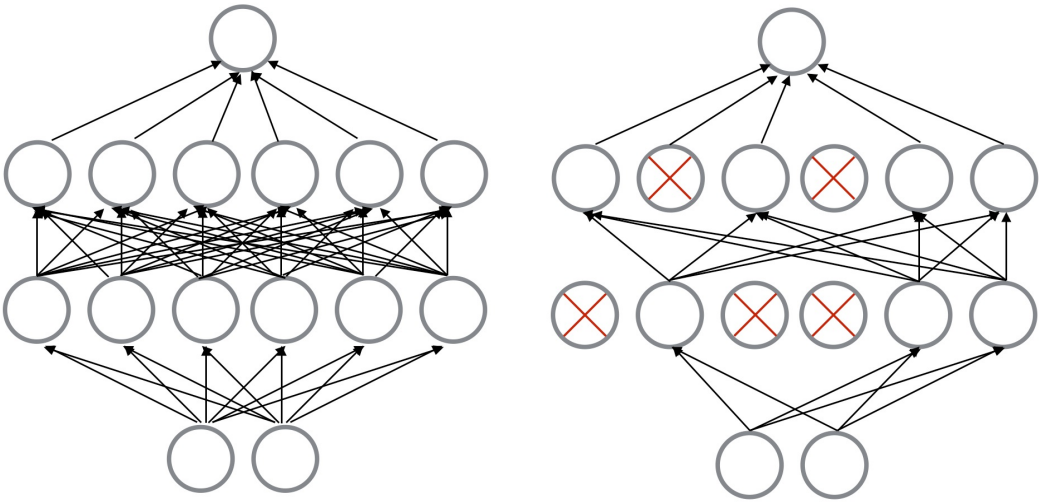


Figure 2.11: GRU Cell (source [36])

2.5.5 Dropout

To avoid the overfitting we deplove dropout which is invented by Nitish Srivastava et al.[42]. The main idea of dropout is to drop some units in hidden layer in neural network while training. When training the model some of the units are removed

temporarily with probability p as Figure 2.12. But while testing the dropout units still join the test. The aim of dropout is to avoid one best state keeping existing, by disabling some units time to time of the network dropout makes the model to be trained in different form every time. Dropout causes the convergence speed slower than without it, the model needs more epochs to find the local best solution. But the benefit is it can avoid the overfitting.



(a) standard NN model (b) NN with dropout

Figure 2.12: Dropout Neural Net Model. (a) a Standard Neural Network (b) after Deploying Dropout

2.6 K Fold Cross Validation

To limit the overfitting and improve the robustness of the model we often use cross validation technology. K fold cross validation is that the dataset was shuffled and split into k subsets. We train the model with some of subsets and others are used for testing.

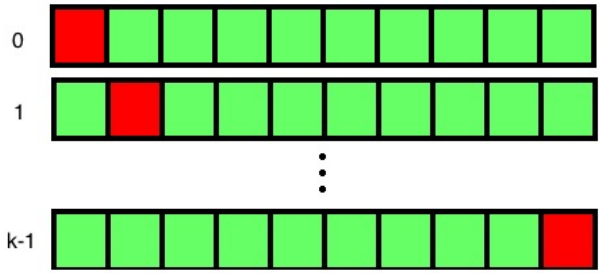


Figure 2.13: K Fold Cross Validation (green rectangle is the training set, red rectangle is the testing set)

In our work we split the data not equally, we consider an event as a unit. We use 10 times cross validation, so we split 260 event into 10 subsets. And we test all models with the same shuffled sequence, which is meaning we test every model with same sequence of training sets and test sets.

2.7 Levenberg-Marquardt Method

The Levenberg-Marquardt algorithm (LMA), as known as, damped least-squares (DLS) method, is used to solve non-linear least squares problems which usually happen in least squares curve fitting.The algorithm was first published in 1944 by Kenneth Levenberg [26]. In our work, we use it to fit SpikeM model, SIS and SEIZ models.

2.8 Tensorflow

TensorFlow[5] is an open source software library for numerical computation using data flow graphs which is used in the single tweet credibility scoring model. It is developed by google. But it still hard for beginner to use and debug. So we use a high-level neural networks library Keras[6] which can run on top of TensorFlow. It is much simpler to configure or to implement.

[5] http://tensorflow.org
[6] https://keras.io/

2.9 Related Work

People research on Twitter for a long time and there are a lots directions to study this complex social network like event detection [19] of Kimmey and David, spam detection [1] [48] or sentiment detection[4]. Those works are similar to our task rumor detection but there are still many differences. In Javier and Yamir's work[6] they use the struct of real world blogs network to identify the influential spreaders of rumor events.

People first studied on rumors in psychology area for years [2] [44]. They summarized the type of rumors and the sentiment with rumors. These research are very helpful for rumor analyzing in Twitter.

But since Twitter becomes an important platform for people sharing and exchanging information including rumors, the detection of misinformation on Twitter turns to be a trending researching topic.

Researcher first began from studying rumor spreading during several special events like natural disasters [33][35][45] or terrorist attacks [43]. In the work of Tanaka et al. [45], they analyzed the how the psychological factors influence the information transmission via tweets. But these results are not general enough to other types of rumors. Our data includes sports, politics, entertainment, also natural disasters and other kinds of rumors.

Carlos Castillo et al. researched the information credibility on Twitter[10][12]. They hired people to label tweets as trustful, mostly untrustworthy and untrustworthy. And they trained a SVMRank model to rank the credibility of Tweets. But his work is based on people's opinion (trustful or not) to a tweet not the credibility of tweet itself. In other words, a false rumor tweet can be trusted by human, but some comments of true news seems not trustful. But it still a good start of researching this problem. Lots of other works are based on Castillo's work [50][27] and used different set of features, for example yang [50] added the location of poster and the client type of user (web client or mobile phone) as two important features.

And there are some simulation studies about the propagation of rumor in twitter. The work of Eunsoo et al. [41] builds a simulation system with random sources of rumor. And they use the propagation graph to detect the sources. Rudra et al.'work [47] use the SIS model to discuss the strategies of preventing rumors' spreading on social networks. They are great works, but the simulation environment cannot be compared with the complex of real network.

There are recent research based on the propagation structure of rumor on Sina Weibo Liu et al. [49]. They use the unique feature of Sina Weibo to study the propagation pattern of rumors and achieved good results. But unfortunately Twitter doesn't give us such detail of propagation process as Weibo, so these works cannot help us. And they focus on one type of rumor "the false photos" on social network not the general type of rumors.

Xiaomo et al. claimed their system is the first real time rumor debunking system on Twitter [27]. But their work are similar with above other previous works, they use the static features of rumors and ignored the feature's changing over time during the event developing on the social network, but one feature called "crowd wisdom" is an new idea and we use it in our work.

Other researchers used propagation models like SpikeM [23] Kwon et al. and SEIZ models [16] Jin et al. to capture the changing of tweets' volume over time, but they didn't use other features into time series model. And these features cannot produce effects in the early stage of rumor spreading. We can provide it in the later Section 4.4.3. Bao et al. used SPNR model which is an extended SIS model as a strategy to control rumors [3].

J Ma et al. used Recurrent Neural Networks for rumor detection [28], they input the tweet of rumor event into RNN time sequentially. Without any other handcrafted features, they got almost 90% accuracy. As the same disadvantage of all kinds of Neural Networks models, the process of training model is a black box to us, so we cannot know the reason why we got this result. In our case, we cannot get the evidences to convince people why the detected event is more likely to be a rumor or rather than a breaking news.

Another work from J Ma et al. used a time series model called Dynamic Series-Time Structure [29] to capture the variation of features. But they only used the features of social content and they didn't further explain how the performances of features change over time or how do they improve the performance of the model in first few hours after the events beginning. We use their Series-Time Structure with our extended data and we will show how do these features change during time.

3 Single Tweet's Creditability Scoring Model

3.1 Introduction

At the beginning of an event, the tweets' volume are limited and there is no clear propagation pattern yet. So same as the method of human verifying rumors we can only focus on the information from each single tweet. Therefore, a single tweet classification model can help us detect the rumors on the early stage.

3.2 Motivation and Related Work

Most of the pervious research focus on event level rumors and claim that the task of classification for individual tweet is not reliable [27] [29] [51], because a single tweet is too short to contain enough features. But Carlos Castillo et al. designed a single tweet's creditability rank system: Tweetcred [12]. So we think it is still enable to build up a single tweet's creditability scoring model.

Recently deep learning is new technology which is used in various areas. And we are inspired by the work of J Ma et al. [24] that we may use neural network without handcrafted features to build up the single tweet's creditability model which in later experiments outputs a better performance. Zhou et al. invented a C-LSTM model [52] for short text classification. The architecture of the model is shown in Figure 3.5. Front hidden layer is a CNN which can split the text into different feature sets and the pooling layer groups the same types of feature together. The last hidden layer is a LSTM layer. They test the model with sentiment classification of comments on IMDb website. According to his paper, C-LSTM achieved the best result for a 2-classification task with 87.8% accuracy. We adapt their work to our rumor detection task.

3.3 Features

In this section, we will introduce the features which we use for single tweet's credibility score model. We use a collection of features majorly from Castillo's Tweetcred system [12] with totally 27 features shown in Table 3.1. These features can be extracted directly from Twitter interface without third part website.

3.3.1 Text Features

The Text features capture the content of tweets. There are 16 Text features like the length of the tweets. **Sentiment Features** are included in text features. We used the python natural language Toolkit (nlTK)[1] to analyze the tweets' sentiment and extract the features: the *NumPositiveWords*, *NumNegativeWords* and *PolarityScores*. *PolarityScores* is a float for sentiment strength of one tweet[2] $Polarity\_scores = \frac{1}{N}\sum_{0}^{n} Polarity(token_n)$.

3.3.2 User Features

We selected total 6 features of the posters. These features are extracted directly from the Twitter interface as in Figure 3.1. ReputationScore is defined as the ratio between #Friends over # Followers. $ReputationScore = \frac{\#Friends}{\#Friends+\#Followers}$.

Figure 3.1: Sample of Users' Information on Twitter Interface

[1] http://www.nltk.org/
[2] http://www.nltk.org/api/nltk.sentiment.html

3.3.3 Twitter Features

Twitter Features are the features which are extracted from the unique functions of Twitter website. It includes *hashtag, mention, number of URLs, number of retweets* and whether this tweet is *retweet* (contains RT as keywords).

Category	Feature	Description
Twitter Features	Hashtag	Whether the tweet contains #hashtag
	Mention	Whether the tweet mentions others @user
	NumUrls	# url in the tweet
	Retweets	How many times this tweet has been retweeted
	IsRetweet	Whether this tweet is retweeted from others
Text Features	LengthOfTweet	The length of tweet
	NumOfChar	# of individual characters
	Capital	Fraction of characters in Uppercase
	Smile	Whether this tweet contains :->, :-), ;->, ;-)
	Sad	Whether this tweet contains :-<, :-(, ;->, ;-(
	NumPositiveWords	# of positive words
	NumNegativeWords	# of negative words
	PolarityScores	polarity scores of the Tweet
	Via	Whether this tweet contains via
	Stock	Whether this tweet contains $
	Question	Whether this tweet contains ?
	Exclamation	Whether this tweet contains !
	QuestionExclamation	Whether this tweet contains multi Question or Exclamation mark
	I	Whether this tweet contains first pronoun like I, my, mine, we, our
	You	Whether this tweet contains second pronoun like U, you, your, yours
	HeShe	Whether this tweet contains third pronoun like he, she, they, his, etc.
User Features	UserFollowers	# of followers
	UserFriends	# of friends
	UserTweets	# of tweets which are posted by this user
	UserDescription	Whether this user has description
	UserVerified	Whether this user is a verified user
	UserReputationScore	Ratio between #Friends over (# Followers + #Friends)

Table 3.1: Features for Single Tweet's Creditability Scoring Model

3.4 Experimental Evaluation

3.4.1 Datasets

In this section, we will introduce the dataset for training and the approach of definition time period of event. We collected rumor stories from a rumor tracking website **snopes.com** and **urbanlegends.about.com**. We crawled 4300 stories from the website and we manually constructed 130 queries of them. The approach of constructing queries is mainly following the work [12]. The regular expression of a query is:

(Object & Subject & Description(Description1||Description2||...))

For example: a rumor story is about Obama removing a flag in pearl harbor. The first term of regular expression *"Object"* is *"Obama"*. The second term *"Subject"* is *"flag"* and its synonym: *flags, flagpole*. The third term *"Description"* is the keyword "remove" and its synonym: *removes, removing* or a URL link about this rumor like *"Departed.co"*. In this example, the proper noun *"pearl harbor"* is also necessary. Finally we transform the regular expression:

(Obama & (flag||flags||flagpole) & (remove||removal|| removing|| "Departed.co") & "pearl harbor")

into Twitter's query:

Obama (flag OR flags OR flagpole) (remove OR removing OR removal OR "Departed.co") pearl harbor

Pervious works like Liu et al. [27] use the news which are reported on **snopes.com** as the training set. But after we test them, we think these news events contain too less tweets. So we use the dataset from the work of Mcminn et al. [32]. They crawl tweets with the Twitter Streaming API from 10th of October 2012 to 7th of November 2012. These events are manually checked that they really happened. From those dataset, we select the top 90 events with most tweets' volume and we add other 40 news events which happened near the time points of rumors. The detail of tweets' volume is shown in Table 3.2:

Type	Min Tweet Volume	Max Tweet Volume	Total Tweet Number	Average Tweet Number
News	18	17414	172616	1327.82
Rumors	44	26010	91268	702.06

Table 3.2: Tweet Volume of News and Rumors

As we mentioned in the Section 2.1.8, the Twitter limits the result of its API by only returning a sample of tweets in recent 7 days. So we have to crawl the data from the web interface. We use Beautiful Soup as the html parsing library to parse the Twitter timeline pages and the users' homepage[3]. Beautiful Soup is a Python library

[3] https://www.crummy.com/software/BeautifulSoup/bs4/doc/

for pulling data out of HTML and XML files. For increasing the speed of parsing html and extracting features from raw data, we use Spark[4] technology, because it can simply manage multithread and its mapReduce in memory technology makes the process much faster.

After we crawled and parsed the whole timeline of an event, we detected the 48 hours time period of the burst in the way which we will mention in following Section 3.4.1. We crawl the homepage of users within the event time period, totally 133,396 users.

Time Period of an Event

In this section, we will introduce the novel approach of defining the time period of event on Twitter. The time period of a rumor event is hard to define. To the best of our knowledge, there is no related work which gave us an approach about how to define the time period of one rumor event. One reason is a rumor may be created serval years ago and kept exiting in Twitter, but it did not attract the crowd's attention. However it can be triggered by other events after a uncertain time and suddenly quickly spread as a bursty event.

For example, a rumor[5] claimed that Robert Byrd was member of KKK. This rumor has been circulating in Internet for a while, as shown in Figure 3.2 that almost every day there were several tweets talking about this rumor. But this rumor was triggered by a picture about Robert Byrd kissing Hillary Clinton in 2016 and twitter users suddenly noticed this rumor and it was spread burstly. And what we are really interested is the tweets which are posted in hours around the bursty peak.

We defined the hour with the most tweets' volume as t_{max} and we want to detect the rumor event as soon as possible before its burst, so we define the time of the first tweet before t_{max} within 48 hours as the beginning of this rumor event, marked as t_0. And the end time of events we defined as $t_{end} = t_0 + 48$. We show the tweets' volume in Figure 3.3 of the above rumor events after defined 48 hours time period.

[4] http://spark.apache.org/
[5] http://www.snopes.com/robert-byrd-kkk-photo/

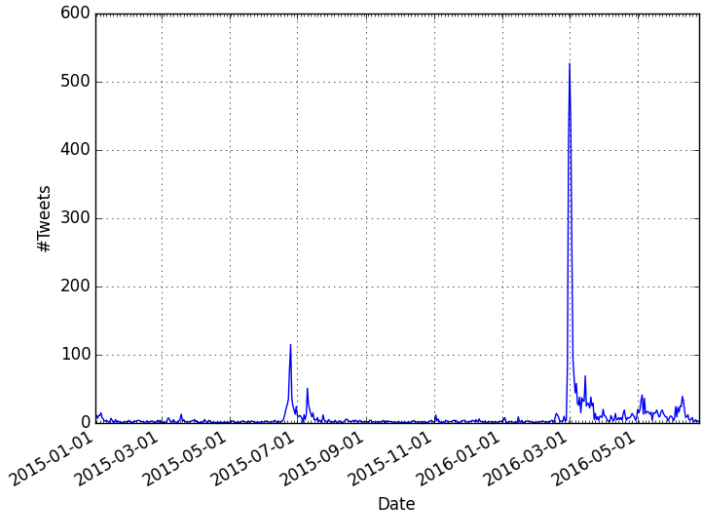


Figure 3.2: Full Scale Tweets' Volume of Event Robert Byrd

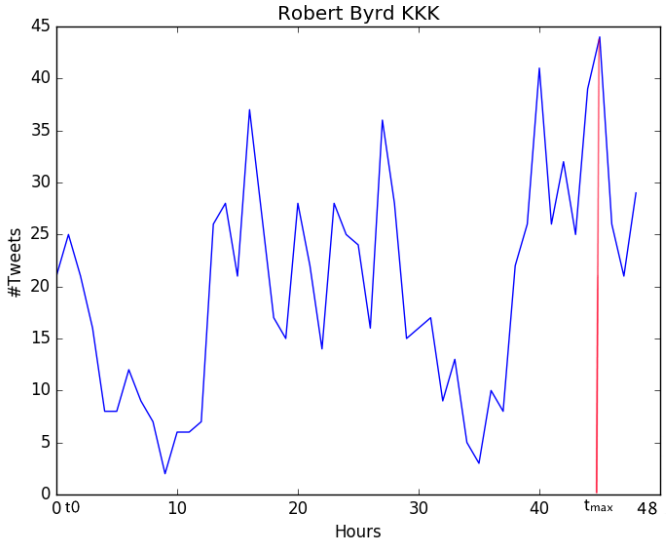


Figure 3.3: tweet volume of the rumor event of Robert Byrd after selected time period

3.4.2 Classification Models

In this section, we will introduce the classifiers of single tweet's credibility model. We developed two kinds of classification models: traditional classifier with handcraft

features and neural network without handcraft features.

Single Tweet's Creditability Model with handcrafted features

We follow Castillo's [12] idea to implement a single tweet's creditability model with above handcrafted features in Section 3.3. We select the most popular classification models for testing: decision trees, decision forest and SVM.

Single Tweet's Creditability Model without handcrafted features

Inspired by the works of Lai et al. [24] and J Ma et al. [28], we test neural networks as the classifier which does not need to extract features from the data. Based on the previous works, we test 6 models: Basic tanh-RNN 3.4(a) as baseline, 1-layer GRU-RNN 3.4(b) ,1-layer LSTM 3.4(c), 2-layer GRU-RNN 3.4(d), FastText 3.4(e) and CNN+LSTM 3.4(f) model as Figure 3.4(d) model. Basic tanh-RNN, 1-layer GRU-RNN, 1-layer LSTM-RNN and 2-layer GRU-RNN models are based on the work of J Ma et al. [28]. FastText comes from work joulin et al. [17] which is a fast text classification model and FastText dose not need GPU for acceleration. The hybrid model of CNN and LSTM (C-LSTM) is an idea from Zhou et al. [52] which has the best performance in our experiments. Zhou et al. test their C-LSTM for sentiment classification by using the dataset of real comments on IMDb website. Those comments are short texts which are similar to tweets and get the best accuracy 86%. And we adapt their model to our work.

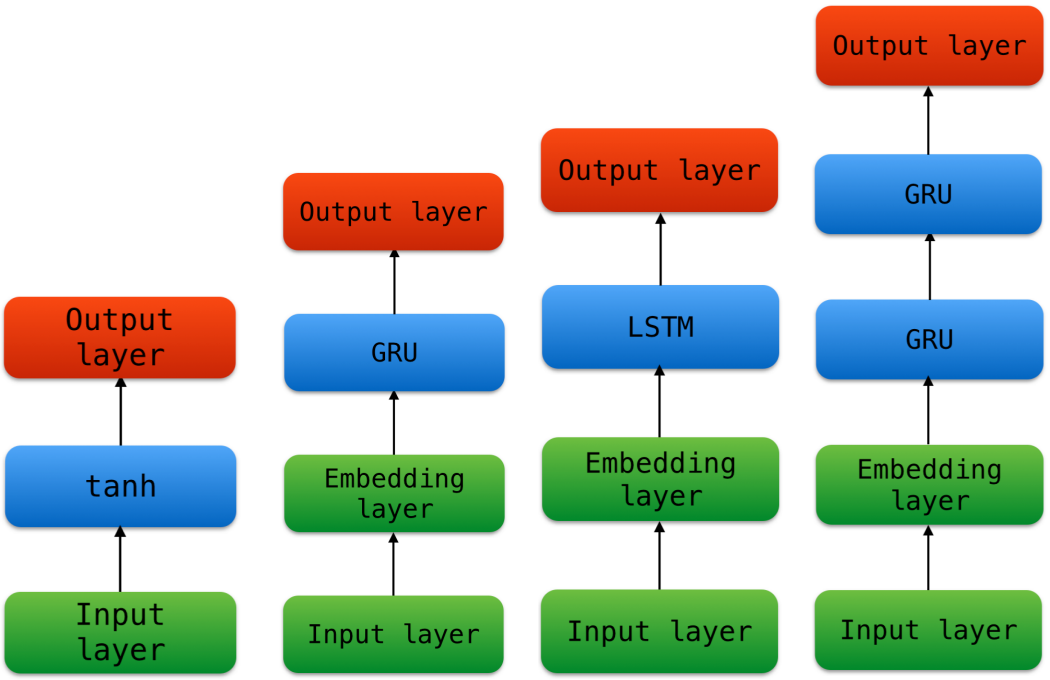
3.4.3 Experiment Setting

In this section, we will introduce the experiment setting of single tweet's credibility model. We shuffle the 260 events and split them into 10 subsets which are used for 10 times cross-validation. We show the parameters after optimization for each model in Table 3.3. And we implement these 3 non-neural network models with scikit-learn library[6]. We implement the neural network with tensor-flow and python library Keras[7].

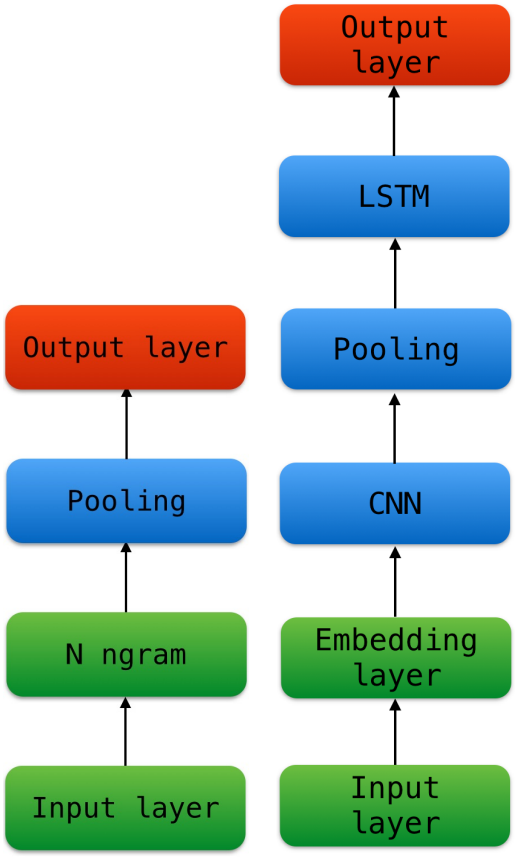
Embedding Layer

The first hidden layer is embedding layer which is set up for all tested models. The embedding size is 50. The output of the embedding layer is vectors representing the words.

[6] scikit-learn.org/
[7] https://keras.io/



(a) Sample RNN Model (b) 1-layer GRU-RNN (c) 1-layer LSTM-RNN (d) 2-layer GRU-RNN



(e) FastText (f) Hybrid CNN + LSTM (C-LSTM)

Figure 3.4: Neural Network Model for Single Tweet Classification

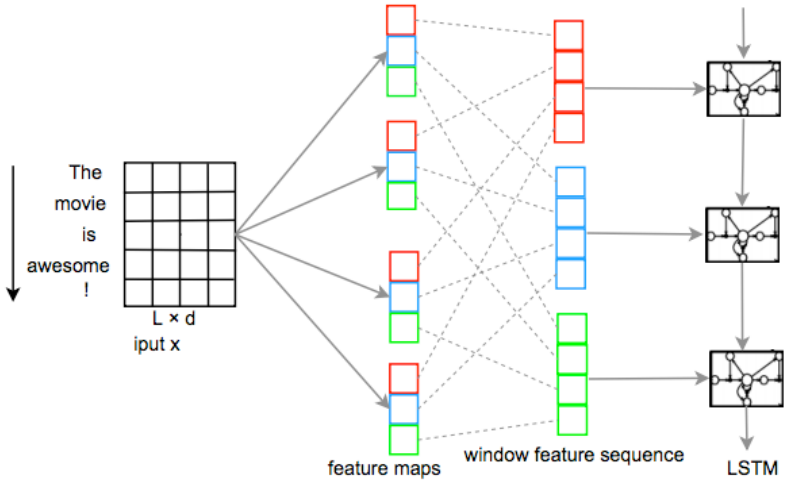


Figure 3.5: The Architecture of C-LSTM Interface (source: [52])

Model	Parameters	Value
Random Forest	Number of Trees	200
SVM	kernel	radial basis function
	penalty parameter of the error term	2.0
	gamma	$\frac{1}{27}$
Decision Trees	criterion	gini

Table 3.3: Parameters of Classification Models

Limit Overfitting

To avoid overfitting, we use 10-fold cross validation and dropout technology.

3.4.4 Experiment Results

In this section, we will introduce the experiment results of above models, which are shown in Table 3.4. The best accuracy result is C-LSTM model. The non-neural network model with the highest accuracy is RF.

C-LSTM has the best performance with 81.19% accuracy, but non-neural network model RF reaches only 64.87% accuracy and the other two models are even lower. So the classifiers with manually handcrafted features are difficulty to accurately distinguish between rumors and news. In Table 3.6 and 3.7 we show some samples of correct labeling and incorrect labeling.

Model	Accuracy
CNN+RNN	**0.8119**
2-layer GRU	0.7891
GRU	0.7644
LSTM	0.7493
Basic RNN with tanh	0.7291
FastText	0.6602
Random Forest	**0.6487**
SVM	0.5802
Decision Trees	0.5774

Table 3.4: Prediction Accuracy of Different Single Tweet's Creditability Scoring Models

3.4.5 Discussion

In this section, we will have some discussions about the experiment results. We rank the features with the features importances and the *PolarityScores* is the feature overall. And we will show some examples of misclassification of neural network .

We rank the features with the features importances which is mentioned in Section 2.4.3, showing in Table 3.5. The best feature is polarity scores of sentiment. It means that there is a big bias between the sentiment of rumors and the sentiment of real events. It was mentioned by previous work of Allport et al. [2] where they gathered a large rumors collection during WW2 which are printed in the Boston Heralds Rumor Clinic. He summarized rumors as several types: pipe-dream, fear and aggression. Most research believe that rumors mostly contain negative sentiment and polarity [22][44]. In our study average polarity score of news event is -0.066 and average polarity score of rumors is -0.1393, it means that rumors contain more negative sentiment.

And we usually think the verified users may have less possibility to be involved in the rumors' spreading, but the result shows that the verified users may be not really trustful like we thought. And "IsReweet" feature is neither a good feature which means the probability of people retweeting the rumors or true events are similar.

In table 3.6 and 3.7, we show some examples of neural network's misclassification and correct classification. The misclassification of news are likely some users' comments to the event like "Who the hell is Mo Yan? Obviously a genious.....or a total bore.". We can compare them with true news tweets like "Congratulations! Mo Yan of this year Nobel Prize!". These misclassified tweets maybe contain doubt,

Feature	Feature Importance
PolarityScores	0.1460686474
Capital	0.09638447209
LengthOfTweet	0.09283739724
UserTweets	0.08750049577
UserFriends	0.08065591431
UserReputationScore	0.08002109553
UserFollowers	0.07938657292
NumOfChar	0.07659755102
Stock	0.04920394972
NumNegativeWords	0.03068379335
Exclamation	0.02304551015
NumUrls	0.02124370609
NumPositiveWords	0.01976939973
Hashtag	0.01851408745
Mention	0.01596532677
Question	0.01486070376
Retweets	0.01349486577
I	0.0109471116
You	0.00998103276
HeShe	0.00774915859
UserDescription	0.007402174886
Via	0.005545157727
QuestionExclamation	0.005422123705
IsRetweet	0.003240079497
UserVerified	0.003081752983
Smile	0.0003979192278
Sad	0

Table 3.5: Features Importance

banter or even rumor related. And on the other hand, the misclassified tweets of real rumors events usually are reports form news website or they may have a news-like style like *"Texas Town Quarantined After Family Of Five Test Positive For The Ebola Virus http://fb.me/3Bbw1uFLS"*.

But sometimes the action of neural network is hard to explain, because we do not know how it exactly works inside. For example *"Dolphins 'deserve human rights' http://zite.to/zEfVKi"* is labeled as rumors, but a similar tweet *"Dolphins 'deserve human rights' http://bbc.in/yFU3og"* is labeled as news.

Catalogue	**Tweet**
News labeled as Rumors	Who the hell is Mo Yan? Obviously a genious.....or a total bore.
	we'll know in a few minutes...EU to win 2012 Nobel Peace Prize: Norwegian broadcaster http://reut.rs/WXXzLU via @reuters
	Wait, are these the Bizarro-world Nobels?
	Little girl in California swims with huge 8 year old pet python http://bit.ly/2a3y7R7 via @BmaxNG
	Head of Fifa partner 'flees arrest': Ray Whelan, head of Fifa partner Match Hospitality, has fled to escape ar... http://bbc.in/1mkvNYZ
	Ahhh really, Dolphins deserve the same rights as humans? What's next, a race option for "porpoise" on legal documents? http://bbc.in/ArhIeb
Rumors labeled as News	#Cancer Cell Phone Use at Night Does Not Cause Eye Cancer http://snopes.comÂă http://goo.gl/i6qNQA
	BBC News: Afghan refugee involved in #Wurzburg attack 'had IS flag in room' http://www.bbc.co.uk/news/world-europe-36832909 #ISIS #Merkel #Germany #EU
	Texas Town Quarantined After Family Of Five Test Positive For The Ebola Virus http://fb.me/3Bbw1uFLS
	Yeahhhhh to stop making music period ! RT ShallowShan_: Bill gates really offered young thug 9 million to stop rapping ?
	Redbox is Hiring Kiosk Ambassadors! http://fb.me/1c5WGQIy2
	Samsung Pays Apple $1 Billion Sending 30 Trucks Full of 5 Cents Coins: http://en.paperblog.com/samsung-pays-apple-1-billion-sending-30-trucks-full-of-5-cents-coins-294795/ ... Comments: http://news.ycombinator.com/item?id=4447550

Table 3.6: Example of False Classification by Single Tweet Model

Catalogue	Tweet
News labeled as News	Congratulations! Mo Yan of this year Nobel Prize!
	The Writer, the State and the Nobel - New York Times (blog): New York Times (blog)The Writer, the State and the ... http://bit.ly/Wa542QÂă
	The incompetent, collapsing EU wins the Nobel Peace Prize? Perhaps next year they could give it to Lance for uniting the world, against him
	Just saw a video of a girl swimming with a Burmese Python on Facebook and i'm just sitting here like WTF?!? O.O
	FIFA Partner, Wanted in World Cup Ticket Scam, Is On the Run: SAO PAULO, Brazil – FIFA partner Ray Whelan gav... http://on.mash.to/1njwnGx
	Dolphins 'deserve human rights' http://bbc.in/yFU3og
Rumors labeled as Rumors	Osama Bin Laden is Still Alive - Edward Snowden http://www.middleeastrising.com/breaking-osama-bin-laden-is-still-alive/ ...
	Samsung pays Apple $1 billion sending 30 trucks full with 5 cents coin! Crazy! #dirtybutgenius
	AMC Announces 'Breaking Bad' To Return For 6th Season; You Won't Believe This Plot Twist http://fb.me/6OYRIMdKx
	For Bill Gates to offer Young Thug $9mill to stop doing music
	Redbox Kiosk Ambassadors #booths http://dragplus.com/post/id/34363441 ...
	Gabourey Sidibe on Joining American Horror Story: Coven: I Hope I Don't Die! http://owl.li/2wzo2t

Table 3.7: Example of Correct Classification by Single Tweet Model

4 Time Series Rumor Detection Model

In this capture, we will introduce the time series model of rumor detection. Firstly, we will intrude the time series data structure (DSTS) and time series model. Secondly, we compare the results with CreditScore and without it. Finally, we will a discussion about the result of experiment.

4.1 Introduction

Most previous works of rumor detection only use the static features, but those features are changing over time during the events' development. For example as we showed in Figures 4.1 and 4.2, the fraction of tweets containing url with top 5000 domains and the fraction of users living in large city which are both constantly changing over time. In order to capture these changes of features, we use Dynamic Series-Time Structure (DSTS) which was presented in the work of Ma et al. [29]. In an Event E_i there is a set of tweets tw_{ij} and we split them into different time intervals according to their creation time, so that we can analyze the features in time series. We test this model with different classifiers and we compare them with static models. In the end, we show our model TS-RF has best performance over all time.

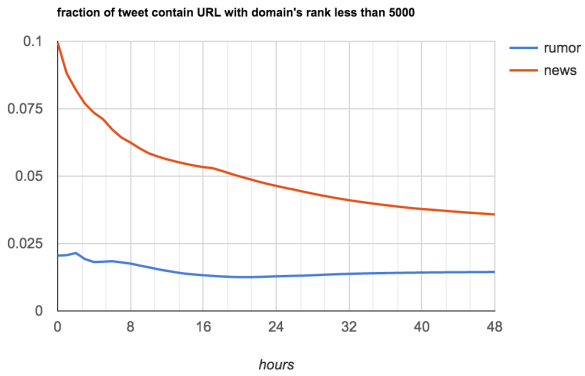


Figure 4.1: The Fraction of Tweets Containing URL with Top 5000 Domain

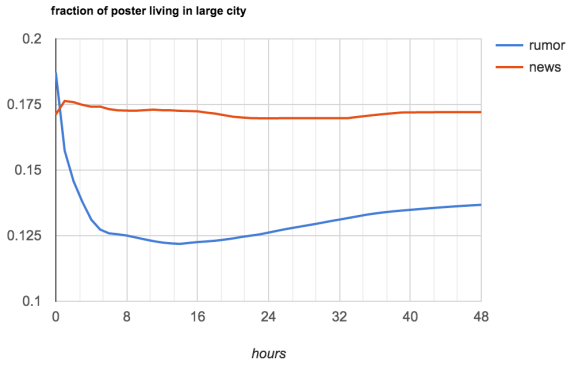


Figure 4.2: The Fraction of Poster Living in Large City

4.2 Dynamic Series-Time Structure (DSTS)

In this section, we will introduce the Dynamic Series-Time Structure which is invented by Ma et al. [29]. We use this structure to capture the temporal features.

4.2.1 Time Stamps Generation

For an event E_i we define $timeFirst_i$ as the start time of the event, $timeLast_i$ as the time of the last tweet of the event. We split each tweet tw_{ij} into N time intervals according to thier creation time. The length of each time interval we define as follows:

$$Interval(E_i) = \frac{\lceil (timeLast_i - timeFirst_i) \rceil}{N} \tag{4.1}$$

And We define the index of time interval $TS(t_{ij})$ where a tweet tw_{ij} which is created in time t_{ij} should fall into, we define as follows:

$$TS(t_{ij}) = \frac{\lfloor (t_{ij} - timeFirst_i) \rfloor}{Interval(E_i)} \tag{4.2}$$

In our work $Interval(E_i)$ as we defined in Section 3.4.1 as one hour and N is constant 48 hours for each event.

4.2.2 Dynamic Series-Time Structure (DSTS)

Since we have all the time intervals of an event E_i and we can generate a vector $V(E_i)$ of features for each time interval. And in order to capture the changes of feature over time, we should not only model the features in individual time intervals, but also model their differences between two time intervals. So the model of DSTS is represented as:

$$V(E_i) = (\mathbf{F}^D_{i,0}, \mathbf{F}^D_{i,1}, ..., \mathbf{F}^D_{i,N}, \mathbf{S}^D_{i,1}, ..., \mathbf{S}^D_{i,N}) \tag{4.3}$$

where the $\mathbf{F}^D_{i,t}$ is the feature vector in time interval t of event E_i. $\mathbf{S}^D_{i,t}$ is the difference between $\mathbf{F}^D_{i,t}$ and $\mathbf{F}^D_{i,t+1}$. $V(E_i)$ is the time series feature vector of the event E_i.

$$\mathbf{F}^D_{i,t} = (\widetilde{f}_{i,t,1}, \widetilde{f}_{i,t,2}, ..., \widetilde{f}_{i,t,D}) \tag{4.4}$$

$$\mathbf{S}^D_{i,t} = \frac{\mathbf{F}^D_{i,t+1} - \mathbf{F}^D_{i,t}}{Interval(E_i)} \tag{4.5}$$

We use Z-score which is implemented by sklearn to normalize feature values .

$$\widetilde{f}_{i,t,k} = \frac{f_{i,t+1,k} - \bar{f}_{i,k}}{\sigma(f_{i,k})} \tag{4.6}$$

where $f_{i,t,k}$ is the k-th feature in time interval t of the event E_i in time interval t. $\bar{f}_{i,k}$ is the mean of the feature k of the event E_i and $\sigma(f_{i,k})$ is the standard deviation of the feature k over all time intervals. We skip this step if we use random forest or decision trees, because they do not need feature normalization.

4.3 Features

In this section, we will introduce the the features of time series model. We use a collection of features based on previous works [10][12] [50][27][28][29] [33][49][16]. They are totally 50 features shown in Table 4.5. These features are extracted not only from Twitter interface but also from other external websites like bluecoat.com which are mentioned in Section 4.4.2.

4.3.1 Text Features

Text features set is a normal feature set of tweets' content. It contains 16 features as shown in Table 4.5. The difference between the text features in single tweet model and in time series model is, in time series model, the features is the ratio of tweets containing some features or average number of some features. For example LengthOfTweet in single tweet model means the length of a tweet, but in time series model it means the average length of tweets in certain time internals.

NumOfChar is the average number of individual characters of tweets. It is case sensitivity. The average number of characters of rumor is 34 and news is 36.

4.3.2 Twitter Features

Twitter features are the features of Twitter's unique functions like hashtag or mention. And we add 3 new features about the URLs of the tweets. The first one is the WOT Score which is crawled from the APIs of website wot.com[1]. WOT is short for Web of Trust and it scores the domains' credibility and safety. The second one is catalog of domain which I crawled from the bluecoat.com[2]. I simply group them into 2 types: news sites or non-news sites. The last one is the rank of the domain which I crawle from alexa.com[3]. I also split them into 2 groups: rank less than 5000 or rank higher than 5000. In our experiment, those 3 features about URLs have better performance than other original Twitter functions like hashtag or mention.

4.3.3 User Features

User's features are also similar with other pervious works. We add one new feature which can only get from the website interface, is how many photos has been posted by the user *(UserNumPhoto)*. And another user feature which has good performance is whether the user lives in a large city. We use the list of large city in the report of demographia[4].

4.3.4 Epidemiological Modeling Features

In this section, we will introduce epidemiological model features. The work of Jin et al., as far as we know, is the first research which uses epidemiological model to analyze rumors' development on Twitter [16]. They describe the propagation pattern of the rumors and news events with two models: SIS (Susceptible, Infected, Susceptible) and SEIZ (susceptible, exposed, infected, skeptic).

[1] https://www.mywot.com/en/api
[2] http://sitereview.bluecoat.com/
[3] http://www.alexa.com/siteinfo/bbc.com
[4] http://www.demographia.com/db-worldua.pdf

SIS is one of the most popular epidemiological models. To adapt to the scenario of Twitter, we define a user who posts a tweet of relevant event as **(I)** infected, a user who dose not we define as **(S)** susceptible. But unlike in the normal epidemiological scenario where infected nodes can be cured and return to susceptible, in Twitter the user once posts a tweet of the certain events, he will be classified into the infected component forever and can not return to susceptible again. At time t, the total number of population is $\Delta N(t) = I(t) + S(t)$ where $I(t)$ is the size of infected population and $S(t)$ is the size of susceptible population. As shown in Figure 4.3, SIS model works as follows:

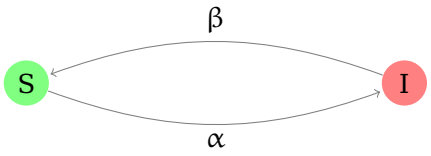


Figure 4.3: SIS Model

- A user who posts tweets about the certain event is regarded as infected.
- A susceptible user has not tweeted about the certain event.
- A susceptible user sees a tweet about the certain event from a infected users and he immediately retweets or posts a tweet about this event, and in that time he turns himself to infected.
- Susceptible user will remain susceptible until he contacts (via tweet) with infected person.

we show SIS model mathematical as follow:

$$\frac{d[S]}{dt} = -\beta SI + \alpha I \tag{4.7}$$

$$\frac{d[I]}{dt} = \beta SI - \alpha I \tag{4.8}$$

SIS model assumes that a susceptible user once exposed to a infected user turns to infected immediately. That is one reason of this model why it dose not fit to Twitter. In fact when twitter users see tweets, they their judgment to truth of the information and they can decide whether further spreading the tweet or ignoring them.

Another popular model is SIR which contains one more term than the SIS. The definitions of **(S)** and **(I)** are the same in SIS, but the term **(R)** stands for recover. Once a susceptible user is recovered, he will be removed from the susceptible component

and he can not be infected again. But we can not get a reasonable explanation of the term R if we model them as an event spreading on Twitter.

Because of the shortcomings of above two models, F. Jin et al. test SEIZ model which is referenced from Bettencourt et al.[5]. To adapt to Twitter context, the

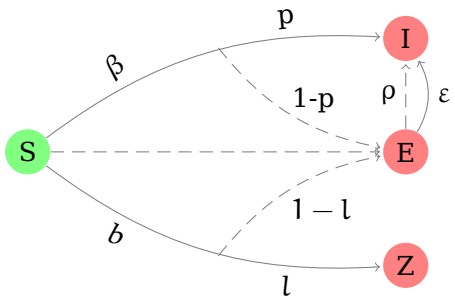


Figure 4.4: SEIZ Model

compartments of the SEIZ model can be mapped like this: **(S)**usceptible is a user who has not been exposed to the event, in the other word, he dose not see any tweets about the certain event yet. **(I)**nfected means a user has posted tweets about the certain events, **(Z)** skeptic is a user who has been exposed to the certain event but he decides to ignore it and **(E)**xposed is a user who has been exposed to the certain event but he will post the tweets after delay.

We show the model in Figure 4.4. And the SEIZ works as follows:

- People recruit from **(S)** susceptible compartment to skeptics with rate b. But with probability l, some of them directly deny the events and turn to **(Z)** skeptic compartments. Others with probability 1-l probability turn to **(E)** exposed compartment.

- People recruit from **(S)** susceptible compartment to infected with rate β. But with probability p, some of them directly believe the events and repost it and turn them to be **(I)** infected compartments. Others with probability 1-p probability turn to **(E)** exposed compartment.

- People from **(E)** exposed compartment may contact again with the Infected nods with ρ probability and turn them to **(I)** infected compartment. And others have ε probability turn into **(I)** infected compartment by themselves, for example driven by external shock.

We show SEIZ model mathematical as follows:

$$\frac{d[S]}{dt} = -\beta S\frac{I}{N} - bS\frac{Z}{N} \tag{4.9}$$

$$\frac{d[E]}{dt} = (1-p)\beta S\frac{I}{N} + (1-l)bS\frac{I}{N} - \rho S\frac{Z}{N} - \varepsilon E \tag{4.10}$$

$$\frac{d[I]}{dt} = p\beta S\frac{I}{N} + \rho S\frac{Z}{N} + \varepsilon E \tag{4.11}$$

$$\frac{d[Z]}{dt} = lbS\frac{Z}{N} \tag{4.12}$$

Symbol	**Definition**
β	S-I contact rate
b	S-Z contact rate
ρ	E-I contact rate
ε	Incubation rate
$1/\varepsilon$	Average Incubation Time
bl	Effective rate of S -> Z
$\beta\rho$	Effective rate of S -> I
b(1-l)	Effective rate of S -> E via contact with Z
$\beta(1-p)$	Effective rate of S -> E via contact with I
l	S->Z Probability given contact with skeptics
1-l	S->E Probability given contact with skeptics
p	S->I Probability given contact with adopters
1-p	S->E Probability given contact with adopters

Table 4.1: Parameters of SEIZ

The author presents an index of SEIZ called R_{SI} as equation 4.13. It contains all rate values of SEIZ and related to the flux ratio of the **(E)** exposed compartment. If R_{SI} is bigger than 1 means the influx of exposed compartment is bigger than the efflux. This index may be a good candidate of feature to analyze rumor spreading on Twitter.

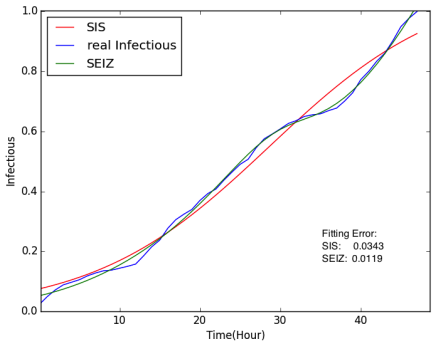
$$R_{SI} = \frac{(1-p)\beta + (1-l)b}{\rho + \varepsilon} \tag{4.13}$$

We use Levenberg-Marquardt algorithm which we present in Section 2.7 to learn the parameters of the SIS and SPEI. The fitting data is the tweets' volume of the 260 events (130 rumors and 130 news). In each time interval from t_0 to t_n, we fit the sequenced tweets' volume from the beginning time t_0 to the current time interval t_n of an event to SIS and SEIZ model and learn the parameters. From SIS we get two features β_n, α_n and from SEIZ we get 7 features $\beta_n, b_n, l_n, p_n, \varepsilon_n, \rho_n, RSI_n$. We add them into our DSTS.

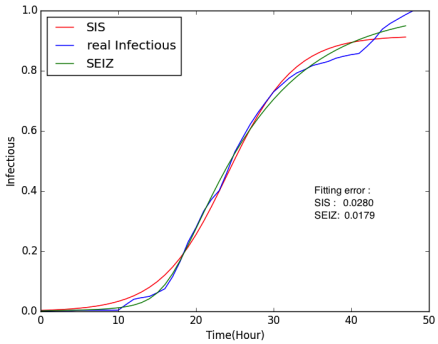
$FittingFunction_{SIS}(TweetVolume_0, ..., TweetVolume_n)$->$\beta_n, \alpha_n$

$FittingFunction_{SEIZ}(TweetVolume_0, ..., TweetVolume_n)$->$\beta_n, b_n, l_n, p_n, \varepsilon_n, \rho_n, RSI_n$

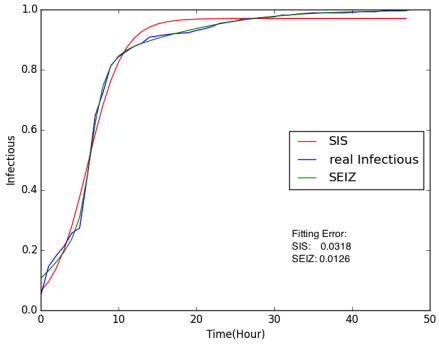
We show 4 examples as following two rumors in Figures 4.6(a) and 4.6(b) and two news in Figures 4.6(c) and 4.6(d). It is obvious that SEIZ is more appropriate than SIS to model in our application of Twitter, because the fitting error of SPEI is less than SIS.



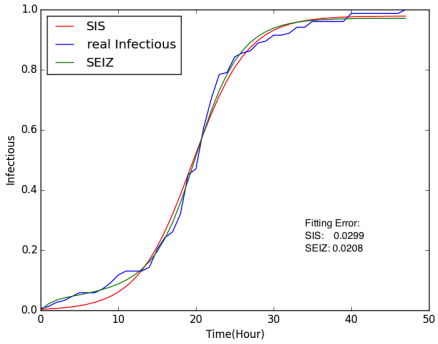
(a) SIS and SEIZ model for rumor 1



(b) SIS and SEIZ model for rumor 2



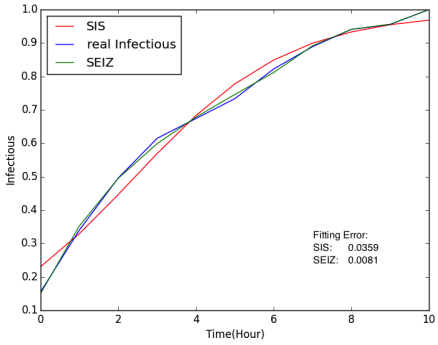
(c) SIS and SEIZ model for news 1



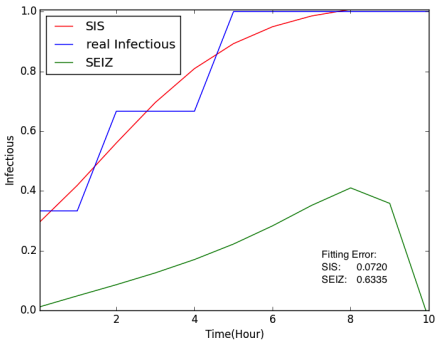
(d) SIS and SEIZ model for news 2

Figure 4.5: Fitting results of SIS and SEIZ model of (a) Rumor: Robert Byrd was a member of KKK (b) Rumor: CNN altered a photograph of a shooter making him look white (c) News: Doctor announces Michael Schumacher is making process (d) News: Two U.S. sailors are arrested over an alleged rape of a Japanese woman on Okinawa

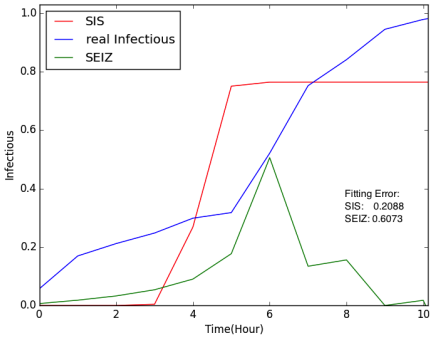
But if we fit the models of the first few hours with limited data, the result of learning parameters is not so accurate. We show the performance of fitting these two model with only the first 10 hours tweets' volume in Figure 4.6. As we can see excepting the first one, the fitting results of other three are not good enough.



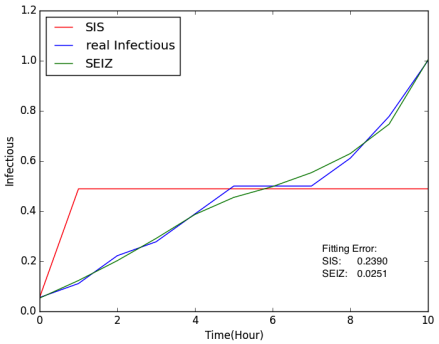
(a) SIS and SEIZ Model for Rumor #1 with 10 Hours Data



(b) SIS and SEIZ Model for Rumor #2 with 10 Hours Data



(c) SIS and SEIZ Model for News #1 with 10 Hours Data



(d) SIS and SEIZ Model for News #2 with 10 Hours Data

Figure 4.6: Fitting Results of SIS and SEIZ Model with Only First 10 Hours Tweets' Volume (same 4 stories as above)

4.3.5 SpikeM model Features

In this section, we will introduce the SpikeM feature set. Kwon et al. showed us another approach of representing the differences between the rumors' propagation pattern and the news events' propagation pattern on Twitter [23]. He adjusted the SpikeM Model and also used the parameters as features.

SpikeM was first introduced by Yasuko Matsubara et al.[30] which can describe the pattern of information diffusion. We present it as follows:

$$\Delta B(n+1) = p(n+1) \cdot (U(n) \cdot \sum_{t=n_b}^{n} (\Delta B(t) + S(t)) \cdot f(n+1-t) + \varepsilon) \tag{4.14}$$

$$p(n) = 1 - \frac{1}{2} P_a(\sin(\frac{2\pi}{P_p}(n + P_s))) + 1) \tag{4.15}$$

$$U(n+1) = U(n) - \Delta B(n+1) \tag{4.16}$$

where

$$f(\tau) = \beta \cdot \tau^{-1.5} \tag{4.17}$$

and initial conditions:

$$\Delta B(0) = 0, U(0) = N \tag{4.18}$$

In addition, adding an external shock $S(n)$, a spike generated at beginning time n_b. Mathematically, it is defined as follows:

$$S(n) = \begin{cases} 0 & (n \neq n_b) \\ S_b & (n = n_b) \end{cases} \tag{4.19}$$

As the definition:

$$B(n) + U(n) = N \tag{4.20}$$

The term of $\sum_{t=n_b}^{n} (\Delta B(t) + S(t))$ is the total number of informed users at time n, so $\Delta B(n+1) = p(n+1) \cdot (U(n) \cdot \sum_{t=n_b}^{n} (\Delta B(t) + S(t)) \cdot f(n+1-t) + \varepsilon)$ means that at time n+1 an infected node n randomly select a node m of all nodes and if the node m is susceptible the probability of m turning to infected is β, so it is a standard SI model. SpikeM extends the SI model from:

- a power-law decay term $f(\tau) = \beta \cdot \tau^{-1.5}$ in equation 4.24. So the infection strength of the nodes which are earlier infected decreases in with a power-law decay pattern.
- a periodic interaction function in equation 4.22. It stands for that people have a periodic interaction patterns, for example people go to sleep at night, so they post much less tweets at night. Parameters P_p, P_a, and P_s are the period, strength, and shift of the periodic interaction function.

Symbol	Definition
N	total population of available bloggers
n_d	duration of sequence
n	time-tick (n=0, . . . , n_d)
U(n)	count of **u**n-informed bloggers
B(n)	count of informed **b**loggers
$\Delta B(n)$	count of informed **b**loggers at time n
f(n)	in**f**ectiveness of a blog-post, at age n
β	strength of infection
$\beta \cdot N$	"first-burst" size of infection
S(n)	volume of external **s**hock at time n
n_b	starting time of **b**reaking news
S_b	strength of external shock at birth (time n_b)
ε	background noise
P_a	strength of periodicity
P_p	period of periodicity
P_s	phase shift of periodicity

Table 4.2: Parameters of SpikeM

- ε is the background noise term.

But the SpikeM can't fit to the events with multi-pike like the Figure 3.3. So the author think the term external shock $S(n)$ in equation 4.25 should not occur once but more. So they extend the SpikeM model by adding a periodic interaction function for the term external shock $S(n)$.

$$\Delta B(n+1) = p(n+1) \cdot (U(n) \cdot \sum_{t=n_b}^{n} (\Delta B(t) + \bar{S}(t)) \cdot f(n+1-t) + \varepsilon) \tag{4.21}$$

$$p(n) = 1 - \frac{1}{2} P_a(\sin(\frac{2\pi}{P_p}(n + P_s))) + 1) \tag{4.22}$$

$$U(n+1) = U(n) - \Delta B(n+1) \tag{4.23}$$

$$f(\tau) = \beta \cdot \tau^{-1.5} \tag{4.24}$$

The external shock $S(n)$ is added a periodic interaction function

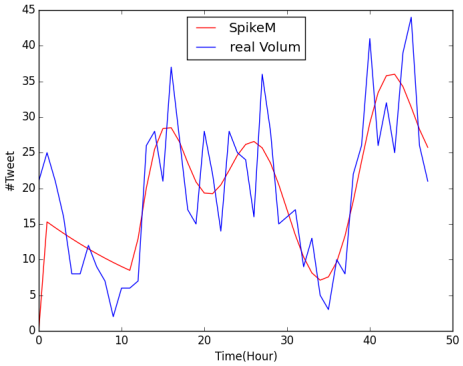
$$\bar{S}(t) = S(t) + q(t) \tag{4.25}$$

$$q(t) = q_a(\sin(\frac{2\pi}{q_p}(t + q_s))) + 1) \tag{4.26}$$

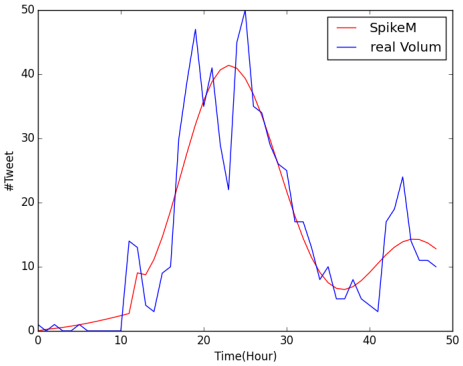
Symbol	Definition
q_a	strength of periodicity of the external shock
q_p	period of periodicity of the external shock
q_s	phase shift of periodicity of the external shock

Table 4.3: New Parameters of Extended SpikeM

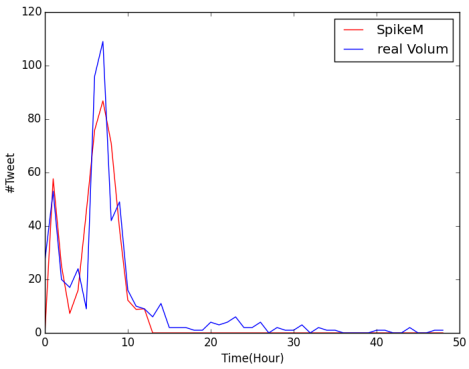
Same approach as fitting SIS model, we learn the parameters of SpikeM model with Levenberg-Marquardt algorithm. We fit the sequenced tweets' volume from the beginning time the t_0 to the current time interval t_n of an event to the model. The output parameters are used as the features adding into DSTS. We use p_a, p_p, p_s and q_a, q_p, q_s as features. We show 4 examples of the SpikeM fitting result in Figure 4.7. But SpikeM has the same problem as fitting SIS or SEIZ model, if we test only within 10 hours data the results are much worse than the results with full 48 hours showing in Figure 4.8.



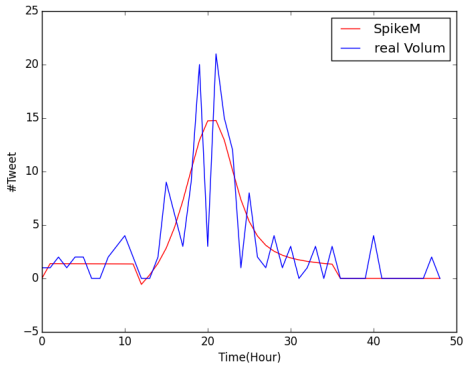
(a) SIS and SEIZ Model for Rumors #1



(b) SIS and SEIZ Model for Rumors #2

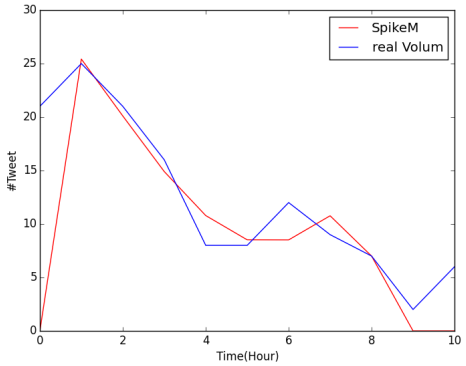


(c) SIS and SEIZ Model for News #1

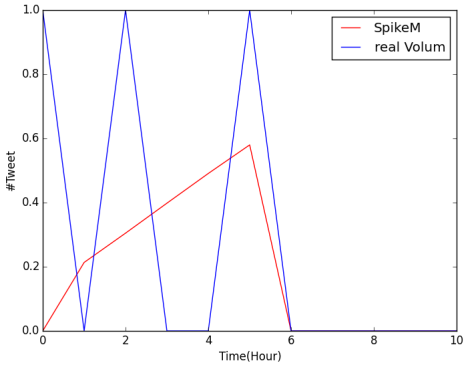


(d) SIS and SEIZ Model for News #2

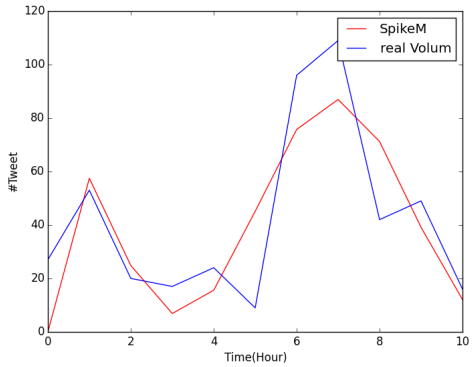
Figure 4.7: Fitting Results of SpikeM Model of (a) Rumor: Robert Byrd was a member of KKK (b) Rumor: CNN altered a photograph of a shooter making him look white (c) News: Doctor announces Michael Schumacher is making process (d) News: Two U.S. sailors are arrested over an alleged rape of a Japanese woman on Okinawa



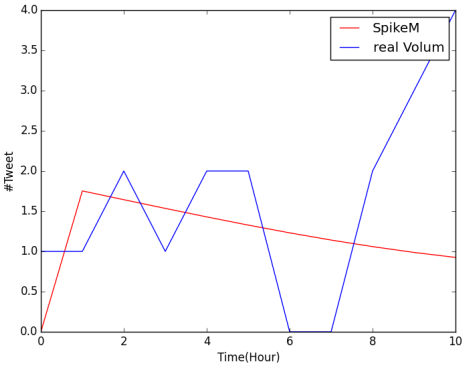
(a) SIS and SEIZ Model for Rumor #1 with 10 Hours Data



(b) SIS and SEIZ Model for Rumor #2 with 10 Hours Data



(c) SIS and SEIZ Model for News #1 with 10 Hours Data



(d) SIS and SEIZ Model for News #2 with 10 Hours Data

Figure 4.8: Fitting results of SpikeM Model with First 10 Hours data (same stories as above)

4.3.6 CrowdWisdom Features

CrowdWisdom feature comes from Liu et al. work [27] but not same. The core idea is using the public's common sense to detect the rumors. If there are more people denying or doubting the truth of an event, this event is more likely to be a rumor. In the Liu et al. work, they use a extensive list of positive, negative and negation keywords and a set of rules like "negative words without negation words means the poster denies the event". And he uses the ratio between number of positive poster (supporter) over the negative poster (deny the events) as a feature.

We simplify from their work by only keeping a set of negative words. We call it "debunking words" like hoax, rumor, not true. Because we think the attitude of denying the event is already enough to distinguish rumors and news. In our test, "debunking words" is a good feature, but it needs 30 hours to "warm up". It is logical because crowds can debunk rumors but they need time to wait the professionals' advices or to unify the attitude the event.

4.3.7 CreditScore Features

To the best of our knowledge, *CreditScore* feature is the first time to be used in rumor detection. Our pre-trained single tweet's creditability model predicts each tweet of the event. If the output is rumor related, we label it 1 otherwise 0. After that, we calculate the average *CreditScore* of the events in certain time interval. We call this feature CreditScore. We will show it in Section 4.4.3 that *CreditScore* is at least the second best feature in our experiment and it improves the performance of time series model especially in the first 24 hours.

4.3.8 Feature Selection

The RF feature selection is a filter method. Firstly we rank all features' average importance. Secondly we increase the number of features with the sequence of the rank, then we compare the performances which are shown in Figure 4.9. The average accuracy of model is improved by adding more features, but after more than 9 features the accuracy stays stable instead of increasing. Therefore, we define the minimal feature set with good enough accuracy as BestSet with 9 features, we show them in Table 4.4.

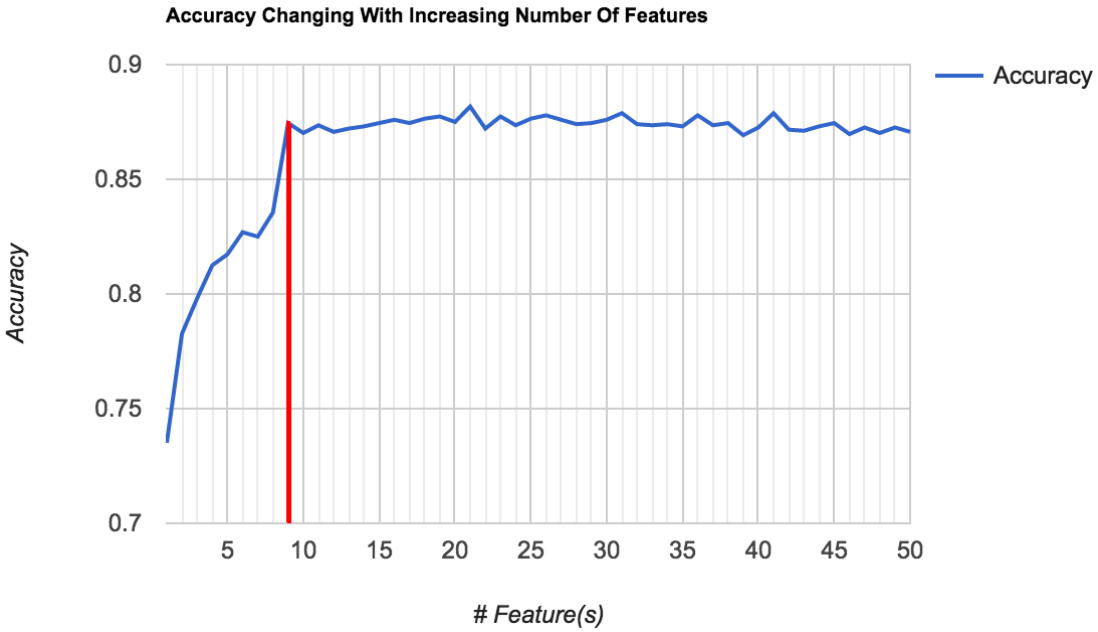


Figure 4.9: Accuracy Changing with Number Of Features

Best Feature Set
CreditScore
ContainNEWS
NumOfChar
UserTweetsPerDays
QuestionExclamation
UserReputationScore
WotScore
Question
UserJoin_date

Table 4.4: Best Features

Category	Feature	Description
Twitter Features	Hashtag	% of the tweets containing #hashtag [10][27][38][12][27]
	Mention	% of the tweets mentioning others @user [10][27][38][12][27]
	NumUrls	# of url in the tweet [10][38][12][50][27]
	Retweets	average times of tweets have been retweeted [27]
	IsRetweet	% of tweets are retweeted from others [10][12]
	ContainNEWS	% of tweets containing URL and its domain's catalogue is News [27]
	WotScore	average WOT score of domain in URL [12]
	URLRank5000	% of tweets contains URL whose domain's rank less than 5000 [10]
	ContainNewsURL	% of tweets contains URL whose domain is News Website
Text Features	LengthofTweet	average length of tweets [10][12]
	NumOfChar	average number of individual characters of tweets [10][12]
	Capital	average fraction of characters in Uppercase of tweets [10]
	Smile	% of tweets containing :->, :-), ;->, ;-) [10][12]
	Sad	% of tweets containing :-<, :-(, ;->, ;-([10][12]
	NumPositiveWords	average number of positive words [10][12][50][27]
	NumNegativeWords	average number of negative words [10][12][50][27]
	PolarityScores	average polarity scores of the Tweets [10][50][27]
	Via	% of tweets containing via [12]
	Stock	% of tweets containing $ [10][12]
	Question	% of tweets containing ? [10][27]
	Exclamation	% of tweets containing ! [10][27]
	QuestionExclamation	% of tweets containing multi Question or Exclamation mark [10][27]
	I	% of tweets containing first pronoun like I, my, mine, we, our [10][12][27]
	You	% of tweets containing second pronoun like U, you, your, yours [10]
	HeShe	% of tweets containing third pronoun like he, she, they, his, etc. [10]
User Features	UserNumFollowers	average number of followers [10][12][27]
	UserNumFriends	average number of friends [10][12][27]
	UserNumTweets	average number of users posted tweets [10][12][50][27]
	UserNumPhotos	average number of users posted photos [50]
	UserIsInLargeCity	% of users living in large city [50][27]
	UserJoinDate	average days since users joining Twitter [10][50][27]
	UserDescription	% of user having description [10][50][27]
	UserVerified	% of user being a verified user[50][27]
	UserReputationScore	average ratio of #Friends over (#Followers + #Friends) [27]
Epidemiological Features	β_{SIS}	Parameter β of Model SIS [16]
	α_{SIS}	Parameter α of Model SIS [16]
	β_{SEIZ}	Parameter β of Model SEIZ [16]
	b_{SEIZ}	Parameter b of Model SEIZ[16]
	l_{SEIZ}	Parameter l of Model SEIZ [16]
	p_{SEIZ}	Parameter p of Model SEIZ [16]
	ε_{SEIZ}	Parameter ε of Model SEIZ [16]
	ρ_{SEIZ}	Parameter ρ of Model SEIZ [16]
	R_{SI}	Parameter R_{SI} of Model SEIZ [16]
SpikeM Model Features	P_s	Parameter P_s of Model Spike [23]
	P_α	Parameter P_α of Model SpikeM [23]
	P_p	Parameter P_p of Model SpikeM [23]
	Q_s	Parameter Q_s of Model SpikeM [23]
	Q_α	Parameter Q_α of Model SpikeM [23]
	Q_p	Parameter Q_p of Model SpikeM [23]
Crowd Wisdom Features	CrowdWisdom	% of tweets containing "Debunking Words" [27] [51]
CreditScore Features	CreditScore	average CreditScore

Table 4.5: Features of Time Series Rumor Detection Model

4.4 Experimental Evaluation

4.4.1 Datasets

We use the same dataset as we mention in Section 3.4.1, totally we collect 260 events and 130 of them are labeled as rumors. Their categories are shown in Table 4.6 and the time span of events are shown in Table 4.7.

Event Categories	News	Rumor
Politics	43	49
Science	12	14
Art	8	17
Business	13	19
Health	6	14
Attacks	27	7
Disaster	7	3
Other	6	7
Total	130	130

Table 4.6: Categories Of Rumors And News

We also extract 11,038 domains which are contained in the tweets in the 48 hours time period and we crawled these domains' categories in bluecoat.com[5], their ranks in alexa.com[6] and WOT score in wot.com[7]. The approach of defining the time period of events is the same as in Section 3.4.1.

Type	Earliest Event	Latest Event	Average Time Span (days)
News	21.02.2012	16.07.2016	4.5
Rumors	20.10.2009	04.08.2016	1679.6

Table 4.7: Time Span of News and Rumors

4.4.2 Experiment Setting

Classification Models

Same reason as in the single tweet's Creditability, we test the time series model also with 3 popular models: Decision Trees, SVM, Random Forest and one more model:

[5] http://sitereview.bluecoat.com/sitereview.jsp#/?search=bbc.com
[6] http://www.alexa.com/siteinfo/bbc.com
[7] https://www.mywot.com/en/api

the multilayer perceptron (MLP). We show the optimized parameters in the Table 4.8.

Model	Parameters	Value
$TS-RF$	Number of Trees	350
$TS-SVM$ [29]	kernel penalty parameter of the error term gamma	radial basis function 3.0 $\frac{1}{50}$
Decision Trees	criterion	gini
MLP	alpha activation function hidden layer sizes weight optimization	0.0001 ReLU 2 layer(50 nodes each layer) adam

Table 4.8: Parameters of Classification models

4.4.3 Experiment Result

In this subsection, we will present the experiment results. We test all models by using 10-fold cross validation with same shuffled sequence, the experiment result is shown in the Table 4.9 and **??**. Time Series Random Forest (TS-RF) is Random Forest with 9 selected time series features in Section 4.3.8, $TS-MLP_{all}$ is time series data structure with MLP, $TS-SVM$ is the baseline from Ma et al. work [29], $TS-SVM_{all}$ is $TS-SVM$ with all features which I mentioned above, $TS-SVM_{Credit}$ is $TS-SVM$ with CreditScore, $TS-SVM_{SpikeM}$ is $TS-SVM$ with epidemiological features.

As shown in the Table 4.9, our model TS-RF has best performance in all case over time and CreditScore can significantly improve the performance of time series model in the first 24 hours of event. For example in the first hour, it improve the accuracy of the model from 0.65 of $TS-SVM$ to 0.71 of $TS-SVM_{Credit}$. And it is at least the second best feature in all case over time.

TS-RF VS Static Models

Firstly, we compare our time series model with the normal static feature model. We show the result in Table 4.10 the full 48 hours details in Appendix Table .2 and in Figure 4.10. As we can see from the result that the accuracy of time series model overall is better than the static model. But after 32 hours the advantage of the time series is very limited. The reason may be after 32 hours the static model already has enough data to ignore the offset of features at the different time points. But the time series model still has benefits for detecting rumors at early stage of rumor spreading.

Model	Accuracy in hours								
	1	6	12	18	24	30	36	42	48
$TS - RF$	**0.8**	**0.86**	**0.86**	**0.87**	**0.87**	**0.87**	**0.88**	**0.88**	**0.91**
$TS - MLP_{all}$	0.67	0.74	0.74	0.78	0.77	***0.81***	0.79	0.80	***0.82***
$TS - SVM$ [29]	0.65	0.69	0.75	0.77	0.77	0.78	***0.80***	***0.81***	0.81
$TS - SVM_{Credit}$	***0.71***	0.74	0.76	***0.79***	***0.80***	0.79	0.79	0.78	0.78
$TS - SVM_{Epi}$	0.67	0.73	0.73	0.73	0.75	0.75	0.74	0.76	0.77
$TS - SVM_{SpikeM}$	0.69	0.70	0.75	0.75	0.76	0.75	0.77	0.77	0.76
$TS - SVM_{all}$	0.74	***0.78***	***0.80***	0.78	0.77	0.76	0.76	0.75	0.75
$SVM_{static} + Epi$ [16]	0.60	0.69	0.71	0.72	0.75	0.78	0.75	0.78	0.81
$SVM_{static} + SpikeM$ [23]	0.58	0.68	0.72	0.73	0.77	0.78	0.78	0.79	0.77
SVM_{static} [50]	0.62	0.70	0.70	0.72	0.75	0.80	0.79	0.78	0.77

Table 4.9: Accuracy Of Different Models Over Time

Hour	TS-RF	Static model	Difference
1	0.82	0.78	0.03
6	0.86	0.8	0.06
12	0.87	0.83	0.04
18	0.88	0.83	0.05
24	0.87	0.85	0.01
30	0.88	0.85	0.03
36	0.87	0.86	0.01
42	0.88	0.88	0
48	0.89	0.87	0.02

Table 4.10: Accuracy: Time Series VS Static Features

Feature Analyzing Over Time

In this subsection, we will present the performance of features changing over time. We rank the features' importance by using the method we introduced in Section 2.4.3, the full result is shown in appendix Table .1. First we split the features in 7 catalogues as in Table 4.5: *Tweet_Feature*, *User_Feature*, *Text_Feature*, *CreditScore*, *SpikeM Features*, *Epidemiological Features*, *CrowdWisdom* and the *BestSet*. The *BestSet* is a combination of the top 9 most important features which is mentioned in Section 4.3.8. The results over 48 hours are in Figure 4.11 .

As we can see in Figure 4.11 the best result on average over 48 hours in the *BestSet* with top 9 features. Second one is the *All features*. Except those two the best group feature is *Text feature*. One reason is the text feature set has the largest group of feature with totally 16 features. But if look into each feature in text feature group we can see the best and the worst features are all in this set. *User features*

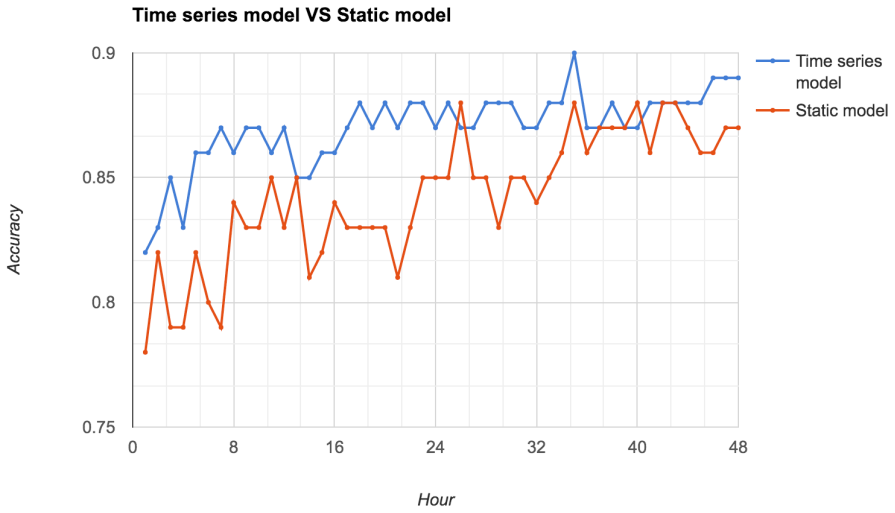


Figure 4.10: Accuracy: Time Series VS Static Features

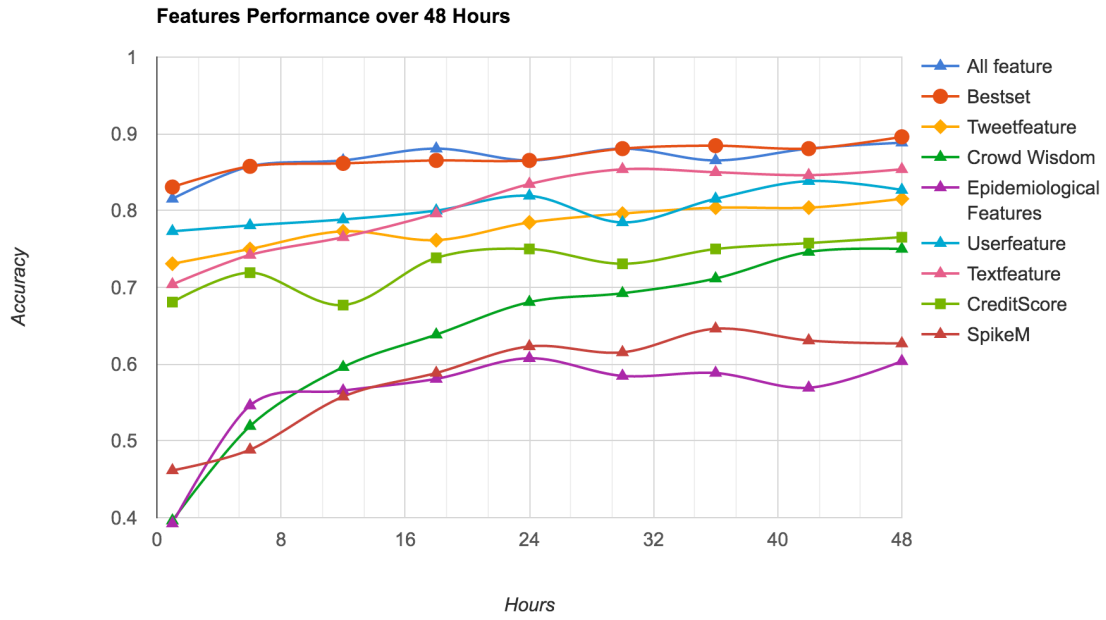


Figure 4.11: Accuracy: Time Series VS Static Features

set and *Twitter features set* are stable over time around 82%. The performances of 3 different models (SIS, SEIZ and SpikeM) describing the propagation pattern of rumors and news are not ideal especially within 24 hours. *CrowdWisdom* and *CreditScore* both contain only one feature, but they already have impressive result comparing with the *User feature* and *Twitter feature*.

BestSet Features

BestSet is a features set of 9 most important features which we mentioned in Section 4.3.8. On average *BestSet* has the best performance.

Text Features

Text features set contains totally 16 features. The ranks of feature as shown in Table 4.11. The best one is *NumOfChar* which is the average number of different characters in tweets.

PolarityScores is the best feature when we tested the single tweets model, but its performance in time series model is not ideal. It is true that rumor contains more negative sentiment, but in an event (rumor or news) people can show their mixed views about this event [33] [43] like discussing or denying, so the *PolarityScores*'s performance becomes worse and worse over time. *Text feature* overall is the the best feature set.

Features	**Ranks**									
Hours	1	6	12	18	24	30	36	42	48	AVG
NumOfChar	3	3	4	3	3	8	7	6	4	4.29
QuestionExclamation	25	16	2	1	1	1	3	7	5	4.79
Question	15	11	13	7	5	4	8	5	8	8.29
LengthOfTweet	6	6	9	6	14	16	13	16	13	11.96
PolarityScores	12	15	23	28	33	33	34	31	32	28
Stock	34	44	47	47	47	47	47	47	48	46.15
Smile	35	45	45	48	48	48	48	48	48	47.06
Sad	36	46	46	49	49	49	49	49	49	47.9

Table 4.11: Rank of Part of Text Feature

Twitter Features

The performance of *Twitter feature* is stable over time from the beginning to the end.

The 3 best of *Twitter Features* are all the features about the contained ULR in tweet: *ContainNEWS, UrlRankIn5000, WotScore* showing in Table 4.12. It is quite reasonable that the news event would have higher probability to be reported by news websites or higher ranked website. And it is clear to see that their performances significantly improve after 24 hours.

But the other original Twitter functions like the retweets or mention do not contribute much.

Features	Ranks									
Hours	1	6	12	18	24	30	36	42	48	AVG
ContainNEWS	8	4	5	4	4	2	2	2	2	3.48
UrlRankIn5000	14	13	7	11	7	3	1	4	6	5.96
WotScore	4	10	6	10	10	6	9	8	7	7.63
Mention	13	5	10	14	13	10	12	12	10	10.98
Hashtag	20	20	15	18	16	13	15	17	17	17.46
Retweets	21	21	27	38	42	35	31	37	34	33.25

Table 4.12: Rank of Part of Twitter Feature

User Features

The performance of *user features* is similar with the *Twitter features*, they are both quite stable from the first hour to the last hour. But one difference is in the first few hours *user feature* is the second best feature set after the *AllFeatures set*.

As shown in Table 4.13, the best feature over 48 hours of user feature is *UserTweetsPerDays* and it is the best feature overall in the first 4 hours, but its rank decreases with time going by. Others user features like UserReputationScore and UserJoinDate also has a better performance in the first fews hours.

That means the sources (the poster in the first few hours) of news and rumors are quite different with each other. But with more and more users joining in the discussion, the bias of two groups of users becomes less. After 6 hours, we can better distinguish the rumors basing on the content of the tweet *(text features)* than basing on the feature of the User.

Features	Ranks									
Hours	1	6	12	18	24	30	36	42	48	AVG
UserTweetsPerDays	0	1	1	2	2	9	5	10	14	4.63
UserReputationScore	1	2	3	5	6	5	6	3	3	5.06
UserJoin_date	5	8	8	8	12	14	16	11	9	10.58
UserVerified	24	17	12	16	17	12	11	14	19	16.25

Table 4.13: Rank of Part of User Feature

SpikeM Features and Epidemiological Features

The performances of these two feature sets are not so convincing. The feature P_α from the SpikeM is the best one of them. The problem of these two models which we have already figured out in Section 4.3.4 is that two models need enough data to fit the parameters. After 24 hours, model with epidemiological features with

SpikeM can reach 60% accuracy. In other words before 24 hour these is no clear propagation pattern of these events. In the work of Kwon et al. [23], the durations of dataset which he uses are more than 60 days. In the work of Jin et al. [16], they uses 160 hours' tweets' volume to fit the SEIZ models. Their data's durations are far larger than ours 48 hours.

P_a parameter from SpikeM is the only feature barely has some contributions for rumor detection in our experiment. It stands for the strength of periodicity in SpikeM. Kwon et al. add 3 more parameters Q_a,Q_p and Q_s to explain the periodicity of the external shock, but they do not produce same effect in our experiment, because 48 hours time period is too short to contain multi-pike pattern.

Features	**Ranks**									
Hours	1	6	12	18	24	30	36	42	48	AVG
P_a	29	28	34	30	33	24	23	21	23	25.75
R_{SI}	47	24	30	23	36	39	38	24	30	29.56
β_{SIS}	49	30	33	28	31	36	28	33	25	30.15
Q_a	44	47	47	21	38	40	44	41	33	38.04

Table 4.14: Rank of Part of SpikeM Features and Epidemiological Features

CreditScore

CreditScore is the output pre-trained Single Tweet Credibility Scoring model's in Section 3.4.2. As shown in Table 4.15, excepting the first 4 hours *CreditScore* is the best feature overall. In Figure 4.12 we show the result of model without *CreditScore* feature and model with full features set. Before the first 24 hours the model without *CreditScore* has worse performance than the full feature set, so the *CreditScore* feature contributes much for the early stage of rumor detecting.

Hour	**Rank**
1	2
2	1
3	1
4	1
5	0
..	0
48	0

Table 4.15: Ranks of CreditScore

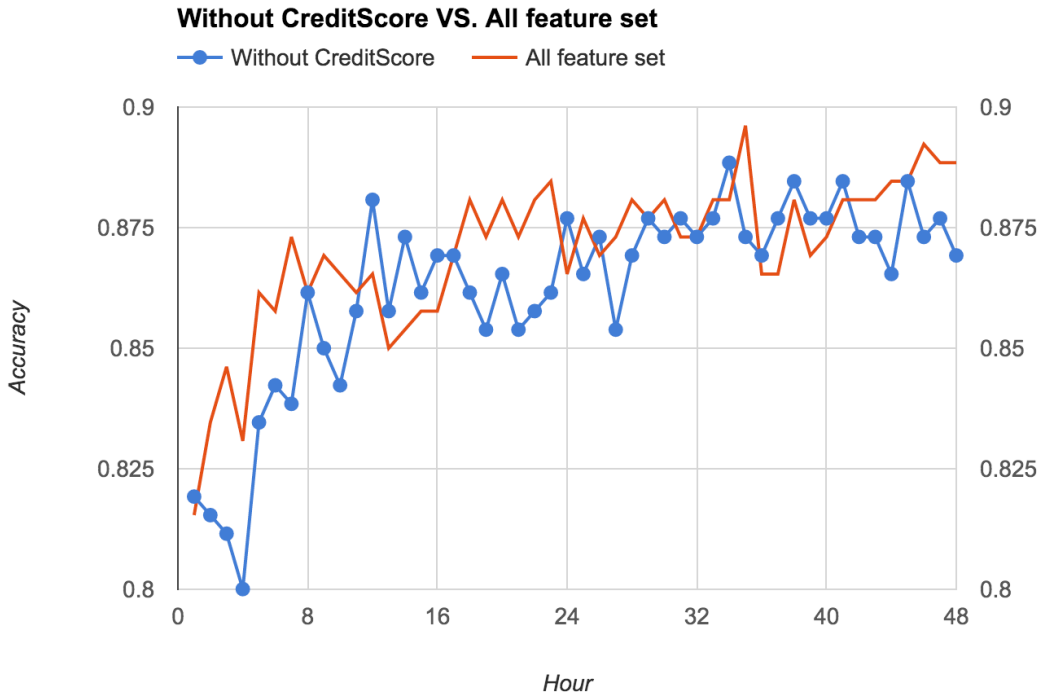


Figure 4.12: Accuracy: Time Series VS Static Features

Features	Ranks									
Hours	1	6	12	18	24	30	36	42	48	AVG
CreditScore	1	0	0	0	0	0	0	0	0	0.08
CrowdWisdom	34	38	21	14	8	5	5	2	2	13.18

Table 4.16: Rank of Part of CreditScore and CrowdWisdom

CrowdWisdom

CrowdWisdom is also a good feature which can get 75.8% accuracy as a single feature. But its performance is very poor (less than 70%) in the first 32 hours. The crowds need time to unify their views to the event after absorbing all kinds of information. That is also one reason why do we need this automatic detecting system.

Machine vs Human

Our system would be meaningless, if the system detects the rumors later than the human verifying them. In this section, we will compare our model with the human rumor debunking website: **snopes.com** and **urbanlegend.com**.

Snopes.com has their own Twitter account[8]. They will post tweets via this account about rumors which they collected and verified. We consider the creation time of the first tweet which contains the keyword *"snopes"* or *"urbanlegend"* in the text or in URLs is the time stamp of human confirming rumors.

But some of the rumors have a long duration, the website may report it several months ago or later the biggest burst peak. For example in Figure 4.13 it is a rumor about the rapper Tupac Shakur, who is thought to have been killed in 1996, is alive and comes out of hiding. This topic bursted in 2012, 2015 and 2016 several times and the tweets' volume of 2012 is the highest, so t_{max} is defined in 2012. But "snopes.com" reported this rumor in the september 2015[9]. So we think that they don't refer to the same rumor affair.

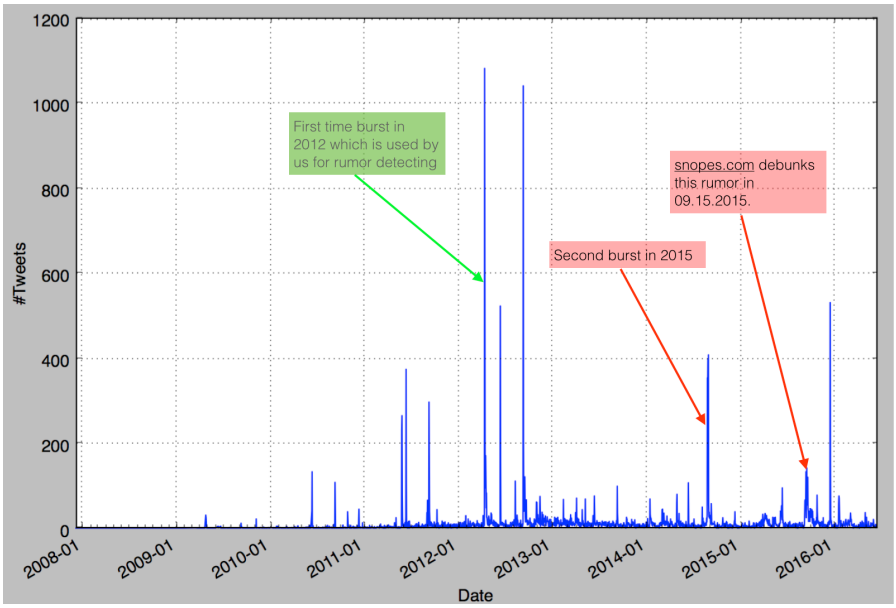


Figure 4.13: Tweet Volume Of Rumor about Tupac Shakur

So we set up a threshold 72 hours. We only consider the first tweet containing snopes in text or URL within 72 hours before or after the beginning time of the event as time stamp of human confirming rumors. We show two examples here. First one in Figure 4.14 is a rumor about okra Curing diabetes[10] which we detected the beginning time is 01.31.2014 04:00. So we scan the first tweet about snopes and we find it in 01.28.2014 21:00 which is 55 hours earlier than the beginning time. Snopes didn't explain their source of this rumor, maybe they detect the story not from Twitter. Other example in Figure 4.15 is that human detect rumor 71 hour

[8] https://twitter.com/snopes
[9] http://www.snopes.com/media/notnews/tupac.asp
[10] http://www.snopes.com/medical/homecure/okra.asp

after the event beginning. The result is shown in Table 4.17. On average the editors of "snopes.com" need 25.49 hours to verify the rumors and post it. Our system already achieves 87% accuracy in 25 hours.

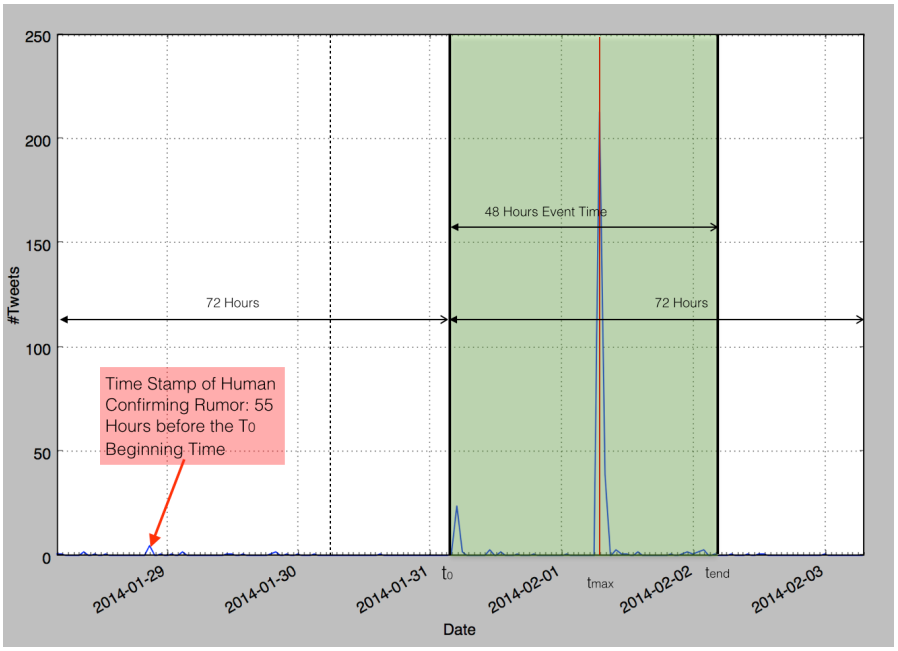


Figure 4.14: The Earliest Time Stamp Of Human Confirming Rumor

	Hours
Latest Time of Human Detection	71
Earliest Time of Human Detection	-55
Average	25.49

Table 4.17: Time of Human Confirming Rumors

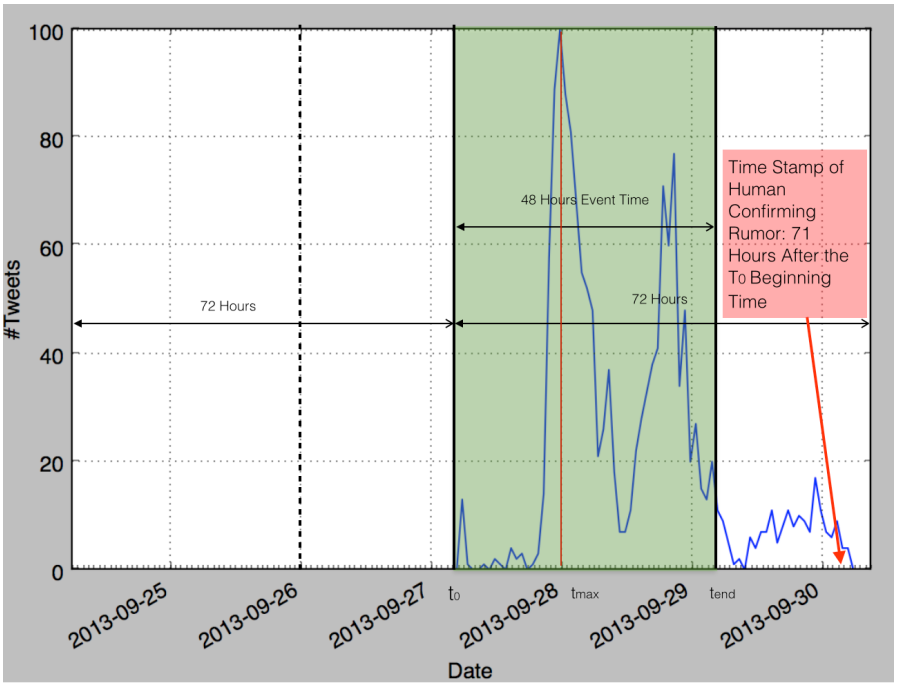


Figure 4.15: The Lastest Time Stamp Of Human Confirming Rumor

4.4.4 Discussion

Why sentiment doesn't work so well in time series model

PolarityScores is the best feature of single tweet model. But it dose not work well in time series. Generally the performance of *PolarityScores* drops down over time. In Table 4.18 we reference the result from Thomas Heverin's work [13]. He analyzed the tweets which responded to the 2009 Violent Crisis. The number of tweets containing original information and emotion are less and less over time. In the other hand, more and more people start to share their different opinions even technology problems after 24 hours. That may make difference of sentiment features between rumors and news less and less over time and after 24 hours it becomes useless.

Time Period	Information	Opinion	Technology	Emotion	Action	Other
0-12hours	90.0%	6.8%	1.1%	5.6%	1.1%	0.0%
12-24 hours	86.6%	13.0%	3.1%	4.5%	1.3%	0.7%
24-36 hours	73.9%	18.3%	7.0%	2.7%	0.5%	3.7%
26-48 hours	74.6%	21.3%	1.0%	3.8%	0.5%	2.8%

Table 4.18: . Percentage of Tweet Type (Non-Exclusive) Per 12 Hour Time Period (Source: [13])

Performance of External URLs features

In this section, We will discuss the performance of external urls features. An interesting phenomenon of external urls features is their performances is much better 24 hours later after the beginning of the events.

In our experiment, we have 3 features about the external URLs: *ContainNEWS*, *UrlRankIn5000* and *WotScore*. It is clear to see in Table 4.12 that after 24 hours the performances of these features are better than the performances before 24 hours.

In Alexander's work [34], he also shows similar phenomenon that credibility of the information from Twitter is higher than external website in the first 24 hours. That may be the reason why the features of external links performance better after 24 hours. 4.16.

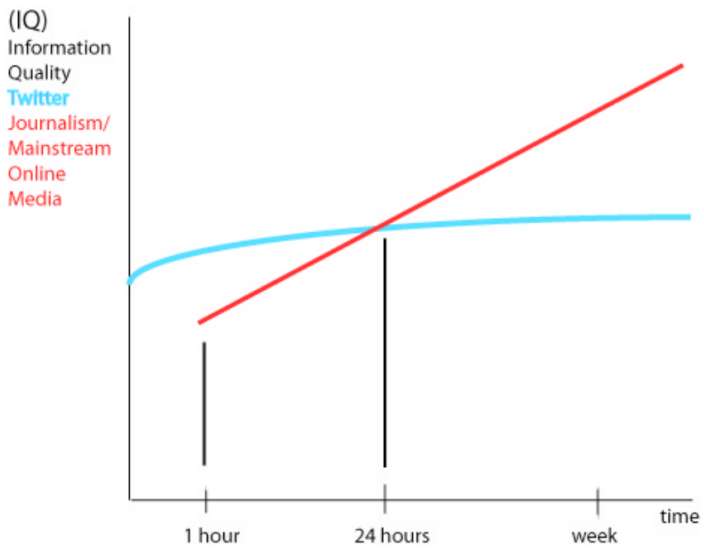


Figure 4.16: Information Quality Over Time (source: [34])

Case Study: Munich Shooting

In this section, We will present a case study of Munich shooting. We will show the time line of event and the the performance of our model in the early stage of event. Finally we will discuss some example of the misclassification of our single tweet's credibility model.

The time of the event:

- At 17:52 CEST a shooter opened fire in the vicinity of the Olympia shopping mall in Munich. 10 people, including the shooter, were killed and 36 others were injured[11].
- At 18:22 CEST first tweet was posted. It might contain some delay here, because our query is constructed in English and maybe the very first tweets are in German. The tweet is *"Sadly, i think there's something terrible happening in #Munich #Munchen. Another Active Shooter in a mall. #SMH"*.
- At 18:25 CEST the second tweet was posted: *"Terrorist attack in Munich????"*.
- At 18:27 CEST the traditional media (BBC) posted their first tweet. *"'Shots fired' in Munich shopping centre - http://www.bbc.co.uk/news/world-europe-36870800a02026 @TraceyRemix gun crime in Germany just doubled"*.

[11] https://en.wikipedia.org/wiki/2016_Munich_shooting

- At 18:31 CEST, the first misclassified tweet is posted. It was a tweet with shock sentiment and swear words: *"there's now a shooter in a Munich shopping centre.. What the fuck is going on in the world. Gone mad"*.

We show the *CreditScore* of Munich Shooting event in Figure 4.17. It is clear to see that the event Munich Shooting has higher *CreditScore* than the average of news events. But we labeled all the tweets about Munich shooting as news related, but because this event contains a large tweets' volume and there must be some rumors spreading with it. It may cause the single tweet credibility model unreliable. But we use the average CreditScore of the event as features, it can limit the influence of the defects of ground-truth.

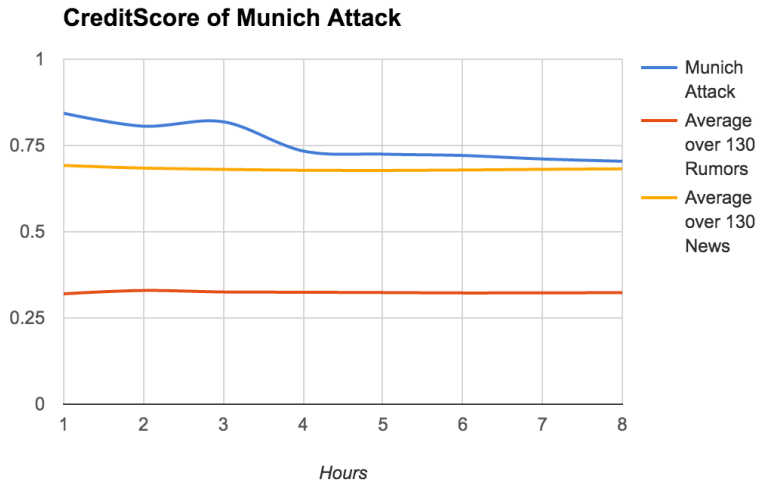


Figure 4.17: CreditScore of Munich Shooting Comparing With Average CreditScore of 130 Rumors and 130 News

There are several kinds of misclassification shown in Table 4.19: unrelated to the events, strong emotional and rumor related:

- We searched the tweet in English, so most users are in America not in Germany who can talk about some extend topics like gun law. These comments are labeled as rumors by our system.
- And some tweets contain strong personal emotion even swear words. These tweets have higher probability being labeled as rumor.
- The third type is rumor. Some rumors were spreading with the news events. For example some tweets claimed Sam Hyde was the shooter and posted a photo with an armed man on it. Or some claimed the shooter was a member of ISIS. They are also detected by our system. But generally the average CreditScore is more like a news event.

Catalogue	**Examples of Tweets**
Event Unreleted	Looks like the EU's gun ban has continued to do its job. What a success. #Munich https://twitter.com/RT_com/status/756525863093538817
	The strict gun laws in Munich kept guns out of innocent hands, didn't stop the terrorists, in fact made their job easier. @realDonaldTrump
	@KummersTim @ShepNewsTeam Shep Smith is colluding with Hillary's camp also what he tried during Munich attack was disgusting #TrumpPence16
Strong Emotional	Munich Another day another attack. when is this shit gonna end. It is becoming the norm now. Saddening .
	there's now a shooter in a Munich shopping centre.. What the fuck is going on in the world. Gone mad
Rumors	ISIS On Munich Terror Attack: Everything Hurting Infidels Makes Us Happy http://www.weaselzippers.us/285113-isis-on-munich-terror-attack-everything-hurting-infidels-makes-us-happy/ via @WeaselZippers
	Obama released photo of shooter #Munich pic.twitter.com/GzJkyNpYDP
	Nice attack 7 days ago, Wurzburg axe attack Monday, Alps knife attack on Wednesday & Munich shootings ongoing. All are Jihad, get used to it
	New info: Munich shooter has been consuming high amounts of a chemical substance called ẗH2Oï #banH2O #banChemistry
	@M7madSmiry @TheBpDShow hearing reports that the shooter is a white supremacist... Has that been confirmed? #Munich
	Witness In Munich Shooting Says: The Shooter Cried Out Allahu Akbar As He Slaughtered Children http://shoebat.com/2016/07/22/witness-in-munich-shooting-says-the-shooter-cried-out-allahu-akbar-as-he-slaughtered-children/ via @walidshoebat
others	@ThatTimWalker seems you were wrong re the Munich attack.

Table 4.19: Example of Misclassification by Single Tweet Model on Munich Shotting

The second best feature is ContainNews which is the percent number of the URLs containing News Website. We show the ContainNews of Munich attack in Figure 4.18. We can see the curve of ContainNews of Munich shooting event is closer to the curve of ContainNews average News events.

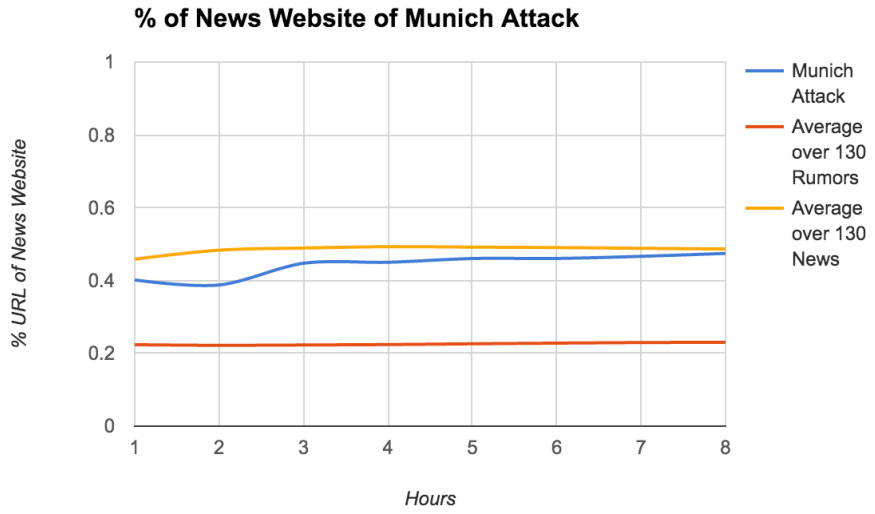


Figure 4.18: % of Tweet Containing News Websites of Munich Shooting Comparing With Average ContainNews of 130 Rumors and 130 News

5 Outlook

5.1 Conclusion

With the developing of social media, Twitter becomes an important platform for exchanging information. But the problem of the low information credibility has troubled all users of Twitter. Without the complex process of verification like traditional media, Twitter users can easily publish breaking news or their opinions, but they also can produce many rumors. Our work is using the temporal features to detecting rumor on Twitter. But at the first few hours after the beginning of the events, the tweets' volume is limited and there is no propagation features yet, so we can only focus on the information of each single tweets. So we develop a single tweet credibility scoring model. We follow the idea of zhou et al. which uses neural network to resolve short text classification task which gets 81% accuracy. We use the output of single tweet credibility scoring model as a new feature CreditScore which is the best feature in our experiment and it can improve the performance of our time series model $TS - RF$.

And we analyze the performance of each features over 48 hours. The CreditScore is the at least the second best feature in all case over time and ContainNews is the second good feature. Sentiment features are useless after 25 hours. And the CrowdWisdom feature starts effect only after 24 hours. And there is no obvious difference of the verified users' behavior in a rumor event or a news event. Famous people are not guarantees of truth. Features of propagation models, no matter SpikeM, SIS or SEIZ are not ideal. Because we limited the time period of events within 48 hours, the propagation pattern is still not so clear.

5.2 Future Work

In the future work we can do following works:

- Because of limitation of time, we crawled only 260 events in Twitter. We can extend it in the future.
- The tweets of a event are all labeled as rumor or news for single tweet credibility scoring model. But in the news event there are rumors or low credibility tweets, on other hand in rumor there are also denying tweet or high credibility tweets. In the future, we can manually label every single tweet to increase the ground truth.
- The time period in our experiment is constant 48 hours and the interval time is 1 hour. In the future work we can use dynamic time period with variable length of time intervals.
- Because of limitation of time and ability, we test only several parameters' combination of neural network. We can optimize the models with more parameters' combination.

Bibliography

[1] F. Ahmed and M. Abulaish. An mcl-based approach for spam profile detection in online social networks. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 602–608. IEEE, 2012.

[2] G. W. Allport and L. Postman. The psychology of rumor. 1947.

[3] Y. Bao, C. Yi, Y. Xue, and Y. Dong. A new rumor propagation model and control strategy on social networks. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1472–1473. ACM, 2013.

[4] L. Barbosa and J. Feng. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 36–44. Association for Computational Linguistics, 2010.

[5] L. M. Bettencourt, A. Cintrón-Arias, D. I. Kaiser, and C. Castillo-Chávez. The power of a good idea: Quantitative modeling of the spread of ideas from epidemiological models. *Physica A: Statistical Mechanics and its Applications*, 364:513–536, 2006.

[6] J. Borge-Holthoefer and Y. Moreno. Absence of influential spreaders in rumor dynamics. *Physical Review E*, 85(2):026116, 2012.

[7] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[8] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[10] C. Castillo, M. Mendoza, and B. Poblete. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM, 2011.

[11] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[12] A. Gupta, P. Kumaraguru, C. Castillo, and P. Meier. Tweetcred: Real-time credibility assessment of content on twitter. In *International Conference on Social Informatics*, pages 228–243. Springer, 2014.

[13] T. Heverin and L. Zach. *Microblogging for Crisis Communication: Examination of Twitter Use in Response to a 2009 Violent Crisis in the Seattle-Tacoma, Washington, Area*. ISCRAM, 2010.

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[15] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65. ACM, 2007.

[16] F. Jin, E. Dougherty, P. Saraf, Y. Cao, and N. Ramakrishnan. Epidemiological modeling of news and rumors on twitter. In *Proceedings of the 7th Workshop on Social Network Mining and Analysis*, page 8. ACM, 2013.

[17] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

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

[19] D. Kimmey. Twitter event detection. 2015.

[20] A. Kohut and M. Remez. Internet overtakes newspapers as news outlet. *Pew Research Centre*, 2008.

[21] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[22] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang. Aspects of rumor spreading on a microblog network. In *International Conference on Social Informatics*, pages 299–308. Springer, 2013.

[23] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108. IEEE, 2013.

[24] S. Lai, L. Xu, K. Liu, and J. Zhao. Recurrent convolutional neural networks for text classification. In *AAAI*, pages 2267–2273, 2015.

[25] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[26] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944.

[27] X. Liu, A. Nourbakhsh, Q. Li, R. Fang, and S. Shah. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM, 2015.

[28] 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.

[29] 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 the 24th ACM International on Conference on Information and Knowledge Management*, pages 1751–1754. ACM, 2015.

[30] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. In Q. Yang, D. Agarwal, and J. Pei, editors, *KDD*, pages 6–14. ACM, 2012.

[31] C. Matthews. How does one fake tweet cause a stock market crash. *Wall Street & Markets: Time*, 2013.

[32] A. J. McMinn, Y. Moshfeghi, and J. M. Jose. Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 409–418. ACM, 2013.

[33] M. Mendoza, B. Poblete, and C. Castillo. Twitter under crisis: can we trust what we rt? In *Proceedings of the first workshop on social media analytics*, pages 71–79. ACM, 2010.

[34] A. Mills, R. Chen, J. Lee, and H. Raghav Rao. Web 2.0 emergency applications: How useful can twitter be for emergency response? *Journal of Information Privacy and Security*, 5(3):3–26, 2009.

[35] O. Oh, K. H. Kwon, and H. R. Rao. An exploration of social media in extreme events: Rumor theory and twitter during the haiti earthquake 2010. In *ICIS*, page 231, 2010.

[36] C. Olah. Understanding lstm networks. *Net: http://colah. github. io/posts/2015-08-Understanding-LSTMs*, 2015.

[37] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. *ICML (3)*, 28:1310–1318, 2013.

[38] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599. Association for Computational Linguistics, 2011.

[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[40] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards. *Artificial intelligence: a modern approach*, volume 2. Prentice hall Upper Saddle River, 2003.

[41] E. Seo, P. Mohapatra, and T. Abdelzaher. Identifying rumors and their sources in social networks. In *SPIE defense, security, and sensing*, pages 83891I–83891I. International Society for Optics and Photonics, 2012.

[42] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[43] K. Starbird, J. Maddock, M. Orand, P. Achterman, and R. M. Mason. Rumors, false flags, and digital vigilantes: Misinformation on twitter after the 2013 boston marathon bombing. *iConference 2014 Proceedings*, 2014.

[44] C. R. Sunstein. *On rumors: How falsehoods spread, why we believe them, and what can be done*. Princeton University Press, 2014.

[45] Y. Tanaka, Y. Sakamoto, and T. Matsuka. Transmission of rumor and criticism in twitter after the great japan earthquake. In *Annual Meeting of the Cognitive Science Society*, page 2387, 2012.

[46] R. Thomson, N. Ito, H. Suda, F. Lin, Y. Liu, R. Hayasaka, R. Isochi, and Z. Wang. Trusting tweets: The fukushima disaster and information source credibility on twitter. *Proc. of ISCRAM*, 10, 2012.

[47] R. M. Tripathy, A. Bagchi, and S. Mehta. A study of rumor control strategies on social networks. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1817–1820. ACM, 2010.

[48] A. H. Wang. Don't follow me: Spam detection in twitter. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1–10. IEEE, 2010.

[49] K. Wu, S. Yang, and K. Q. Zhu. False rumors detection on sina weibo by propagation structures. In *2015 IEEE 31st International Conference on Data Engineering*, pages 651–662. IEEE, 2015.

[50] F. Yang, Y. Liu, X. Yu, and M. Yang. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, page 13. ACM, 2012.

[51] Z. Zhao, P. Resnick, and Q. Mei. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1395–1405. ACM, 2015.

[52] C. Zhou, C. Sun, Z. Liu, and F. Lau. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015.

Appendix

time	CreditScore	ContainNEWS	NumChar	UserTweetsPerDays	QuestionExclamation	UserReputationScore	UrlRankIn5000	WotScore	Question	UserJoin_date	Mention	LengthOfTweet	DebunkingWords	UserFollowers	Exclamation	UserVerified	Hashtag	Capital	You	UserNumPhoto	numUrls	UserFriends	NumPositiveWords	Via	P_a	UserIsInLargeCity	PolarityScores	UserDescription	R_{SI}	β_{SIS}	ξ_{SEIZ}	NumNegativeWords	P_s	b_{SEIZ}	β_{SEIZ}	Retweets	p_{SEIZ}	α_{SIS}	Q_s	P_p	ρ_{SEIZ}	I	Qp	l_{SEIZ}	Q_a	HeShe	IsRetweet	Stock	Smile	Sad
1	2	8	3	**0**	25	1	14	4	15	5	13	6	33	7	27	24	20	10	28	9	22	11	18	30	29	19	12	23	45	42	39	17	31	43	46	21	40	41	47	44	49	16	38	37	48	26	32	34	35	36
2	1	5	2	**0**	18	3	11	4	12	7	15	6	43	8	36	22	21	14	40	9	30	10	23	34	25	24	13	38	31	28	41	16	29	27	20	19	39	32	48	26	35	17	47	33	49	37	42	44	45	46
3	1	4	2	**0**	17	5	12	3	10	8	11	6	44	9	38	24	23	15	41	7	30	13	20	32	25	22	14	39	21	26	27	18	36	37	35	19	29	33	48	28	40	16	47	31	49	34	42	43	45	46
4	1	5	3	**0**	16	2	13	4	14	7	11	9	43	6	39	34	21	12	30	8	37	10	23	32	26	25	15	41	28	33	31	17	19	22	27	24	38	40	48	18	29	20	47	35	49	36	42	44	45	46
5	**0**	4	3	1	16	2	13	10	11	8	5	6	42	7	38	17	20	14	33	9	36	12	22	28	35	30	15	41	31	24	23	18	32	25	34	21	29	19	48	37	40	27	47	26	49	39	43	44	45	46
6	**0**	4	2	1	14	3	11	9	8	10	7	6	41	5	35	29	22	15	34	12	40	16	17	21	28	25	18	42	24	30	13	19	36	20	27	26	23	37	48	33	31	32	47	39	49	38	43	44	45	46
7	**0**	6	2	1	7	3	11	12	10	8	4	5	23	9	22	28	21	15	39	13	37	14	20	19	27	26	18	42	24	38	17	16	34	35	25	31	29	41	48	30	33	32	47	40	49	36	43	44	45	46
8	**0**	4	2	1	5	3	7	9	8	10	13	6	24	11	18	15	21	14	41	12	34	19	20	31	28	17	22	40	23	29	27	16	25	35	33	26	32	37	48	30	36	38	47	39	49	42	43	44	45	46
9	**0**	8	2	1	4	3	7	5	10	6	13	9	25	11	15	14	23	12	32	16	28	19	22	24	26	33	20	37	31	30	18	17	41	35	29	21	27	36	48	39	34	38	47	40	49	42	43	44	45	46
10	**0**	8	3	1	2	4	5	10	11	6	12	7	18	9	17	19	20	13	32	14	26	24	23	27	29	36	16	39	41	30	21	15	35	28	34	31	22	25	48	38	37	33	47	40	49	43	42	44	45	46
11	**0**	5	4	1	2	3	7	6	13	8	10	9	19	11	17	12	15	14	24	16	18	20	25	31	33	32	23	40	26	29	36	21	37	39	22	27	38	30	48	35	42	28	47	34	49	41	43	44	45	46
12	**0**	6	3	1	2	4	7	5	13	8	9	11	18	12	16	15	17	10	21	20	23	14	28	27	34	25	19	41	30	33	36	31	40	26	37	24	32	22	48	35	29	39	47	38	49	42	43	44	45	46
13	**0**	7	3	1	2	4	9	5	12	6	8	11	15	13	22	14	17	10	35	18	20	16	31	27	36	24	23	43	40	30	26	34	37	28	32	21	29	19	42	33	25	38	39	41	44	45	46	47	48	49
14	**0**	6	3	1	2	4	5	9	12	7	8	11	15	13	17	14	22	10	26	19	18	16	27	36	34	29	21	39	24	23	25	33	41	28	32	20	40	42	31	35	30	44	38	37	43	46	45	47	48	49
15	**0**	5	4	1	2	3	6	13	8	7	11	10	12	16	9	15	27	14	20	26	19	18	22	25	32	34	30	40	24	37	17	21	38	33	41	28	35	44	29	39	23	43	36	42	31	46	45	47	48	49
16	**0**	4	3	2	1	5	7	10	11	9	12	6	13	17	8	15	21	14	20	22	23	24	43	36	18	30	29	39	33	37	16	28	41	26	38	32	31	34	35	44	19	42	40	27	25	45	46	48	47	49
17	**0**	4	3	2	1	5	11	10	7	8	14	6	12	17	9	16	18	13	15	21	20	19	44	26	35	43	28	31	22	32	36	34	37	39	40	38	30	23	25	41	27	33	24	29	42	45	46	47	48	49
18	**0**	5	3	2	1	4	9	8	7	10	12	11	14	16	6	15	17	13	18	24	20	19	40	26	30	32	29	35	23	28	25	31	41	27	38	37	33	36	34	43	39	44	22	42	21	45	46	47	48	49
19	**0**	4	3	2	1	7	8	9	6	11	12	10	13	15	5	18	17	14	16	24	19	20	36	26	22	38	29	43	28	33	35	30	23	32	25	41	27	39	34	40	37	42	21	31	44	45	46	47	48	49
20	**0**	3	4	2	1	7	5	6	8	13	12	11	10	16	9	17	18	15	14	21	20	23	35	33	28	30	41	29	22	27	39	36	37	43	32	31	25	24	38	40	42	46	19	26	34	45	44	48	47	49
21	**0**	4	3	2	1	7	6	8	10	14	12	13	5	17	9	18	15	16	11	27	20	19	30	35	24	33	29	28	25	22	44	37	26	31	36	38	40	21	39	32	42	43	23	41	34	46	45	47	48	49
22	**0**	4	3	2	1	7	6	8	11	14	12	13	5	18	9	17	15	16	10	20	21	22	26	33	24	36	30	28	35	25	38	27	23	37	41	43	31	19	40	42	29	39	34	44	32	46	45	47	48	49
23	**0**	4	3	2	1	6	7	10	5	12	13	14	9	15	8	17	16	18	11	19	24	23	34	35	25	43	33	32	27	38	29	39	21	41	28	42	37	40	22	30	36	46	20	26	31	44	45	47	48	49
24	**0**	3	5	2	1	6	4	14	7	13	10	11	9	17	8	16	15	19	12	20	18	26	25	24	33	32	30	22	36	31	35	37	23	39	41	40	42	38	21	27	34	45	29	43	28	44	46	47	48	49
25	**0**	3	4	2	1	10	7	9	5	13	11	14	6	17	8	19	18	15	12	22	16	30	23	20	37	26	29	24	21	32	31	40	41	28	39	42	36	35	27	38	43	44	25	34	33	45	46	47	48	49
26	**0**	4	6	2	1	8	3	9	7	12	14	13	5	18	11	16	15	17	10	22	20	24	19	32	23	28	36	21	26	37	31	34	27	33	43	44	29	35	25	39	41	38	40	42	30	46	45	47	48	49
27	**0**	3	8	4	1	7	2	9	6	15	12	13	5	17	14	16	11	18	10	22	21	29	19	25	23	24	33	20	26	30	36	31	34	42	27	44	38	40	28	32	41	37	45	39	35	43	46	47	48	49
28	**0**	1	7	9	2	8	3	6	4	14	10	15	5	17	13	16	11	21	12	23	19	27	20	26	35	24	29	18	32	30	22	38	25	28	41	42	44	31	36	37	43	40	34	39	33	46	45	47	48	49
29	**0**	2	8	9	1	5	3	6	4	14	10	16	7	17	15	12	13	22	11	23	21	24	19	25	18	32	33	20	31	34	27	36	28	44	39	35	41	38	26	29	42	40	37	43	30	46	45	47	48	49
30	**0**	3	8	7	1	9	2	5	4	14	10	16	6	17	13	15	12	20	11	22	21	31	18	23	24	25	29	19	39	26	27	32	36	38	33	34	43	37	35	28	44	30	42	41	40	45	46	47	48	49
31	**0**	3	9	4	2	7	1	5	6	16	10	15	8	20	13	14	12	21	11	22	18	24	23	29	17	34	28	19	25	26	30	37	31	36	35	32	44	39	27	46	43	38	41	40	33	45	42	47	48	49
32	**0**	2	6	4	3	7	1	9	5	16	10	15	8	17	12	14	13	21	11	22	18	24	23	30	19	26	33	20	28	25	32	42	31	38	34	27	36	44	40	41	43	39	35	29	37	45	46	47	48	49
33	**0**	2	4	7	3	8	1	6	11	13	10	15	5	17	12	9	14	19	16	22	20	30	21	24	25	26	42	18	29	23	36	37	33	35	32	27	43	31	28	40	34	39	41	38	46	44	45	47	48	49
34	**0**	1	5	3	4	6	2	8	12	16	9	15	7	13	11	10	17	23	14	24	20	27	21	25	19	28	36	18	37	26	32	33	22	39	34	31	45	29	38	40	35	42	43	41	30	44	46	47	48	49
35	**0**	2	7	5	3	6	1	9	8	16	12	13	4	10	14	11	15	24	17	21	19	38	20	22	27	23	34	18	32	30	37	33	28	25	26	31	40	45	29	42	36	35	44	43	39	41	46	47	48	49
36	**0**	2	6	7	5	4	1	8	9	14	12	15	3	13	10	11	18	25	16	24	19	30	21	20	23	22	32	17	38	28	31	37	27	39	26	36	42	45	33	29	40	35	34	43	44	46	41	47	48	49
37	**0**	1	6	9	3	5	2	7	8	12	11	14	4	16	10	13	18	27	15	23	17	26	21	24	20	22	32	19	39	25	28	34	30	43	29	41	45	33	38	31	35	40	36	44	37	46	42	47	49	48
38	**0**	1	6	8	4	7	3	5	9	10	14	16	2	15	12	11	18	26	13	22	23	30	24	19	21	20	35	17	29	31	38	44	28	25	32	37	46	36	33	27	34	40	41	45	39	42	43	48	47	49
39	**0**	1	5	10	4	7	3	6	9	13	14	16	2	8	12	11	17	27	15	23	24	32	21	20	22	19	39	18	25	26	33	37	28	29	36	41	34	35	38	30	31	45	40	46	42	43	44	47	48	49
40	**0**	1	7	13	4	6	3	8	5	11	12	17	2	9	14	10	16	27	18	26	24	28	23	20	22	19	33	15	45	31	38	40	25	36	39	32	34	41	21	30	35	46	37	43	29	42	44	47	49	48
41	**0**	2	6	10	7	3	4	8	5	11	12	16	1	9	15	14	17	29	13	21	23	26	25	18	22	19	31	20	32	35	40	42	27	33	28	37	44	46	24	30	41	39	36	34	38	45	43	47	48	49
42	**0**	1	4	11	8	3	6	7	5	9	10	16	2	12	14	13	19	27	15	20	25	29	23	18	21	17	35	22	24	33	40	45	26	30	32	43	36	31	39	37	38	42	34	46	41	44	28	47	49	48
43	**0**	2	3	11	7	4	6	8	5	9	10	15	1	16	12	14	17	28	13	21	23	30	24	18	22	19	32	26	25	31	35	45	36	27	20	44	37	42	29	43	41	38	33	34	39	46	40	47	48	49
44	**0**	2	4	10	5	6	7	8	3	9	11	15	1	14	12	16	18	26	13	23	22	27	24	20	19	17	31	21	25	29	37	40	36	32	30	44	38	43	35	39	28	46	41	33	34	45	42	47	49	48
45	**0**	2	3	10	6	4	7	8	5	9	11	16	1	14	12	15	19	28	13	22	20	30	24	18	23	17	31	21	26	33	41	36	42	25	29	39	27	46	35	43	45	40	34	37	32	44	38	47	49	48
46	**0**	2	5	11	4	3	6	7	8	9	12	17	1	16	10	18	15	25	13	23	21	35	26	14	24	19	29	20	30	32	40	44	33	22	34	38	27	46	28	42	43	45	39	41	31	37	36	48	49	47
47	**0**	2	4	13	3	6	5	7	8	9	11	15	1	19	10	18	16	25	12	22	20	37	24	17	21	14	33	23	29	27	41	36	32	26	30	45	38	44	28	43	42	35	34	40	31	46	39	48	47	49
48	**0**	2	4	14	5	3	6	7	8	9	10	13	1	18	11	19	17	24	12	22	20	35	27	15	23	16	32	21	30	25	43	40	38	29	26	34	28	42	31	45	36	37	39	46	33	41	44	48	47	49
AVG	0.10	3.48	4.29	4.63	4.79	5.06	5.96	7.63	8.29	10.58	10.98	11.96	12.35	13.63	15.15	16.25	17.46	18.65	19.15	19.65	22.85	23.54	24.73	25.42	25.75	26.13	28.00	28.06	29.52	30.00	31.67	31.69	31.81	32.46	32.65	33.35	35.27	35.54	35.75	35.83	36.50	37.31	37.60	37.96	38.58	42.81	43.08	46.15	47.06	47.90

Hour	Time series model	Static model	Difference
1	0.82	0.78	0.03
2	0.83	0.82	0.01
3	0.85	0.79	0.05
4	0.83	0.79	0.04
5	0.86	0.82	0.04
6	0.86	0.8	0.06
7	0.87	0.79	0.08
8	0.86	0.84	0.03
9	0.87	0.83	0.04
10	0.87	0.83	0.04
11	0.86	0.85	0.01
12	0.87	0.83	0.04
13	0.85	0.85	0
14	0.85	0.81	0.04
15	0.86	0.82	0.03
16	0.86	0.84	0.02
17	0.87	0.83	0.04
18	0.88	0.83	0.05
19	0.87	0.83	0.05
20	0.88	0.83	0.05
21	0.87	0.81	0.06
22	0.88	0.83	0.05
23	0.88	0.85	0.04
24	0.87	0.85	0.01
25	0.88	0.85	0.02
26	0.87	0.88	-0.01
27	0.87	0.85	0.02
28	0.88	0.85	0.04
29	0.88	0.83	0.04
30	0.88	0.85	0.03
31	0.87	0.85	0.02
32	0.87	0.84	0.03
33	0.88	0.85	0.03
34	0.88	0.86	0.02
35	0.9	0.88	0.02
36	0.87	0.86	0
37	0.87	0.87	0
38	0.88	0.87	0.02
39	0.87	0.87	0
40	0.87	0.88	0
41	0.88	0.86	0.02
42	0.88	0.88	0
43	0.88	0.88	0
44	0.88	0.87	0.01
45	0.88	0.86	0.02
46	0.89	0.86	0.03
47	0.89	0.87	0.02
48	0.89	0.87	0.02

Table .2: Time seriers VS static Features in detail

time	All Feature	BestSet	Tweet Feature	Crowd Wisdom	SIR	User Feature	Text Feature	CreditScore	SpikeM
1	0.815	**0.831**	0.731	0.396	0.392	0.773	0.704	0.681	0.462
2	0.835	**0.838**	0.727	0.396	0.562	0.762	0.685	0.712	0.515
3	0.846	**0.850**	0.769	0.423	0.565	0.758	0.708	0.708	0.492
4	0.831	**0.854**	0.762	0.477	0.538	0.765	0.738	0.712	0.550
5	**0.862**	0.854	0.758	0.508	0.554	0.781	0.746	0.692	0.500
6	**0.858**	**0.858**	0.750	0.519	0.546	0.781	0.742	0.719	0.488
7	**0.873**	0.854	0.754	0.573	0.554	0.769	0.754	0.700	0.554
8	0.862	0.862	0.773	0.573	0.542	0.754	0.758	0.692	0.542
9	**0.869**	0.858	0.754	0.573	0.565	0.785	0.754	0.700	0.588
10	0.865	**0.881**	0.758	0.588	0.565	0.788	0.746	0.708	0.573
11	0.862	0.869	0.769	0.585	0.538	0.781	0.735	0.708	0.542
12	**0.865**	0.862	0.773	0.596	0.565	0.788	0.765	0.677	0.558
13	0.850	**0.873**	0.754	0.588	0.554	0.796	0.785	0.692	0.550
14	0.854	**0.865**	0.777	0.600	0.573	0.800	0.762	0.708	0.542
15	0.858	**0.865**	0.777	0.619	0.546	0.804	0.777	0.731	0.550
16	0.858	**0.862**	0.773	0.627	0.565	0.792	0.788	0.754	0.565
17	**0.869**	0.858	0.777	0.635	0.538	0.792	0.788	0.735	0.577
18	**0.881**	0.865	0.762	0.638	0.581	0.800	0.796	0.738	0.588
19	**0.873**	0.862	0.773	0.646	0.585	0.808	0.819	0.742	0.608
20	**0.881**	0.862	0.773	0.662	0.604	0.812	0.815	0.746	0.596
21	**0.873**	**0.873**	0.785	0.669	0.592	0.808	0.838	0.742	0.600
22	**0.881**	0.873	0.788	0.685	0.600	0.819	0.827	0.746	0.588
23	**0.885**	0.865	0.785	0.677	0.592	0.815	0.838	0.731	0.608
24	**0.865**	0.865	0.785	0.681	0.608	0.819	0.835	0.750	0.623
25	**0.877**	**0.877**	0.769	0.692	0.600	0.819	0.842	0.746	0.623
26	0.869	**0.881**	0.785	0.696	0.581	0.800	0.846	0.735	0.600
27	0.873	**0.888**	0.788	0.696	0.612	0.804	0.854	0.735	0.600
28	**0.881**	0.881	0.788	0.692	0.596	0.796	0.846	0.735	0.612
29	**0.877**	**0.877**	0.781	0.688	0.596	0.792	0.854	0.727	0.608
30	**0.881**	**0.881**	0.796	0.692	0.585	0.785	0.854	0.731	0.615
31	0.873	**0.888**	0.800	0.692	0.604	0.785	0.850	0.735	0.588
32	0.873	**0.892**	0.808	0.696	0.588	0.792	0.850	0.746	0.631
33	0.881	**0.892**	0.808	0.700	0.581	0.812	0.846	0.762	0.642
34	0.881	**0.896**	0.804	0.704	0.558	0.808	0.854	0.735	0.658
35	**0.896**	0.881	0.815	0.712	0.585	0.804	0.854	0.731	0.658
36	0.865	**0.885**	0.804	0.712	0.588	0.815	0.850	0.750	0.646
37	0.865	**0.885**	0.796	0.715	0.577	0.812	0.862	0.727	0.646
38	**0.881**	0.873	0.804	0.738	0.577	0.808	0.846	0.754	0.646
39	0.869	**0.888**	0.800	0.731	0.573	0.827	0.850	0.742	0.635
40	0.873	**0.885**	0.808	0.735	0.588	0.819	0.854	0.758	0.646
41	0.881	**0.892**	0.804	0.738	0.577	0.812	0.854	0.758	0.631
42	**0.881**	**0.881**	0.804	0.746	0.569	0.838	0.846	0.758	0.631
43	0.881	**0.888**	0.812	0.750	0.600	0.838	0.854	0.742	0.627
44	0.885	**0.896**	0.815	0.754	0.577	0.835	0.858	0.746	0.600
45	0.885	**0.900**	0.804	0.754	0.577	0.835	0.858	0.738	0.646
46	**0.892**	0.888	0.804	0.754	0.612	0.835	0.854	0.731	0.662
47	0.888	**0.904**	0.819	0.754	0.596	0.827	0.858	0.750	0.619
48	0.888	**0.896**	0.815	0.750	0.604	0.827	0.854	0.765	0.627

Table .3: Accuracy of Different Feature Sets over 48 Hours

List of Figures

1.1 pipeline of the rumor detecting system 2

2.1 Decision Tree 7
2.2 An Example of Random Forest with 3 Trees 8
2.3 An Example of 1 Layer Neural Network 9
2.4 tanh Function 10
2.5 CNN for Text Classification (source from[18]) 11
2.6 2 Layer Recurrent Neural Network 11
2.7 Multi-input Single-output Recurrent Neural Network 12
2.8 Single-input Multi-output Recurrent Neural Network 12
2.9 Multi-input Multi-output Recurrent Neural Network 12
2.10 Two Models of RNNs (a) Normal Model of RNN with tanh Unit (b) RNN with LSTM Cells (source [36]) 13
2.11 GRU Cell (source [36]) 13
2.12 Dropout Neural Net Model. (a) a Standard Neural Network (b) after Deploying Dropout 14
2.13 K Fold Cross Validation (green rectangle is the training set, red rectangle is the testing set) 15

3.1 Sample of Users' Information on Twitter Interface 20
3.2 Full Scale Tweets' Volume of Event Robert Byrd 24
3.3 tweet volume of the rumor event of Robert Byrd after selected time period 24
3.4 Neural Network Model for Single Tweet Classification 26
3.5 The Architecture of C-LSTM Interface (source: [52]) 27

4.1 The Fraction of Tweets Containing URL with Top 5000 Domain 34
4.2 The Fraction of Poster Living in Large City 34
4.3 SIS Model 37
4.4 SEIZ Model 38

4.5 Fitting results of SIS and SEIZ model of (a) Rumor: Robert Byrd was a member of KKK (b) Rumor: CNN altered a photograph of a shooter making him look white (c) News: Doctor announces Michael Schumacher is making process (d) News: Two U.S. sailors are arrested over an alleged rape of a Japanese woman on Okinawa 40
4.6 Fitting Results of SIS and SEIZ Model with Only First 10 Hours Tweets' Volume (same 4 stories as above) 41
4.7 Fitting Results of SpikeM Model of (a) Rumor: Robert Byrd was a member of KKK (b) Rumor: CNN altered a photograph of a shooter making him look white (c) News: Doctor announces Michael Schumacher is making process (d) News: Two U.S. sailors are arrested over an alleged rape of a Japanese woman on Okinawa 45
4.8 Fitting results of SpikeM Model with First 10 Hours data (same stories as above) 46
4.9 Accuracy Changing with Number Of Features 48
4.10 Accuracy: Time Series VS Static Features 53
4.11 Accuracy: Time Series VS Static Features 53
4.12 Accuracy: Time Series VS Static Features 57
4.13 Tweet Volume Of Rumor about Tupac Shakur 58
4.14 The Earliest Time Stamp Of Human Confirming Rumor 59
4.15 The Lastest Time Stamp Of Human Confirming Rumor 60
4.16 Information Quality Over Time (source: [34]) 62
4.17 CreditScore of Munich Shooting Comparing With Average CreditScore of 130 Rumors and 130 News 63
4.18 % of Tweet Containing News Websites of Munich Shooting Comparing With Average ContainNews of 130 Rumors and 130 News 65

List of Tables

3.1 Features for Single Tweet's Creditability Scoring Model 21
3.2 Tweet Volume of News and Rumors 22
3.3 Parameters of Classification Models 27
3.4 Prediction Accuracy of Different Single Tweet's Creditability Scoring Models 28
3.5 Features Importance 29
3.6 Example of False Classification by Single Tweet Model 30
3.7 Example of Correct Classification by Single Tweet Model 31

4.1 Parameters of SEIZ 39
4.2 Parameters of SpikeM 43
4.3 New Parameters of Extended SpikeM 44
4.4 Best Features 48
4.5 Features of Time Series Rumor Detection Model 49
4.6 Categories Of Rumors And News 50
4.7 Time Span of News and Rumors 50
4.8 Parameters of Classification models 51
4.9 Accuracy Of Different Models Over Time 52
4.10 Accuracy: Time Series VS Static Features 52
4.11 Rank of Part of Text Feature 54
4.12 Rank of Part of Twitter Feature 55
4.13 Rank of Part of User Feature 55
4.14 Rank of Part of SpikeM Features and Epidemiological Features 56
4.15 Ranks of CreditScore 56
4.16 Rank of Part of CreditScore and CrowdWisdom 57
4.17 Time of Human Confirming Rumors 59

4.18. Percentage of Tweet Type (Non-Exclusive) Per 12 Hour Time Period (Source: [13]) 61
4.19 Example of Misclassification by Single Tweet Model on Munich Shotting 64

.1 All feature's importances over 48 hours 76
.2 Time seriers VS static Features in detail 77
.3 Accuracy of Different Feature Sets over 48 Hours 78