AN INFORMATION SYSTEM FOR RUMORS CHECKING

by

Hao Xu

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Master of Science in Electrical and Computer Engineering

Summer 2018

AN INFORMATION SYSTEM FOR RUMORS CHECKING

by

Hao Xu

Approved: _____
Hui Fang, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: _____
Kenneth E. Barner, Ph.D.
Chair of the Department of Electrical and Computer Engineering

Approved: _____
Babatunde A. Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____
Douglas J. Doren, Ph.D.
Interim Vice Provost for Graduate and Professional Education

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor Prof. Fang for the constant help of my study and related research. Thanks to her patience, motivation, and immense knowledge, I made significant progress in studying and research. Not only did she guide me how to do research, but also taught me how to think problems in analytical and critical way.

I would also like to thank my colleagues Peilin Yang, Yue Wang, Kuang Lu and Ye Wang who have helped me during my research. They gave me lots of advices on research and helped me figure out lots of problems.

Last, but not least, I would like to thank my family, for the continuous support and encouragement throughout my time in graduate school.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The rapid development of the Internet has already helped the social media become a significant player as sources for news. However, due to the lack of supervision, social media is also becoming the fertile land for the spread of malicious rumors, which primarily emerges during breaking news. The malicious damage they do to individuals and society is enormous when they spread online. This thesis develops an information system for checking rumors. The system could automatically extract candidate rumors from tweets, and the average distance between extracted candidate rumors and target claims is 0.37. By leveraging the stance classification method, our system could use an alternative way that utilizes the stances of claims on candidate rumors from different information to help users to check rumors. Experiment results show that this method could get the same results on the Snopes website[1] in most cases. The extraction of claims is implemented through parsing tweets based on dependency parser for tweets, merging similar claims into same groups by using clustering methods, and selecting representative claims from groups as candidate rumors based on proposed features. The stance classifier used in this thesis is proposed by Augenstein et al.[1]. It was state-of-the-art stance classification among the SemEval 2016 Task 6.

To evaluate our rumor exploration system, we tested it on thirty-one events representing about 84,297 tweets in total. Twenty two events of them are selected from Snopes website and their hashtags on Twitter are: #BandyLee, #JackBreuer,

[1] https://www.snopes.com

#Gabapentin, #SanctuaryCities, #Capriccio, #JetLi, #SantaFeShooting, #WhereAreTheChildren, #immigrants, #Ingraham, #ItsJustAJacket, #SouthwestKey, #pavingforpizza, #CanadianDoctors, #TrumpKimSummit, #JoeJackson, #RobertDeNiro, #AnthonyBourdain, #dogjealousy, #Irma, and #TrumpRally, #TrumpSalary. Our system collects tweets of these twenty-two events. Besides, nine events of total events are from PHEME dataset[14]. These nine events include Charlie Hebdo Shooting, Eblo-Essien, Ferguson Shooting, German Wings Crash, Gurlitt, Ottawa Shooting, Prince-Toronto, Puttin Missing, and Sydney Siege Hostage. The data of these nine events had been collected and labeled by the journalists. Among twenty-two events from the Snopes website, our system can precisely extract the meaningful claims embedded in tweets of twelve events with the average distance of 0.37 to claims shown on the Snopes website. Besides, the meaningful claims of eight out of nine events in the PHEME dataset are extracted. Thus, among the thirty-one events, meaningful claims of twenty events are retrieved by our system. Moreover, most of the results about candidate rumors inferred from our system are as same as those present on the Snopes website and those labeled in the PHEME dataset. Based on the evaluations of twenty events, the ability of our system is competitive with that of the Snopes website, which contents generated by professional persons.

Chapter 1

INTRODUCTION

Now more than ever, the social media platforms are used as the primary tool for people to get news. They provide the opportunity for people to take part in news by sharing their thoughts and being reporters. The free of charge, ease-of-use, and accessibility of social media platforms help news spread faster than ever before. However, the features that lead them to be used for good things may also lead them to be convenient for bad things. Due to the unsupervised and unconstrained nature of the social media platforms, many rumors may spread together with useful information about an event on them and involve lots of people to discuss them. Thanks to the openness of the Internet, people could publish their own opinions about unconfirmed claims on different platforms. Some of the opinions from on platform might support the claims while others from another platform might disapprove of the claims. These different opinions could be valuable information for us to determine whether these claims are rumors or not. An example of such a phenomenon is the rumor about gabapentin. On 28 June 2017, Kaiser Health News reported on an apparent increase in abuse of gabapentin (brand name Neurontin) in Ohio. The claims that gabapentin is "the most dangerous drug in America" gained widespread exposure thanks to a viral 1 January 2018 Facebook[1] post that alleged, without sources, that gabapentin is the "newest" killer prescription. On Twitter, most of the tweets show the similar claim that is "gabapentin may be the new non-opioid drug of abuse". However, based on the relevant information from NewsAPI, gabapentin is a treatment of epilepsy in the United States. It is also approved to treat neuropathic pain as well. Besides, gabapentin is

[1] http://archive.is/ujmsz

not a narcotic like an opioid, and it has only an indirect effect on the central nervous system, making a direct overdose from the drug alone. Based on the Snopes website[2], this rumor is false.

Therefore, it is crucial to design a system that can automatically explore and detect as many rumors embedded in a massive amount of data propagated on social media platforms as possible. Some current rumor detection approaches based on supervised learning need lots of annotated data, which require experts to perform this annotation, to make the judgments about claims[47, 29]. Besides, Tolosi et al.[32] found it challenging to distinguish rumors and non-rumors as features change dramatically across different events. Thus, we propose a method that presents sufficient information related to candidate rumors and leaves the final judgments to users. In this thesis, we propose and implement the system that can automatically explore the claims as candidate rumors from tweets and present the stances of related claims and relevant news to users to let them have sufficient information to judge whether candidate rumors are rumors or not.

1.1 Rumour Definition and Types

Before we start to introduce the details of our system, we need to provide a clear and reasonable definition of rumor. The definitions of rumor differ from one research literature to another. Some of them define rumor as statements that are deemed false, but it is inappropriate for rumor detection because if the veracity of statements has been determined the problem will become fact-checking which is unrelated to the rumor detection. In this thesis, we want to expand the definition of rumor based on the dictionaries and the survey [46] to make it tailored to our research. In Oxford English Diction, the rumor definition is "a currently circulating story or report of uncertain or doubtful truth" [3]. Also, Merriam Webster dictionary defines it as "a statement

[2] https://www.snopes.com/fact-check/gabapentin-newest-prescription-drug-killer/

[3] http://en.oxforddictionaries.com/definition/rumour

or report current without known authority for its truth" [4]. Finally the Zubiaga et al.[46] defines it as "an item of circulating information whose veracity is yet to be verified at the time of posting." Given the above definitions of rumors, we can find the characteristic feature of the rumor is that the veracity of a statement has not been deemed yet. Besides, we can expand this feature, the stances of rumors may be conflict. Hence, in this thesis, we only consider the rumors embedded in statements on social media and adhere to the above definitions of rumor, we make the definition of rumor more specific, which is, the widely spreading claims, which truth is yet to be determined and stances are conflicting, on social media.

There are two different types of rumors that proposed by Zubiaga et al.[46]. Each type of rumor has corresponded approaches to deal with it. The first one is long-standing rumors that are discussed for long periods of time. Since this kind of rumors has already been determined, the rumor detection may not be necessary. By using the known rumors as a priori information, the system can track the rumors and classify the stances of the rumors. The other one is new rumors that emerge during breaking news. Since new rumors may be different from the training data, the system needs to utilize the features that are similar to that of new rumors from past rumors. This thesis only focuses on the new rumors that emerge during hot events from Twitter.

1.2 Motivation

In the absence of the supervision of the social media platforms, there are lots of rumors emerging and polluting the environment of the Internet. These rumors confuse people with misinformation(simply false information) and even harm innocent people with disinformation(deliberately false information). They are making the free and open environment of the Internet chaotic and frightening.

One of the solutions to the rumor is rumor detection, which is the first stage of the rumor classification system proposed by Zubiaga et al.[46]. Rumour detection aims to determine whether the social media posts are rumors or not. Though it is designed

[4] http://www.merriam-webster.com/dictionary/rumor

for identifying new emerging rumors, some of the existing works have been limited to finding rumors known as a priori. Known rumors input to the rumor detection are used as training data in the classifier, which classifies the new data as rumors related to predefined rumours[9][10][29]. These approaches are useful for long-standing rumors instead of new emerging rumors.

There are some approaches for dealing with new emerging rumors. One of the state-of-the-art approaches is proposed by Zubiaga[46]. It built a sequential classifier based on Conditional Random Fields(CRF) to capture the context information of the tweets. However, like many other supervised learning methods, it highly relied on a large, comprehensive, and accurate training data. As far as we know, the work of labeling data as rumor or not is not only time-consuming but labor intensive. Moreover, it needs professional persons who can look up references and have critical thinking to make reasonable judgments. It is a first problem that we meet. The second problem is that even though we get sufficient labeled data at any cost, the performance of the system based on supervised learning methods still may be low. Since the training data may be irrelevant to the testing data, thus the features that found from training data may not be shared with testing data[32]. The last problem is that finding useful claims as candidate rumors from a considerable amount of social media posts is also labor intensive. Taking the Snopes website as an example, it aims to debunk or confirm widely spread urban legends. All the urban legends are selected and verified by professional fact checkers, who need to spend lots of time researching legends, including deciding which legends are worth to explore and finding relevant information.

To solve the first two problems, we propose an alternative method to detect rumors. Instead of building a rumor detection classifier, we can leave the final judgment to users. Since rumor detection is subjective based on the definition of rumor, its veracity is yet to be determined. Besides, our system aims to make users think critically instead of blindly believing one opinion of a candidate rumor from one information source. Thus we can leverage the stance classification method to get the stances of related claims and news about a candidate rumor and then present them to users.

Then users can have useful information to help them make final judgments about candidate rumors. Although the stance classifier is also based on a large number of training data, the work of labeling data for it is much easier than that for rumor detection. Since people only need to determine whether claims agree with or disagree with relevant candidate rumors. It is unnecessary for people to look up references and have background knowledge. Thus this method can save time and labor. The last problem can be solved by automatically extracting claims embedded in tweets about an event with the help of clustering and Natural Language Processing(NLP) techniques.

1.3 Approach and Contributions

This thesis develops methods to extract claims from tweets about an event and then leverages a clustering method to merge similar claims to the same groups. Moreover, one representative claim is selected as candidate rumor for each group. In the end, it uses existed state-of-the-art stance classification approach to classify the claims and news that are relevant to candidate rumors. Specifically, The sentences extracted as candidate claims for each group are achieved through leveraging the dependency parser for tweets to identify the subject, verb, and object contained in each tweet. For claims clustering, we first utilize embeddings for sentences to encode claims and then use the Density-based spatial clustering of applications with noise(DBSCAN) method with the cosine similarity as the metric to cluster similar claims into same groups. In each group, one representative claim would be selected based on features we proposed as the candidate rumor. After ranking the claims with the features, we finally select top five candidate rumors and associated claims to do further analysis. For the stance classification method, the existed approach used bidirectional conditional Long short-term memory(LSTM) to get stances of relevant tweets and news for each candidate rumor.

Based on the above approaches, we can implement the solutions that we proposed effectively. Our contributions are as follows.

1. An automatic function for candidate rumors extraction is proposed, shown to be precise in the experiment.

2. An alternative way that leverages the stances on claims from different information sources is proposed to help users detect rumors.

For the candidate rumors extraction function, experiment results show that our system could extract candidate rumors that are related to target claims in twelve events out of twenty two events on the Snopes website and the average distance between extracted candidate rumors and target claims is 0.37. Besides, the meaningful claims of eight events are retrieved among the nine events in the PHEME dataset. Thus, our system could extract meaningful claims embedded in tweets precisely in most cases. Besides, experiment results demonstrate the possibility of leveraging stances on claims from different information to help users check candidate rumors.

1.4 Thesis Overview

In Chapter 2, we review the related literature. Chapter 3 introduces the system overview. Chapter 4 describes Tweets Crawler. Moreover, Chapter 5 describes the Candidate Rumor Finder subsystem. Then chapter 6 introduces Stance Classification method, and the experiments are done in In Chapter 7. In the end, the conclusion of the current and future work is made in Chapter 8.

Chapter 2

RELATED WORK

This thesis implements the system for Candidate Rumors Finder and Stance Classification about trending events on Twitter. Related work includes the trending events detection, rumor detection, stance classification and veracity classification.

2.1 Trending Events Detections

Twitter has already become one of the fastest-growing social media. The information contained in tweets cover almost all kinds of things related our daily life. Thus it has widely been used as one of the communication tools for spreading trending events. At the meantime, analyzing and tracking this huge amount of user-created content can yield valuable and useful information. Effectively detecting trending events propagated on Twitter is crucial for rumor detection, because rumors need attention and it can use trending events as a medium to quickly spread online and draw attention. Thus, the rumors may always spread with trending events. To track the breaking news occurred on Twitter, Phuvipadawat et al.[28] proposed method to collect, group, rank, and track breaking news on Twitter. This method used reliability, popularity, and freshness as ranking factors to efficiently present breaking news to the mass audience. Besides, Wang[37] utilized the tie-breaking approach in microblog retrieval and implemented it in ranking methods and evaluation measures of microblog retrieval. Lu[19] introduced Wikipedia concepts in tie-breaking to perform ad-hoc microblog retrieval and deployed the Maximal Marginal Relevance(MMR) criterion to summarize relevant tweets. In order to handle millions of events on Twitter, Yang[40] described TSAR (TimeSeries AggregatoR), a robust, scalable, real-time event time series aggregation framework built primarily for engagement monitoring: aggregating interactions with

Tweets, segmented along a multitude of dimensions such as device, engagement type. Though a large number of tweets generated every day, lots of noise may be contained in them. TwitterStand proposed by Sankaranarayanan et al. tried to address this issue by building Bayes classifier to distinguish junk from news[30]. It also considers the structure of network existed on Twitter and geographical feature associated with the tweets. Dynamic is another feature of Twitter, lots of tweets contained different information are emerged and buried in the flood of information. To capture remarkable feature, Lau et al. presented a novel topic modeling-based methodology to track emerging events on Twitter. This approach can deal with dynamic changes in vocabulary to avoid itself growing over time and catch the shift in the topic model to track emerging events effectively[17]. One of the most commonly used features of tweets for events detection is a hashtag since it is developed as a function to grouping on Twitter. It can represent the topic of tweets and make it easier for users to share the same subject and track specific events in real time. By taking full advantage of the hashtag, Tokarchuk et al. proposed a refined adaptive crawling model to detect popular topics and extract more highly relevant data for hot events by monitoring and analyzing the traffic pattern of hashtags [36].

2.2 Rumour Detection

Rumor detection problem is the first stage of rumor classification architecture proposed by Zubiaga et al.[46]. It can be cast into a binary classification. The input is a stream of posts, and the binary classifier needs to determine each post is rumor or not. The key factors are extraction and selection of discriminating features of rumors embedded in posts. Qazvinian et al. explored three striking features of known rumors: content-based, network-based, and microblog-specific memes[29]. Other useful features used in rumour detection include semantic features[31, 34, 42, 43], syntactic features[35, 22, 24, 6] and Twitter specific features[42, 43].

The most challenging problem for this task is how to detect rumors in the new emerging posts. Though lots of work try to tackle this problem, they all have been

limited to finding existing rumours[9][10][29]. Based on this situation, Tolosi et al. tried to verify the difficulty of distinguishing rumors and non-rumours[33]. After analyzing the features of rumors across different events, he found the reason was that the features of rumors in different events kept changing. The first work successfully solved this problem is proposed by Zhao et al.[45]. This approach is based on the assumption that rumors will raise the skepticism among the people. It leverages the predefined regular expression of questions that may occur in the discussion about rumors. The first limitation of this approach is that the regular expressions of questions are not general. The other one is that the assumption cannot cover all situations and lead to a low recall. Another approach proposed by Zubiaga et al. which achieve the satisfied results[46]. Its context-based approach uses Conditional Random Fields as a sequential classifier that captures the changes during events. Hence it can infer whether a statement is a rumor from previous information. While it exploited the context information, the performance of early time is not satisfied enough due to the absence of a priori information. Besides, this approach highly relies on a large, comprehensive, and accurate training data. McCreadie et al. tried to solve this problem by crowdsourcing platform[23]. Based on the crowdsourcing platform, rumors can be identified by annotators who have the high inter-annotator agreement.

2.3 Stance Classification

Stance Classification problems are the third stage of rumor classification architecture defined by Zubiaga et al.[46]. It is designed for identifying how each post is orienting to the related rumors' veracity. Generally, it can be cast as sentiment analysis problem, which detects the stance of each post for the specific target. Most of rumor stance classification studies are implemented in supervised approach. Qazvinian et al[29] proposed the first study that solves the stance classification automatically. It leveraged the content-based features, network-based features, and Twitter specific memes to build a classifier to determine whether the author of each tweet believe the rumor or not. Hamidian et al. introduced the Tweet Latent Vector(TLV) approach,

which can capture the semantic textual similarity, to achieve the better performance. Instead of focusing the classification of tweets in isolation, Zubiaga et al. focused on conversations under each original tweets. The novel approach that considered the context of data is proposed based on Conditional Random Fields as a sequential classifier[46]. Based on the same idea, Kochkina et al. proposed an LSTM-based sequential model based on a conversational structure of tweets[13]. Besides, a new approach exploited both temporal and textual information based on Hawkes Processes is proposed by Lukasik[20]. This approach posited the importance of making use of temporal information existed in tweets.

2.4 Veracity Classification

The veracity classification tried to determine the truth of rumors. It attempted to collect other trustworthy sources, such as news, government websites, and database to make final judgments. Instead of the true value, some other work will provide sufficient extra reliable information to help users make final judgments. Some work tried to identify the believability of the sources of posts instead of the truth of rumours[2, 44, 25]. Liu et al. proposed other features about source include source identification, source diversity, source and witness location, event propagation and belied identification [18]. Kwon et al. proposed a set of features in veracity classification: temporal, structural, and linguistic[16]. Based on the previous work, two extended features proposed by Yang et al. are client-based and location-based[39]. Other features used in veracity include: linguistic[8], characteristics of users[3], sentiment and writing style[5], and repetition[4]. Some models tried to capture the time features, and they built models based on features over time[21, 38].

Chapter 3

SYSTEM OVERVIEW

This chapter introduces each component of the system briefly. For each component, we will describe reasons to implement it and what can it do. Figure 3.1 shows the general workflow of the system. As we can see, it consists of three major parts: Tweets Crawler, Candidate Rumors Finder, and Stance Classifier.

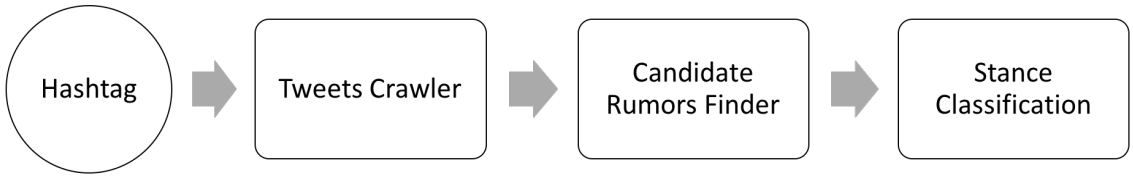

Figure 3.1: The workflow of the rumour collection system. The input of the system is shown in the circle, and the intermediate outputs are omitted.

3.1 Tweets Crawler

To acquire useful information for the system to analyze, we need to collect sufficient data. In this system, the needed tweets are about events since rumors always circulate together with events online. On Twitter, events are associated with hashtags in most cases. Thus, hashtag detection and hashtag selection are significant for the system to focus on popular events that might contain rumors. After a hashtag is selected, the system could collect sufficient relevant tweets by using the hashtag as a query for further analyze. To get enough tweets, we leverage the powerful tool GetOldTweets-python[1], which overcomes the limitations of Twitter API, to help us

[1] https://github.com/Jefferson-Henrique/GetOldTweets-python

crawl the related tweets based on the provided query. This subsystem takes as input the hashtag of an event of interest on Twitter and collects the relevant tweets.

3.2 Candidate Rumors Finder

On Twitter, there are lots of tweets about an event. These tweets convey different information and also hold various claims. Some of these claims, which might not be verified and widely spread online to cause much influence, could be candidate rumors to be analyzed. Thus, the first goal of this sub-system is to extract claims about an event. Moreover, various claims exist in tweets about an event, some of them are similar, and others are different. Hence, another goal is to merge similar claims into the same groups. To implement these two goals, we first calculate the number of clusters based on the topic distribution of tweets generated by LDA and then cluster tweets with K-means clustering algorithm. In each group, claims would be extracted, and relevant information would be acquired. However, after observing the claims in different groups, the differences between the claims of different groups are not obvious. Therefore, we propose an alternative procedure to implement this sub-system. We first use a dependency parser for English tweets[2] to parse the tweets. Based on the structure of a tweet, the sentence which root is verb would be selected to get the subject that is dependent on the root, and the object that is dependent on the root would be extracted based on our proposed rule-based method. The simple sentence consisted of the subject, the root verb, and the object could be used as a claim. We then use DBSCAN to find the distinct claims groups and select one representative claim as the candidate rumor for each group based on the features we proposed. Since more than one groups of claims might exist in an event in most cases, we select top 5 distinct candidate rumors, which are ranked by their features, and associated claims to further analyze. To analyze candidate rumors, we need to inquire them in other information sources to present more relevant information to users. Therefore, we build queries based on candidate rumors and search them on two information sources. We could get

[2] https://github.com/ikekonglp/TweeboParser

google snippets from Google Search as the first additional relevant information. The second additional relevant information is got from the NewsAPI[3]. These two information sources both have pros and cons, and we are going to compare their results and to select one as the final information source. The Candidate Rumor Finder subsystem takes tweets about an event as input then generates candidate rumors and gets relevant information for each group of an event.

3.3 Stance Classifier

For each candidate rumor about an event, relevant claims would show different stances about it. Some of them might support it, while others might disapprove of it. These different stances could be a valuable clue for uninvolved people to know the circumstance and to help them make their judgments about each candidate rumor. Based on this assumption, we utilize the stance classification method to get stances of related claims to each candidate rumor. The Stance Classifier subsystem takes relevant claims as input, then produces their stances to each candidate rumor.

[3] https://newsapi.aylien.com

Chapter 4

DATA COLLECTION

Data Collection is the first part of the system, as can be seen in Figure 1.1, it consists of two parts: the Hashtag and Tweets Crawler. The first part of this section describes how to detect and select an event on Twitter based on Hashtags and the second part introduces a powerful tool that can help us acquire tweets based on queries generated by hashtags.

4.1 Event Detection and Selection

Since we want to build a baseline system, we directly use the hashtags on Twitter to represent events instead of complicated approaches. The hashtag - written with a # symbol - is used to index keywords or topics on Twitter. This function was created on Twitter and allows people to follow topics they are interested in.[1] A hashtag is a straightforward and powerful function provided by Twitter. It also can help us categorize the tweets. Moreover, popular hashtagged words are always shown on Trending Topics provided by Twitter. It is general and concrete enough to be a keyword or phrase used in a query to search tweets that are related to popular events. So the event detection and selection problem can be cast into hashtag detection and selection. The following sections will introduce how to use hashtags to generate a reasonable query to get sufficient tweets related to an event.

4.1.1 Hashtag Expansion

The system provides two ways for users to get events from Twitter: an active way and a passive way. For the active way, a user can input his interested hashtag of

[1] https://help.twitter.com/en/using-twitter/how-to-use-hashtags

an event. Also, for the passive way, the system will automatically acquire a popular hashtag of an event from Trending Topics. However, it is not enough to use a single hashtag of an event as a query if we want to get sufficient tweets that can represent whole events. There are various forms of a hashtag about an event. Taking the Texas Shooting event as an example, the #TexasShotting hashtag is the most popular one on Twitter. However, other forms of it also existed, such as #texashotting and #Texasshooting. It is necessary to consider these similar forms of a hashtag to collect as many related tweets as possible. Besides, lots of particular and new hashtags related to subtopics of an event will emerge over time. These newly generated hashtags may represent more specific aspects of an event or new development of an event. For example, #texaschurchshooting and #GunControlNow were also circulating on Twitter together with #TexasShotting. Therefore, it is crucial for the system to discover these new emerging hashtags of an event. Above all, we need to expand the number of hashtags of an event.

To expand the number of related hashtags for an event, we propose a naive approach to deal with it. First, we get the user's input hashtag or representative and popular hashtag from Trending Topics. Moreover, we retrieve the fixed number, 1,000 for this system, of relevant tweets from Twitter. Then, based on these relevant tweets we can extract the hashtags embedded in them and rank them based on the number of occurrences of each hashtag. In the end, we can build a new query by conjugating the top 10 hashtags to collect all the relevant tweets. For capturing the dynamics of an event, this process will be repeated every day of a week. Following the above example, more relevant hashtags could be added to the query of the Texas Shooting event, such as #TexasChurchMassacre, #SutherlandSpringsShooting, #Texas, #2A, and #TexasStrong. Compared with the extracted tweets based on the initial query, we can observe that the final query can extract much more relevant tweets than initial query.

4.1.2 Hashtag Selection

It seems like we can get a reasonable query based on the proposed approach. However, this procedure will have a severe consequence. Not only will it add remarkable hashtags to a query, but it also appends weakly correlated general hashtags to query. For the above event, many tweets also mentioned Trump, which adds hashtag #Trump to the query. Though Trump was involved in this event, he did not play an important role in it. Given the disjunction way of hashtags connection in a query, the system will collect a huge amount of tweets related to Trump as well. Moreover, these tweets add much noise in data. So it is essential for the system to select relevant hashtags.

To select related hashtags, we propose a straightforward solution to it. We can filter out hashtags whose popularities are more significant that of initial hashtag about an event. The popularity of hashtag can be measured by the number of results of Google Search. Since the number of results of Google search can effectively reflect how many web pages related to hashtags, and it can reveal how popular the hashtags are on the Internet. Therefore, we can get the number of results of Google Search for an initial hashtag and relevant hashtags respectively, and then filter out hashtags which the number of results is an order of magnitude larger than that of the first hashtag. After applying this approach, general hashtags such as #MAGA, #NRA, #Trump, #Texas, #2A, and #texas, would be removed. Thus the query would be more concrete and more related to the event.

4.2 Tweets Crawler

Traditionally, the Twitter Application programming interface(API) is widely used for retrieving tweets for research. The main reason is that it is the most open API service compared with others. Besides, it also provides very detailed documentation[2] of ways to use it, which offers developers access to a Representational state

[2] https://dev.twitter.com/docs

transfer(REST) API to retrieve data from the database, and a streaming API to harvest data in real time. Though it is compelling and useful, it still has some limitations. It only provides 1% of the whole tweets. Also, it only present real-time or recent data, so it is hard to collect data that is older than the last few weeks. To overcome these limitations, we utilize the powerful tool GetOldTweets-python proposed by Jefferson Henrique on Github.

GetOldTweets-python tool leveraged the Twitter Advantage Search function to break the limitations of number and time constraints. By constructing a search Http request, it mimics the human's search behavior on Twitter and automatically scrolls down the web pages to let Twitter load more and more relevant tweets. Then it extracted the useful information of tweets embedded in HTML files of web pages. Without charge and registration, GetOldTweets-python can retrieve tweets that we need in the JSON file. After setting the content of query and time range, it can return tweets with abundant information, including id, permalink, username, text, data, retweets, favorites, mentions, hashtags, and geo. Though this tool is useful and convenient, the format of tweets in HTML files changed a little bit since this tool implemented. So we hack its source code and fix it based on the current format of tweets in HTML files. Now, this tool works perfectly in our system.

Chapter 5

CANDIDATE RUMORS FINDER

In the previous chapter, we have introduced how to leverage the hashtag provided by Twitter to retrieve the relevant tweets of an event. Given the sufficient raw data, we will analyze the data in the next stage of the system: Candidate Rumors Finder. In this stage, the system will preprocess data first, and then cluster the tweets based on their similarities into different groups. In each group, we will extract the claims as candidate rumors and find the relevant information of candidate rumors. Candidate Rumors Finder is a significant part of the system to extract candidate rumors embedded in tweets and to collect additional information.

5.1 Data Pre-processing

Before we further analyze the raw data, we need to clean the raw data. It is well known that garbage in, garbage out. Though tweet only has 140 characters, it contains lots of things other than text, such as website address and emoji. It also includes special forms of text, such as smileys, reversed words, mentions, and hashtags. Since our system only focuses on plain text, these things contained in tweets need to be removed effectively. We utilize an online tool preprocessor implemented by Said Ozcan[1]. However, we keep the term with hashtag because it may contain important information in tweets. To make it normal in tweets, we build a regular expression to detect the term with a hashtag and remove the hashtag #.

[1] https://github.com/s/preprocessor

5.2 Sub-Events Clustering

Since the term with a hashtag is a general representation of an event on Twitter, we assume that tweets collected based on the term with hashtag could contain independent sub-events under a general event. Even though we can find more specific terms with hashtags of an event and these terms with hashtags could help us categorize the tweets, the accuracy of clustering only based on terms with hashtags is low since related terms with hashtags always co-occur in one tweet, we cannot determine this kind of tweets belongs to which sub-event. Hence, more reliable and efficient methods should be considered. To tackle this problem, we decide to use the K-means algorithm to cluster tweets, but the number of clusters k needs to be pre-defined. To get a reasonable number of the clusters, we propose an efficient method based on the topic modeling. The following sub-sections will describe the details.

5.2.1 Data Representation

Since the data needs to be input into the clustering algorithm, it needs to be well represented for calculation. Word embedding could be used to do that. Word embedding is a method to map the words or phrases from vocabulary to vectors of real numbers. There are many different types of word embedding. Naturally, the counter vector will be used to represent the text into vector space. It extracts all unique tokens from corpus to form the vocabulary, and build a vector with the size of vocabulary for each document. The element in the vector for each document is the frequency of each word in the vocabulary. As we can see, the information in the text recorded by a counter vector is the frequency of words in documents, but other meaningful information, such as grammar and semantic meaning are disregarded. To capture the context and semantics of words in a text, we applied a more advanced method Word2vec in this system. Word2vec is a two-layer neural net that processes text, and it can produce a vector space with several hundred dimensions by taking a large corpus of text as input. Each unique word in the corpus is assigned a specific vector in the space, and this vector can competently represent the word since it captures

the semantic meaning of the word by considering the context information surrounding the word in the corpus. After building the word2vec model for tweets, we could get vectors for each term of tweets. Moreover, we need to get the vector for each whole tweet since the basic unit for the clustering algorithm is a tweet. The simplest way to implement that is by averaging word vectors for all words in a tweet. To capture more information about words in tweets, we utilize term frequency-inverse document frequency(TF-IDF) weighting scheme since TF-IDF can reflect how important a word is to a tweet in the corpus.

5.2.2 Number of Cluster

After the data representation, there is still one more step before clustering. The number of clusters, which is represented in symbol K, needs to be defined. To get the reasonable K, we proposed an approach to defining the number of clusters based on LDA.

LDA is a generative statistical model that uses unobserved groups to explain sets of observations and those unobserved groups can explain why some parts of the data are similar. In LDA, the document can be represented as mixtures of various topics with specific probabilities. LDA examines a collection of documents to learn what words would be used to represent the same document. Also, these words with different probabilities form multiple word distributions to represent different topics for a document. These words in word distributions can be very informative. We can rank the words based on their probabilities to see which words are more reasonable to represent this topic, or we can use them to measure the similarity between different topics.

Based on the powerful LDA, our proposed approach works in the following procedures. First, we define a large number of topics in advance, which is 10 in this system, since the LDA needs the pre-defined number of topics as well. For a large number of topics, it is likely that some topics word distributions are similar to each other. Thus, we can decrease the number of topics by measuring the similarity between

different topics in the second step. The metric used in the measurement is the Kullback-Leibler(KL) divergence. KL divergence is a measure of how one probability distribution diverges from the other one. The larger the result of it is, the more different two distributions are. Since it is a distribution-wise asymmetric measure, we will only calculate the KL divergence between two topic word distributions in one orientation, that is from a small topic number to a large one. For example, we have topic0, topic1, and topic2, the KL divergence only will be calculated on combinations of topic0 and topic1, topic0, and topic2, topic1, and topic2. To implement that, we will first construct the combinations between any two topics for a document from a small number to large number. After getting the similarities between any two topic word distributions, we need to set a threshold to determine whether two topics need to be merged or not. To avoid merging topics excessively, we set a strict threshold: the mean of similarities minus standard deviation of similarities. Compared with the threshold, the pair-wise similar topics can be generated. Since they may overlap with each other, we need to merge them. This problem can be solved if we represent them in a graph. Each topic could be a node, and edge could connect pair-wise similar topics. Then, by counting the number of disconnected components in the graph, we can get the final number of topics in a document.

5.2.3 Clustering

After representing the data and defining the number of clusters, the remaining problem is how to cluster data into different groups based on the K-means algorithm.

K-means clustering methodology establishes clusters and clusters centers in a set of unlabeled data. It chooses a desired number of clusters and iteratively adjusts the cluster to minimize the within-cluster variance. The specific procedures are shown in the Algorithm 1. There are many different kinds of similarity metrics in the K-means clustering algorithm, such as Euclidean distance, cosine similarity, Manhattan distance, and so on. Cosine similarity is a measure of similarity between two vectors based on calculating the cosine of the angle between them. The range of the result is

from 0 to 1, and the larger the result is the similar two vectors are. It is a reasonable choice for our system since inputs of the system are vectors of tweets.

Algorithm 1: Keyword extraction by word distance

1 **Given an initial set of cluster centers:**

(I) If $k \geq n$, then the m_2.

(II) If $h \geq j$, then m_1.

2 **Reiterate these steps until covergence is reached**

5.2.4 Problems

Following the above procedures, we did some experiments on datasets. However, based on the results, there are still many similar tweets in separated groups that represent independent sub-events. After analyzing procedures and results, we found some problems. The first one is that the whole content of a tweet contains lots of noises. Thought the tweet has already been preprocessed, the contents except for claims in a tweet could also be noises that impact the results of clustering. The second one is that the embeddings of tweets obtained by simply averaging the embeddings of terms in tweets might be not accurate. Besides, the proposed method of calculating the number of clusters is the third problem. It is hard to define a reasonable threshold that could determine a pair of topic distributions should be merged or not precisely and it might be a key factor to affect the performance of clustering since the pre-defined number of clusters is an important parameter in K-means clustering algorithm.

To solve the above problems, we propose some solutions to them. For the first problem, we could extract the simple sentences which structures are subjects, verbs, and objects. Since the optimal goal of this sub-system is to find claims that could be candidate rumors and we assume that forms of claims are as same as forms of simple sentences. Thus, we could use the dependency parser for tweets to acquire the structures of tweets and to extract claims based on the structures of simple sentences. To tackle the second problem, we could utilize two state-of-the-art methods which

build embeddings for sentences, the Skip-Thought vector[12] and Sent2Vec[27]. In the end, the last problem could be solved by leveraging an alternative clustering method, DBSCAN. The DBSCAN does not need the pre-defined number of clusters and could filter the noises in data. Thus the errors in calculating the number of clusters could be avoided. Above all, the whole procedure of this sub-system is changed to the following way. Instead of clustering tweets in advance, we will extract claims from tweets first. The extracted claims then are clustered by DBSCAN with embeddings for sentences and representative claims for each group would be selected as candidate rumors. In the end, the relevant information to representative claims would be searched from two information sources. The details of the procedure will be explained in the following sections.

5.3 Claims Extraction

In this section, we will extract claims embedded in tweets. Certain claims could be used as candidate rumors in our system since the claim is rudiment of the rumor. Based on the definition of the claim on Oxford Dictionaries, a claim is "an assertion of the truth of something, typically one that is disputed or in doubt."[2]. Thus, due to the similarity between the definitions of rumor and claim, we can conclude that an unsubstantiated claim circulating widely online could be a candidate rumor. SemEval-2017 Task 8 also proposed a shared task where participants analyze rumors in the form of claims in user-generated content[7]. To extract claims, we need to parse the tweets structures first and extract important components to build claims. Due to a large number of claims would exist in tweets and diversity of claims, we first cluster claims into different groups based on their similarities, and then in each group, we only select one representative claim as the candidate rumor based on the popularity of the claim. In the end, we will rank the groups based on the popularity of groups and the top five candidate rumors, and associated claims would be further analyzed. Before

[2] https://en.oxforddictionaries.com/definition/us/claim

introducing how to extract claims, the concerned components of a claim needs to be analyzed first.

5.3.1 Informative Components of Claims

A claim is a sentence consisted of subject, verb, object, adjective, prepositional phrase and so on. Though the sentence has various structures: simple sentences, compound sentences, complex sentences, and compound-complex sentences, generally the core components of them are the subject, the verb, and the object. These three components are like the skeleton of the whole complex sentence. As long as we recognize these three parts, we can determine the whole complex sentence since additional parts will exist between them. We use the SVO skeleton as a phrase to represent these three components of the sentence. The next sub-section will introduce how to use methods in NLP to find SVO skeleton from tweets.

5.3.2 SVO Skeleton Extraction

SVO skeleton extraction is a function that aims to extract the claim which consisted of the subject, the verb, and the object. To identify the subject and the verb and the object of the claim, we need to analyze the structure of the tweet first. We utilize the TweeboParser to parse the tweets.

TweeboParser proposed by Lingpeng Kong et al.[15] is a dependency parser for English tweets, and it is trained on a subset of a newly labeled corpus drawn from the POS-tagged tweet corpus of Owoputi et al.[26], Tweetbank. TweeboParser could predict tweet syntactic structure, which is represented in the CoNLL-U format. The CoNLL-U format[3] includes word index, word form, the stem of a word, Universal part-of-speech tag, Language-specific part-of-speech tag, morphological features, head of the current word, Universal dependency relation, head-deprel pairs, other annotation. Since a tweet often contains more than one utterance, the TweeboParser will generate a multi-rooted graph over the tweet based on the dependencies generated from the head

[3] http://universaldependencies.org/format.html

of the words. Besides, the TweeboParser could exclude hashtags, URLs, and emoticons in tweets precisely since in most cases they have no syntactic function.

To find the short and straightforward claims in the tweets, we only focus on the sentences which parsed tree root is a verb since we assume that the root of the parsed tree of a simple sentence is a verb. After we collect sentences which parsed tree root is a verb, we need to find nouns that are depended on the roots as subjects for the sentences. However, in some cases, the subject is not only a single word but a combination of multiple words. Only when these kinds of words or phrases are treated as a whole, do they refer to specific objects. Moreover, these specific objects carry the specific and useful information. Especially in tweets, this type of subject will often be used due to its conciseness and informativeness. Also, in many cases, a rumor may be related to a specific location, person, or company. Thus, it is critical for the system to capture this kind of subject in tweets to acquire this particular kind of information. In the English language, words can be considered as the smallest elements that have different meanings. Based on their functions and proper positions in a sentence, words can be categorized into different parts of speech. If the system knows the parts of speech of two adjacent words Prof. and BandyLee are both nouns, they can be combined to form a subject. To implement the above method, we could use the universal part-of-speech tag in the result of the parser.

After introducing how to find the subject in the sentences, the remaining part of a claim is the object. The method to find the object is much more complicated than the subject since the subject not only is a single word or a phrase but also could be a clause. Besides, the forms of the object could be varied in different sentences. To handle those problems, we skim through lots of structures of sentences and propose a rule-based method to traverse the parsed tree to find the whole object recursively. We start from the root which part-of-speech tag is V, then try to find the following words depended on it. Based on what we found, there are words with some part-of-speech tag need to be extracted. These part-of-speech includes V(verb), P(post-position), O(pronoun), N(noun), $\hat{}$(proper noun), S(nominal possessive), A(adjective), R(adverb)

and &(conjunction). Words with these part-of-speech tags could also depend on each other so we use the recursive way to find the whole object that may include the words.

In the end, by combining the subject, the verb root, and the object, we could generate sentences. These extracted sentences then could be used as claims for further analysis. However, before the further analysis, these claims still need to be preprocessed. Since there are many similar claims in tweets, we need to cluster these claims into the same group. Moreover, these claims could also be used as a query to find the relevant information from other sources. Thus, the claims need to be as concise as possible. Excluding the whole object, we still need to get the concise object. The compact object could be got with the same procedure as the whole object if we add two restrictions on part-of-speech tags. The first restriction: if verb as the previous word has already existed in a claim, then the adjacent adverb, adjective, and pronoun could not be shown in the claim. The second restriction: if noun as the previous word has already existed in a claim, then the adjacent adjective could not be shown in the claim. For example, the whole object in claim "Capriccio sangria is not giving people HIV." is "giving people HIV" so that the final claim is "Capriccio sangria is giving people HIV.". However, the compact object is "not giving people HIV" so that the final claim is "Capriccio sangria is not giving people HIV.". The claims with a compact object would be used as further analysis, and the corresponding claims with the whole object would be used in the stance classifier to detect their stances.

5.4 Claims Clustering

In this subsection, we will do the first analysis of claims, that is, clustering claims. Among the claims, some of them may convey the same information. To get the similar claims together, we can use the clustering method to group them. One of the popular notions of clusters is groups with small distances between cluster members. Thus the distance between sentences would be an important part of the clustering method.

5.4.1 Sentence Representation

To calculate the distance between sentences, we need to represent sentences for calculation. As mentioned before, we will leverage two existing state-of-the-art work to get the semantic embedding for sentences.

The first one is Skip-Thought vector introduced by Ryan Kiros et al.[12]. It is unsupervised learning of a generic, distributed sentence encoder. It trains an encoder-decoder model that tries to rebuild the surrounding sentences of an encoded passage by using the following text from books. Thus, sentences that have similar semantic and syntactic properties could be mapped to similar vector representations. Besides, Ryan Kiros et al. introduces a simple vocabulary expansion method to encode words that were not seen as part of training. In this work, there is a pre-trained model that is based on BookCorpus[4] could be used. The second one is Sent2Vec proposed by Matteo Pagliardini et al.[27], it is a simple unsupervised model that could compose sentence embeddings. It uses the word vectors along with n-gram embeddings and trains the composition and the embedding vectors themselves. There are three pre-trained models based on the different corpus, including Wikipedia, tweets, and the BookCorpus in this work. As we can see, the first work only has a model that is based on BookCorpus, but we can use the word expansion method to encode words in our dataset into the model. However, the second work has the model based on tweets, which could be good at dealing with tweets. Thus, in our work, we compare these two methods of claims clustering. By looking through the results of clustering based on two methods, the Sent2Vec is better to encode the claims in tweets for clustering than Skip-Thought vector. Thus, we adopt the Sent2Vec in this system.

5.4.2 Clustering

After sentence representation, the remaining problem is how to cluster data into different groups based on the clustering algorithms. Claims for an event could be

[4] http://yknzhu.wixsite.com/mbweb

about different things. Lots of claims may talk about the same thing in the most cases. However, some claims may not share the similarity between with each other. Since the rumor is a claim that is well propagated and could cause discussions, in this system we only consider the first type of claims and treat the second one as noise. Besides, the number of clusters is not known in advance. To handle these two features, we decide to use the DBSCAN as our clustering method.

DBSCAN is a density-based clustering algorithm. Given a set of points in some space, the DBSCAN groups together points that are tightly packed together, and treat as outliers points that lie alone in low-density regions. The specific procedures are shown in the Algorithm 2[5]. There are many different kinds of similarity metrics in DBSCAN clustering algorithm, such as Euclidean distance, cosine similarity, Manhattan distance, and so on. Cosine similarity is a measure of similarity between two vectors based on calculating the cosine of the angle between them. The resulting range is from 0 to 1, and the larger the result is the similar two vectors are. It is a reasonable choice for our system since the input of the system is vectors of tweets.

Algorithm 2: DBSCAN Algorithm

1 A point p is a core point if at least minPts points are within distance ϵ(ϵ is the maximum radius of the neighborhood from p) of it (including p). Those points are said to be directly reachable from p.

2 A point q is directly reachable from p if point q is within distance from point p and p must be a core point.

3 A point q is reachable from p if there is a path p1, ..., pn with p1 = p and pn = q, where each pi+1 is directly reachable from pi (all the points on the path must be core points, with the possible exception of q).

4 All points not reachable from any other point are outliers.

[5] https://en.wikipedia.org/wiki/DBSCAN

5.4.3 Representative Claims and Claims Ranking

After we get the groups of claims, we need to select a representative claim for each group. Besides, since lots of claims groups could be generated, we need to rank them to select some popular claims groups. The measurement of popularity is based on the features of the tweet corresponded with the claim. When the tweets crawler crawl tweets, the relevant information of tweets are also crawled. The information includes the number of favorites, the number of retweets and the number of comment. All these information can be used as features for measurement of popularity of each tweet. Since the number of favorites represents how many users like it, the number of retweets describes the influence of the tweet on Twitter and the number of comments shows how many users participant the discussion. Besides, in some cases, the same tweet will be posted on many different Twitter account. This phenomenon reveals a situation that lots of fake accounts will be generated and manipulated by code to help the information with some specific purposes propagate online to cause influence. Moreover, the occurrence number of tweets could be a clue to detect rumors. Thus, the metric of selecting representative claims is the sum of these four features of each claim in core points of DBSCAN, and the metric of ranking groups is the average of these four features of claims in each group.

5.5 Authoritative Data Collection Acquirement

Since the optimal goal of the system is to let users determine whether a candidate rumor can be confirmed as true, debunked as false, or its true value is still to be resolved based on the stances of relevant information. It is necessary to provide additional information that is related to a candidate rumor from other resources except for Twitter for users. Two information sources will be considered in this system. The first one is Google search, and the second one is NewsAPI. To search for relevant information, a query that is related to a candidate rumor needs to be generated first, then the platform for searching also needs to be determined.

5.5.1 Query Generator

Since the final presentations of the system are stances of information related to each candidate rumor, the acquired information should be closely related to each candidate rumor. Therefore, the query generated in our system is a question form of a candidate rumor instead of being reconstructed based on some terms of a candidate. Based on this kind of query, the searching results could be more relevant and unbiased. To implement the query generator, we utilized the POS tagger to get POS tag for each word of a candidate rumor. There are three types of the verb in question: modal, auxiliary verb and copular verb "be" in any grammatical tense. All these verbs have a corresponded POS tag, respectively. Therefore, based on the POS tag for the verb in a claim, we can build a perfect question with the right auxiliary verb. In Figure 5.1, an example of query generation is shown.

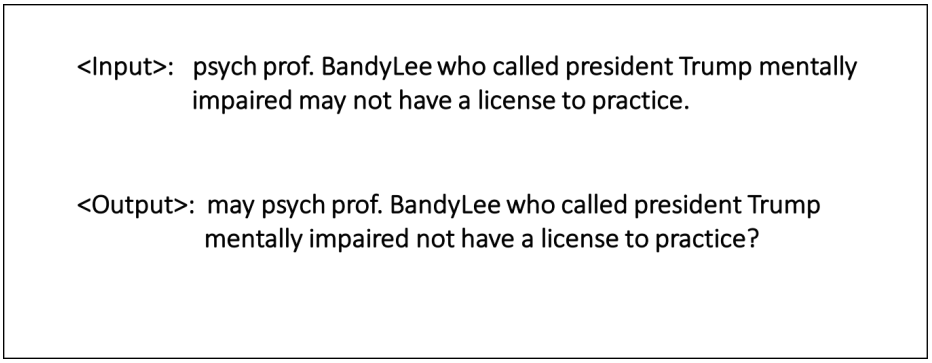

Figure 5.1: An example Query Generation

5.5.2 Google Crawler

There are lots of powerful search engines can help us get tremendous amounts of information, such as Google, Bing, and Yahoo. However, Google is the most powerful search engine with the page ranking algorithm. Thus, we decided to choose Google as a platform to acquire relevant information of claims.

The Google crawler used in this system is built on the previous work proposed by Yang[41]. The method of the crawler is to simulate how people search on Google, however, instead of opening the browser to input the query, the crawler will construct an HTTP request that contains the query and sends it to the Google Search server. Specificity, the crawler simulates the underlying network request. When people input the query and click the search button, the HTTP request will be sent to the server with domain `https://www.google.com` by using the API `https://www.google.com/search?q=`. Then relevant HTML files will be sent back to the browser and converted into simple web pages for people. The crawler will do the same job without the browser. The whole procedure is shown in Figure 5.2. For HTTP request URL construction, the query will be split into multiple tokens and conjugated together with symbol +. Then the new form of a query will be placed after "q=" as shown in Figure 5.3. When the HTML returned, it could be parsed by BeautifulSoup, which is a Python library to extract the useful information from HTML or XML files. Finally, the information we need is going to be stored in files in JSON format.

Figure 5.2: The procedure of Google Crawler

The information extracted from HTML files is a snippet, which is shown in Figure 5.4. The Google snippet is solely applied to the description. It is a "way to provide a concise, human-readable summary of each page's content".[6] There are two reasons why we only crawl the snippets instead of the content of each website returned by Google Search. One is that it is hard to crawl the content of each webpage automatically since different web pages have distinct formats. Designing different crawlers

[6] https://support.google.com/webmasters/answer/35624?hl=en

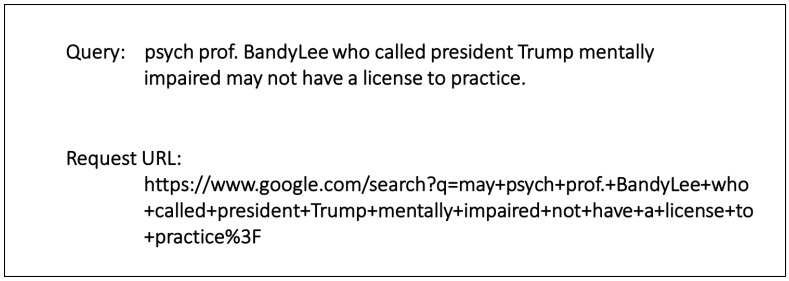

Figure 5.3: An example of HTTP Request

for different formats of web pages is unpractical. The other is that the snippet shown on the Google Search page is the main content of the whole web page. Though the snippet is not intact, it is enough for our system.

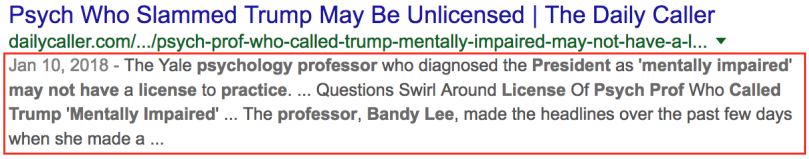

Figure 5.4: An example of Google Snippet

Chapter 6

STANCE CLASSIFIER

Stance Classifier is the task that decides whether each relevant claim supports, disapproves or holds the neutral attitude to the candidate rumor. It is an important part of the system since stances detected by it are going to be an important criterion for users to make their final judgments on candidate rumors. This chapter will describe its workflow.

6.1 Methodology

The input to the stance classifier is a collection of claims about a candidate rumor. A candidate rumor about an event includes a group of claims that are expressed by several tweets, so for a candidate rumor, there is a collection of claims that have spread on Twitter. For example, a candidate rumor about Bandy Lee was that she did not have a license to practice. There were lots of claims talking about this event. All these claims should be collected together to analyze the stances to this candidate rumor. It should be noted that a claim can be determined to be true or false in the end but much time and labor should be cost to look up relevant references manually. During an event, different people may hold different opinions about the same candidate rumor. This difference might be helpful for users to understand this event and an important clue for users to judge whether this candidate rumor is rumor or even its veracity. Thus, instead of showing whether a candidate rumor is a rumor and the final veracity of it, the stances of information about a candidate rumor from different sources will be aggregated and compared. Based on those, users can make a final judgment about the candidate rumor.

As explained above, the core function that needs to be implemented is stance classifier. The goal of stance classifier is to classify the opinion expressed in a text towards a given target. However, targets are not always mentioned in texts, so a more powerful tool should be introduced to solve this problem. Since we planned to build a baseline system at first, we directly used the method proposed by Augenstein et al.[1] The details of it will be introduced in the next section.

6.2 Stance Detection with Bidirectional Conditional Encoding

The main function of this method is to classify the attitude expressed in a text towards a target to be "positive", "negative", or "neutral" when targets are not shown in texts. It leveraged the conditional (Long Short-Term Memory)LSTM encoding to build a representation of the tweet that is dependent on the target. Compared with the method of encoding the tweet and the target separately, its performance was better. As we can see, this method can perfectly satisfy the requirement that our methodology needs.

6.2.1 Recurrent Neural Networks(RNNs) and LSTM Networks

Before introducing the method of Augenstein et al.[1], we will introduce the basic knowledge on which this method built: RNNs and LSTM. RNNs is "a class of artificial neural network where connections between nodes form a directed graph along a sequence."[1] There are networks with a loop in them, allowing information to persist. Also, this is the key feature of the recurrent neural network is different from other traditional neural networks. Since the information can be stored in the memory of RNNs, it can be used as the context to present current task. However, the range of previous information depends. Unfortunately, as the range increases, RNNs become unable to learn to connect the information. To handle this problem, LSTMs is proposed by Hochreiter et al.[11] LSTMs is a special kind of RNNs, and it can lean long-term dependencies. LSTMs has the form of a chain of repeating modules of a

[1] https://en.wikipedia.org/wiki/Recurrent_neural_network

neural network, which is the same as RNNs, but its repeating module has four neural network layers instead of a single one. These four neural network layers assign LSTMs the ability to remove or add information to the cell state. Based on this ability, LSTMs can easily handle the problem of long-term dependencies in RNNs.

6.2.2 Methods

To combine the stance target with the claim in a way that generalizes to unseen targets, this task focus on learning distributed representations and ways to combine them. It leverages the conditional encoding to get target-dependent tweet representations. First, one LSTM is used to encode the target. Then the claim is encoded in another LSTM whose initial state is the representation of the target. Finally, the last output vector is used to predict the stance of the target-tweet pair.

To enrich the context information, we adapted the bidirectional conditional encoding in this work. Two vectors are used to represent the target and the claims respectively, one obtained by reading them from left to right and another obtained from right to left. To achieve this, the initial states of LSTM for the claim are the last state of the forward and reversed encoding of the target.

Since the target in this method is a word or a phrase, the stances of claims to candidate rumors could not be got directly. Thus, we propose a procedure to implement that based on this method. Because the target in the candidate rumor and targets in relevant claims are same, we could use the stance classification method to get the their stances, respectively. If the stance of a claim is as same as the stance of the candidate rumor, then we could conclude that this claim supports the candidate rumor, otherwise disapproves the candidate rumor. By using this procedure, the system could get the stance of relevant claims to the candidate rumors.

Chapter 7

EXPERIMENTS

We design three experiments on our rumor explorer system. First, the performance of the claim finder subsystem will be tested. Second, the performance of the stance classifier is going to be analyzed. Third, the usage of the system will be evaluated.

7.1 Performance of Claim Finder

To evaluate the performance of the claim finder, we first select twenty-two rumors from the Snopes website, then find the general hashtag that represents each rumor on Twitter as the initial query to search the relevant tweets. Besides, we also get claims of events on the Snopes website as the references to be compared with candidate rumors extracted by the claim finder subsystem. After extracting the candidate rumors and getting the top five of them by ranking them based on the popularity, we would look through these results and select events in which the top five candidate rumors contain the similar referenced claims. In the end, twelve events are selected. The performance of the claim finder can be evaluated by comparing the similarities between referenced claims and candidate rumors in these twelve events. The distance between each pair of them could be got by calculating one minus cosine similarity of their embedding vectors, and the embedding vectors are generated from Sent2Vec model. This method is the same one in section 5.4.1.

The results are shown in Table 7.1 and Table 7.2. As shown in these two tables, the column Event represents the hashtag for each event, the column Candidate Rumor represents the claim extracted from relevant tweets of each event, the column Target Claim represents the claim shown on the Snopes website, and the column Distance

represents the distance between them for each event. We can see all distances are around 0.5 and even lower than 0.5, and the average distance is about 0.37. Based on this result, we can conclude that candidate rumors extracted from our Claim Finder subsystem are highly correlated with claims on the Snopes website.

7.2 Performance of Stance Classifier

In this section, we will analyze the performance of the Stance Classifier subsystem in details. In our experiment, Twenty events are further analyzed since meaningful claims are extracted from them. Thus we will analyze three events from these twenty events to show the accuracy of the Stance Classifier on Tweets and Google snippets. The events include ebola-essien, JetLi, and prince-toronto. Since the stance classifier used in this subsystem is supervised learning method, in order to achieve the better performance we decide to divide twenty events into two sets, and each set includes ten events. First, we test the first set of data on the pre-trained model to see its performance and to check what kinds of patterns the pre-trained model could not catch. We then try to label the data in the first set and build a new model based on the new training data and the training data of the pre-trained model. In the end, we test the second set of data on the new model. The event ebola-essien is in the first dataset, and others are in the second dataset.

7.2.1 Experiment on Event "ebola-essien"

For the event "ebola-essien", the extracted candidate rumor is "AC Milan midfielder Michael Essien has been diagnosed with Ebola", and some of relevant tweets and google snippets are shown on Table 7.3 and Table 7.4, respectively. As we can see, three supported tweets are as same as the candidate rumor so that they convey the same information. However, the information conveyed from the candidate rumor was denied by the first opposed tweet and was deemed to be unconfirmed by the second opposed tweet as well. Totally, 66% tweets support the candidate rumor and 33% tweets disapprove the candidate rumor. Thus, most of the relevant information

Table 7.1: Results of Claim Finder I

Event	Candidate Rumor	Target Claim	Distance
BandyLee	Lib Prof BandyLee practicing medicine without license	Dr. Bandy Lee, who has warned the United States that Donald Trump is dangerously impaired, lacks a medical license.	0.56
Gabapentin	new drug for nerve pain gabapentin can enhance opioid high but go undetected in drug screening	Gabapentin is now considered the most dangerous drug in America and will surpass opioids as the largest prescription drug killer.	0.52
Ingraham	Laura Ingraham says immigrant child detention centers are summer camps	Laura Ingraham Compares Child Immigrant Detention Centers To Summer Camps	0.07
JackBreuer	Former White House intern denies flashing white power symbol in photo with Trump JackBreuer	An intern made a white supremacist hand gesture in a photograph with President Trump.	0.39
WhereAreTheChildren	why did Administration lose track of 1,475 children	U.S. Government Lose Track of 1,475 Migrant Children	0.29
Capriccio	Capriccio sangria is giving people HIV	Is Capriccio Sangria Spreading HIV	0.27

Table 7.2: Results of Claim Finder II

Event	Candidate Rumor	Target Claim	Distance
ItsJustAJacket	first lady wears jacket saying to visit children	Did Melania Trump Wear This Jacket on Her Way to Visit Children Separated from Their Families	0.34
JetLi	JetLi medical conditions have forced to quit doing action films	Did Melania Trump Wear This Jacket on Her Way to Visit Children Separated from Their Families	0.59
SouthwestKey	Texas company earned 1.5 billion federal dollars to operate shelters for immigrant children	The Trump administration is paying Southwest Key $458 million to run immigrant child detention centers, and its CEO earns a $1.5 million salary.	0.42
dogjealousy	dogs get jealous babybrother dogjealousy	A study showed that dogs could show jealousy if they caught their owners behaving sweetly toward other dogs	0.52
Irma	Hurricane Irma is Category 6 storm	Hurricane Irma is projected to be so big that it may become a "Category 6" hurricane	0.23
RobertDeNiro	Robert De Niro Was Client Of Prostitution Ring	Robert De Niro linked to a prostitution ring that used children	0.23

Table 7.3: Supported and Opposed Tweets in ebola-essien event

Support	Oppose
AC Milan midfielder Michael Essien has been diagnosed with Ebola	AC Milan have denied reports that midfielder Michael Essien has contracted Ebola while on national duty with Ghana SSFootball
AC Milan midfielder Michael Essien has been diagnosed with Ebola	Unconfirmed reports claim that Michael Essien has contracted Ebola
AC Milan midfielder Michael Essien has contracted Ebola virus	

on Twitter agree with the candidate rumor, but the conclusion is different on Google Search. 37.5% snippets support the candidate rumor and 62.5% snippets disapprove of it. In the "Support" column of the Table 7.4, the first snippet, and last snippet think that Michael Essien had contracted Ebola. Besides, the second one said that Micheal Essien had been diagnosed with Ebola and the third one said that the Ebola on Micheal Essien had been caught in the early stages. All these snippets consent to the candidate rumor, but the fourth one is misclassified. The main content in this snippet is about the fact that Michael Essien was diagnosed with Ebola, but at the beginning of the content, there is a word "Rumor". That means the following content is not reliable. The stance classifier might not catch this key term so that this snippet is misclassified. In the "Oppose" column of the Table 7.4, the first snippet shows that the reports about Micheal Essien slammed Ebola was false, and others show that AC Milan denied that Michael Essien had contracted Ebola.

7.2.2 Experiment on Event "prince-toronto"

In the event "prince-toronto", the extracted candidate rumor is "Prince will be performing at Massey Hall Tonight". Some of relevant tweets are presented in Table7.5. As we can see Prince would play a surprise show and would play for 2 hours from first and second tweets in "Support" column. Besides, based on fourth and fifth

Table 7.4: Supported and Opposed Google Snippets in ebola-essien event

Support	Oppose
Oct 12, 2014 ... AC Milan midfielder Michael Essien has reportedly contracted the deadly ... that treated Mr. Duncan is exhibiting signs of Ebola virus – she was...	Oct 13, 2014 ... AC Milan, Ghana midfielder Michael Essien slams Ebola reports as "false" ... Michael Essien and his club AC Milan have both categorically denied ... Such reports, totally unfounded, have also never been confirmed by any ...
12 Oct 2014 ... AC Milan midfielder Michael Essien has been diagnosed with Ebola. Get well soon Michael. Daily Times Transfer Related @TransferRelated ...	Oct 13, 2014 ... Serie A club AC Milan have "categorically denied" reports claiming that ... media reports that midfielder Michael Essien was being treated after ...
Oct 12, 2014 ... ... Who Treated Late Patrick Sawyer Contracts Ebola / Did Michael Jackson Just Die ... Ghanaian football star and AC Milan striker, Michael Essien, has ... 'He is a very strong person and the Ebola has been caught in the early stages. ... The AC Milan midfielder, who joined the Italian giants in January this ...	Oct 13, 2014 ... AC Milan have "categorically denied" that Ghanaian midfielder Michael ... claimed Essien was being treated after contracting the deadly virus.
Michael Essien: Legal Action Against Ebola Rumor ... that he is diagnosed with the deadly Ebola virus Ghana's midfielder,. ... Michael Essien, the Ghanaian midfielder has said that he's going to the FIFA World Cup 2014 campaign is to die for his country. The AC Milan star joined the camp on Monday and he participate.	Oct 13, 2014 ... Milan AC Milan have categorically denied that Ghanaian midfielder Michael Essien has contracted Ebola while on ... Reports in Ghana over the weekend claimed Essien was being treated after contracting the deadly virus. ... Michael Essien and Carlton Cole are facing problems in Indonesia after their ...
ct 13, 2014 ... Thanks to a pair of local reports, AC Milan had to issue a statement ... Ghanaian midfielder Michael Essien has contracted Ebola while on national team ... claimed Essien was being treated after contracting the deadly virus.	

Table 7.5: Supported and Opposed Tweets in prince-toronto event

Support	Oppose
Prince Playing Surprise Show in Toronto Tonight	Massey Hall has confirmed Prince will not be playing surprise show tonight
Prince played for 2 hours in backyard Toronto	Prince will not be performing at tonight
Prince playing Massey Hall tonight	Prince will not be performing at Massey Hall tonight
Event promoter says surprise Prince show at Massey Hall Tuesday	LiveNation confirms Prince will not be playing Massey Hall tonight
Live Nation says PrinceTO cbcto Prince playing Massey Hall	Prince won't be performing tomorrow night either

tweets in this column, event promoter and Live Nation both said that Price would play at Massey Hall. On the other hand, in the "Oppose" column the first and forth tweets show that Massey Hall and Live Nation confirmed Price would not be playing. Moreover, others tweets in this column presented the same information as well. 53.7% tweets support the candidate rumor, that is, they think that Prince would play at Massey Hall. However, 66.6% of google snippets disapprove of the candidate rumor, and they think that the Price show would be a rumor. In the "Oppose" column of Table 7.6, the first and second snippets both think that the Prince show would be a rumor. Based on the results of the previous experiment, we tried to label some data that includes "rumor" terms to let stance classifier model recognize this pattern. Thus, these two snippets could be classified correctly. Besides, sources in the third snippet had claimed the Price was a no-show. Moreover, the last snippet said that the Live Nation confirmed that there would be no Prince show.

7.2.3 Experiment on Event "JetLi"

The third event is about "JetLi", and its extracted candidate rumor is "JetLi medical conditions have forced to quit doing action films". In Table 7.7, we present some of the tweets that are related to the candidate rumor. In the "Support" column,

Table 7.6: Supported and Opposed Google Snippets in prince-toronto event

Support	Oppose
Nov 4, 2014 ... It looks like your purple dreams may come true: rumours are flying this morning about a secret Prince show at Massey Hall going on tonight.	Can you keep a secret? ... Rumours of a line forming outside of #Toronto's masseyhall for two secret $10 Prince shows tonight. #AlwaysON.
Nov 4, 2014 ... Fans and music business types started lining up early Tuesday morning outside Toronto's Massey Hall for a concert tonight by Prince	"May 19, 2015 ... CTV Toronto: Two surprise Prince shows in T.O ... the cold when it was rumoured the ""1999"" singer would be playing a secret show in the city."
	Nov 2, 2016 ... Sources have claimed the Prince was a "no-show" for a 11.30am flight ... flight from Heathrow to Toronto yesterday when the media broke the ...
	Nov 4, 2014 ... ... anticipation of a secret concert, before promoter Live Nation confirmed there would be no Prince show on Tuesday and apologized to fans. ... Social media had picked up on the rumour early Tuesday morning after the band...

Table 7.7: Supported and Opposed Tweets in JetLi event

Support	Oppose
viral photo has Jet Li fans worried about health	Manager Says Hyperthyroidism Is Nothing Life-Threatening
new photo leaves fans worried about health GenevieveBlog jetli jetlishealth	Illness Is Nothing Manager Assures Fans
Jet Li looking like being turned into Poddling slave in Dark Crystal	Illness Is Nothing Manager Assures Fans
Jet Li is suffering from and spinal problems action star known for physical roles in films	Manager Says Hyperthyroidism Is Nothing Life-Threatening
Shocking Photo Ignites Health Concerns	Manager Says Hyperthyroidism Is Nothing Life-Threatening

we could know that photos about Jet Li health spread online and left fans worried about him based on the first, the second, and the last tweets. Moreover, Jet Li was looking like a slave in the photo in the third tweet. The fourth tweet also shows that Jet Li was suffering from spinal problems. In the "Oppose" column, tweets hold different opinions. Three same tweets show that Hyperthyroidism of Jet Li was nothing life-threatening and other two same tweets indicate that manager assured fans that illness was nothing. Most of the tweets are classified correctly into their corresponding groups. 78.4% of tweets agree with the candidate rumor, which means most of the users on Twitter worried about Jet Li's health condition and thought Jet Li was sick. The claims of news from News API are shown in Table 7.8. As we can see in the "Support" column, Jet Li still suffered from injuries to his legs and spine and had a battle with hyperthyroidism. Moreover, his spinal conditions had largely forced him to retire from acting. However, information showed in "Oppose" column are different. Jet Li was doing great and feeling great and had recovered from hyperthyroidism. Besides, the manager responded that Jet Li was fine. 64.3% of snippets disapprove of the candidate rumor and think that Jet Li's condition was good.

Table 7.8: Supported and Opposed Google Snippets in JetLi event

Support	Oppose
May 21, 2018 - He had also suffered severe injuries to his legs and spine while ... Related: Disney's Live-Action Mulan Movie Casts Jet Li & Gong Li ... in the upcoming live- action Mulan film, using an earlier video from this year ... he credits his religion with helping him through his health issues. ... Email Leave A Comment ...	May 21, 2018 - A viral photo has Jet Li fans worried about his health. 'Doing great and feeling great! ... spanned decades of action movies, Li also known as Li Lianjie has in recent years battled hyperthyroidism, a condition that ... He's all well and good , Chasman added, saying he had just spoken with Li's assistant.
May 19, 2018 - Jet Li fans share personal battles with illness that has stricken martial ... Jet Li fans share their personal battles with hyperthyroidism, as martial arts icon's health problems continue to shock ... Li had aged far too much having lived a tough life as an action movie star. ... Love what's money got to do with it?	Jul 27, 2016 - In the movie League Of Gods, Jet Li plays the role of the wise strategist ... Li, 53, says he has recovered from hyperthyroidism, a condition he was ... Of Fury (2013) and American and Hollywood action blockbuster The Expendables 3 (2014). ... Jane made the news recently after appearing at a fundraiser, ...
May 23, 2018 - Do you have the same illness haunting Jet Li? ... Lights, camera, action. ... And he had to quickly come back from that setback because the movie studio was ... From his days as a performing martial artist in his teens, ... battles with hyperthyroidism, as martial arts icon's health problems continue to shock.	Dec 27, 2013 - The action-movie actor says he kept his 2010 diagnosis under control with medication, but the condition recently came back with a vengeance. ... In Tuesday's taping, the 50-year-old Li appeared to have a fuller face and ...
May 21, 2018 - Internationally renowned martial arts action hero Jet Li, who now has several health ailments, ... Li's elderly appearance in that film wasn't just the magic of makeup. ... and spinal conditions, which have largely forced him to retire from acting. ..... Meghan Markle Is 'Doing Amazing' in New Chapter with Prince ...	May 23, 2018 - A picture of Li looking old and frail had been doing the rounds on ... Jet Li's manager has responded to concern for his health online by saying that the action movie star is in ... performer - have left him with leg and spine problems that have led ... We all agree, but if he says he's fine, let's just leave him alone.

7.2.4 Conclusion

Based on the analysis of these three events, we could see that the performance of the stance classifier is good in most cases. Since the stance classifier is built on a supervised learning algorithm, the training data is a key factor of it. Labeling stances of training data are much easier than labeling veracities of data. Thus, we could conclude that the stance classifier subsystem would be a useful part of our system.

7.3 Usage of System

In this section, we will introduce how to use our system to infer the same conclusion shown on the Snopes website. To do that, we will take a #BandyLee event as an example and show the workflow step by step in Figure 7.1, 7.3, Figure 7.4, and Figure 7.2. At first step, we will select the claim in topic0 under #BandyLee event. At the second step, on Tweets page, we can see the promoted tweets and opposed tweets from left to right. Lots of supported tweets said that Bandy Lee might not have a license. At the third step, on the Google page, we can see the contents of web pages hold different stances returned from Google Search. At the fourth step, two percentage charts are presented on the Chart page. The left one shows that lots of tweets support the claim but the left one shows that most of the web pages disapprove the claim. Besides, the percentage of opposing in Google is higher than the percentage of support on Twitter. Since we put more weight on the contents of web pages returned by Google Search, we could conclude that this claim might not be valid. This conclusion is as same as a result shown on Snopes, Some far-right websites have posted misleading stories casting doubt on whether Dr. Bandy Lee holds a current medical license.[1] By presenting this example, we can see user could infer to the same conclusion shown on Snopes website based on the information automatically generated by our system.

[1] https://www.snopes.com/fact-check/does-psychiatrist-trump-lack-license/

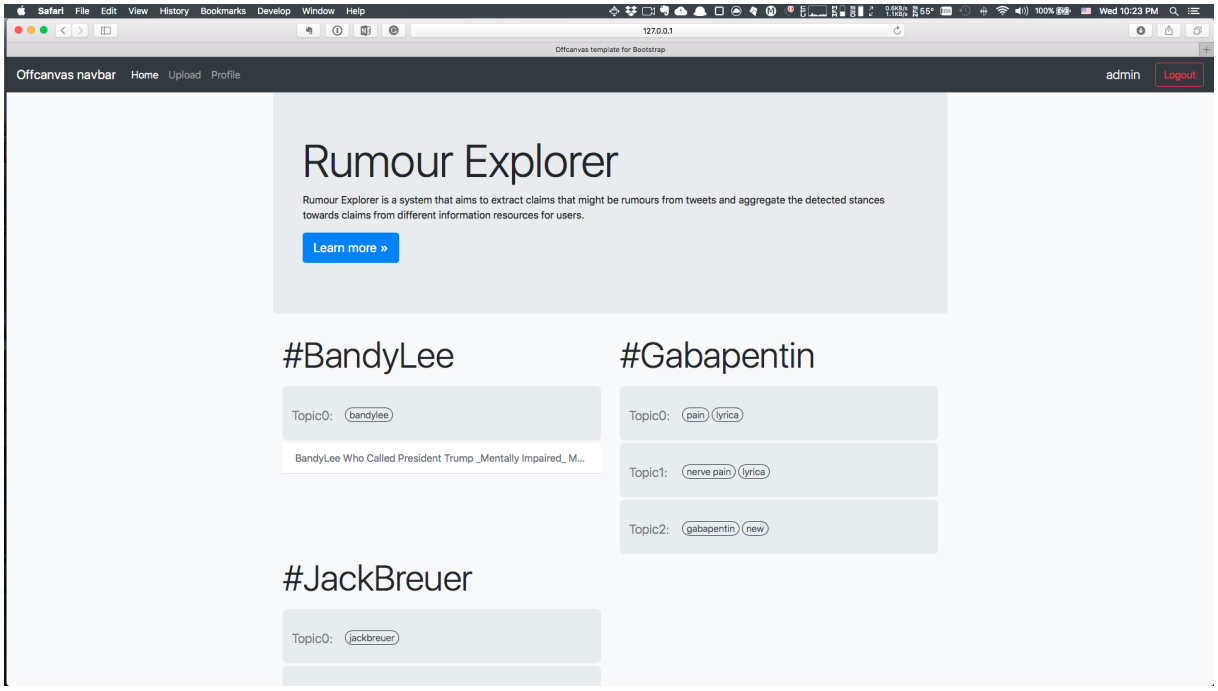

Figure 7.1: Hashtags of All Events

7.4 Evaluations of Thirty One Events

After introducing the usage of the whole system, we will test all data on our system and see the results. There are thirty-one events, and our system correctly extracts candidate rumors of twenty events. Figure 7.5 and Figure 7.5 present the results of these twenty events. In these two figures, we present candidate rumor topics, the extracted candidate rumors that are most related to the referenced claims, their ranks in all extracted candidate rumors, the ground truth of the candidate rumors, and the percentage of different stances on candidate rumors from Twitter and Google snippets. As we can see, for some events, tweets and google snippets hold the same stance on the candidate rumor, such as event #SouthwestKey, #WhereAreTheChildren, and #JackBreuer. However, they hold different stances on candidate rumors in most cases. Thus, we could conclude that different information sources may hold different stances on the same candidate rumor. For the veracities of stances, stances of tweets are correct nine times and stances of google snippets are correct 15 times. Based on these results, we

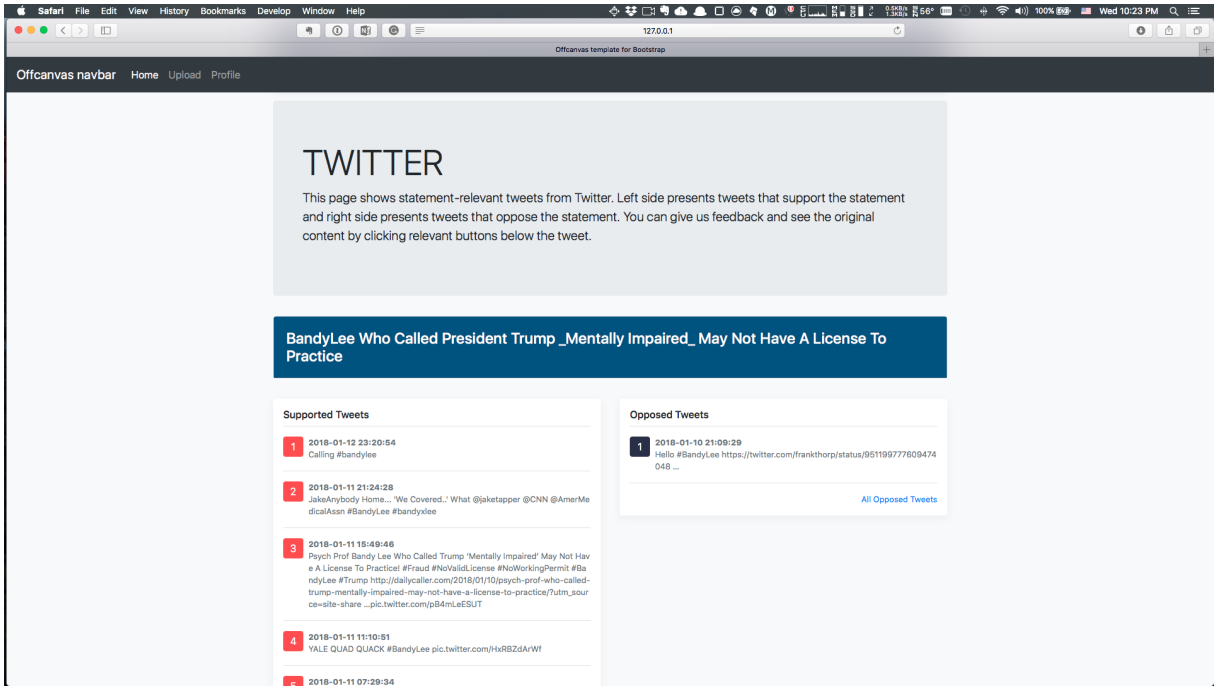

Figure 7.2: Tweets Related to the Candidate Rumor

could conclude that information from Google search is more reliable than that from Twitter. Since most of the information on Google is from authoritative websites, the results we get from our systems are reasonable. In the end, the results of experiments demonstrate the probability of using stances of different information on claims to detect candidate rumors.

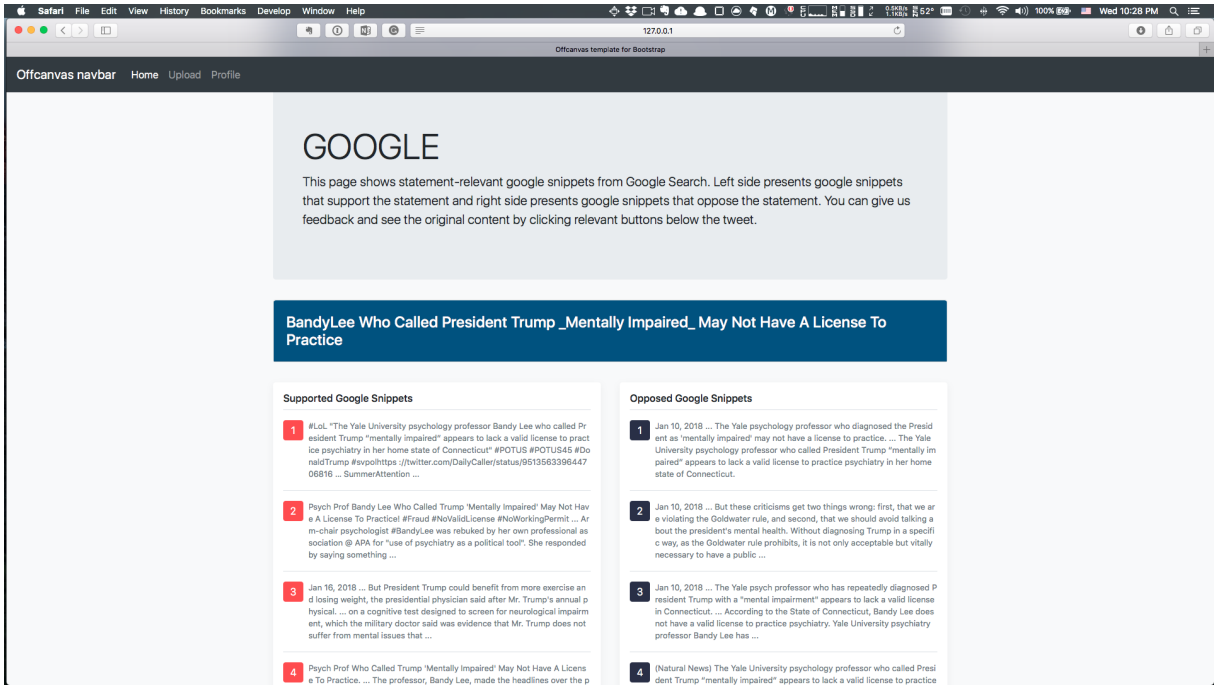

Figure 7.3: News from Google Snippets Related to the Candidate Rumor

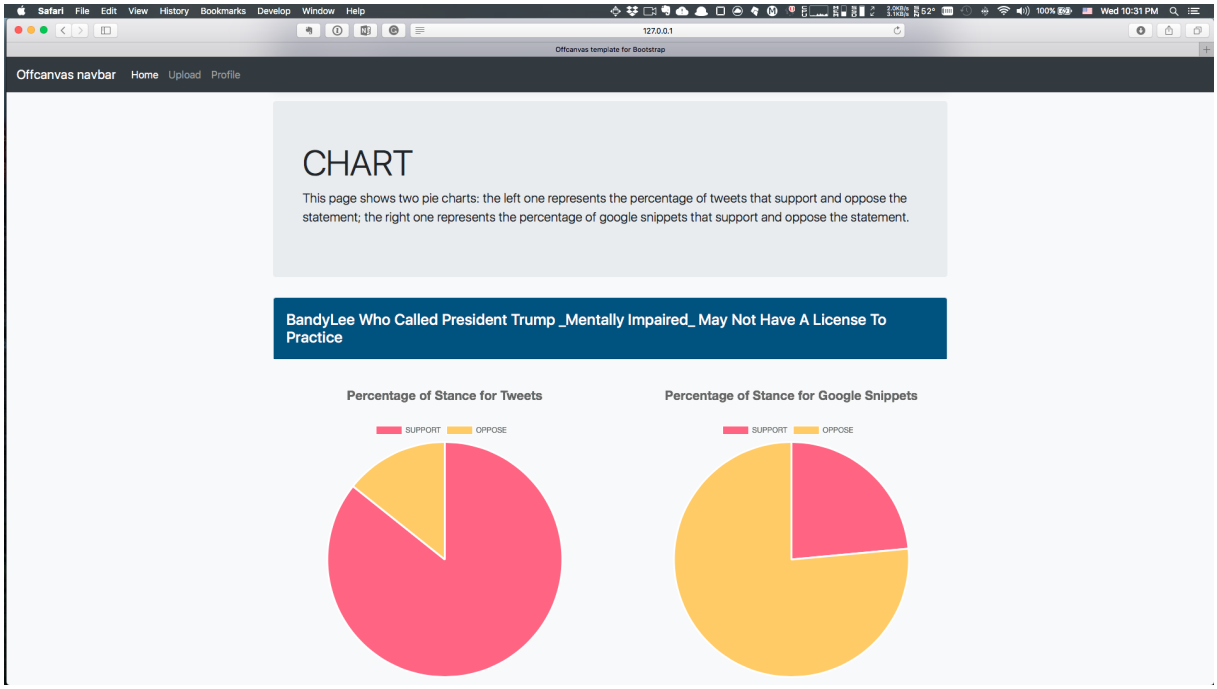

Figure 7.4: Stances on Claims from Tweets and Google Snippets

Candidate Rumor ID	Candidate Rumor Source	Candidate Rumor	Rank	Ground Truth	Twitter Result on Candidate Rumor	Snippet Stances on Candidate Rumor
1	#BandyLee	Lib Prof BandyLee practicing medicine without license	1st	False	75% supports 25% disapproves	45% supports 55% disapproves
2	#Capriccio	Capriccio sangria is giving people HIV	1st	False	50% supports 50% disapproves	10% supports 90% disapproves
3	#Gabapentin	Gabapentin may be new non-opioid drug of abuse	1st	False	100% supports 0% disapproves	20% supports 80% disapproves
4	#Ingraham	Laura Ingraham says immigrant child detention centers are summer camps	2nd	True	95% supports 5% disapproves	50% supports 50% disapproves
5	#ItsJustAJacket	first lady wears jacket saying to visit children	1st	True	100% supports 0% disapproves	70% supports 40% disapproves
6	#JackBreuer	Former White House intern denies flashing white power symbol in photo with Trump JackBreuer	1st	Neutral	85.7% supports 14.3% disapproves	70% supports 30% disapproves
7	#SouthwestKey	Texas company earned 1.5 billion federal dollars to operate shelters for immigrant children	1st	True	94.7% supports 5.3% disapproves	90% supports 10% disapproves
8	#WhereAreTheChildren	why did Administration lose track of 1,475 children	2nd	True	100% supports 0% disapproves	60% supports 40% disapproves
9	#sydneysiege	Sydney terrorists have nothing to do with Islam	2nd	True	100% supports 0% disapproves	35.3% supports 64.7% disapproves
10	#ebola-essien	AC Milan midfielder Michael Essien has been diagnosed with Ebola	1st	False	60% supports 40% disapproves	37.5% supports 62.5% disapprovr

Figure 7.5: Evaluations of Events 1-10

Candidate Rumor ID	Candidate Rumor Source	Candidate Rumor	Rank	Ground Truth	Twitter Stances on Candidate Rumor	Snippet Stances on Candidate Rumor
11	#dogjealousy	A study showed that dogs could show jealousy if they caught their owners behaving sweetly toward other dogs	1st	True	100% supports 0% disapproves	84.6% supports 15.4% disapproves
12	#Irma	Hurricane Irma is projected to be so big that it may become a "Category 6" hurricane; that a new "Category 6" will be invented specifically for Hurricane Irma	1st	False	22.2% supports 77.8% disapproves	36.8% supports 63.2% disapproves
13	#TrumpSalary	In May 2018 President Trump donated one-fourth of his $400,000-per-year salary to the Department of Veterans Affairs.	1st	True	100% supports 0% disapproves	95% supports 5% disapproves
14	#charliehebdo	Eiffel Tower went dark Thursday evening to honor victims of CharlieHebdo attack	3rd	False	100% supports 0% disapproves	89.5% supports 10.5% disapproves
15	#JetLi	JetLi medical conditions have forced to quit doing action films	1st	Neutral	78.4% supports 21.6% disapproves	35.7% supports 64.3% disapproves
16	#ferguson	Unarmed teenager MikeBrown was shot ten times by Ferguson police officer for accusation of shoplifting	1st	False	82.6% supports 17.4% disapproves	14.3% supports 85.7% disapproves
17	#germanwings-crash	Co-Pilot Was MUSLIM CONVERT Hero of Islamic State	3rd	False	100% supports 0% disapproves	15.4% supports 84.6% disapproves
18	#gurlitt	Swiss art museum accepts artworks bequeathed by late art dealer Gurlitt	3rd	False	69.7% supports 39.3% disapproves	50% supports 50% disapproves
19	#prince-toronto	Prince will be performing at Massey Hall Tonight	2nd	False	53.7% supports 46.3% disapproves	33.3% supports 66.7% disapproves
20	#putinmissing	Vladimir disappearance could mean undergoing coup	1st	False	100% supports 0% disapproves	25% supports 75% disapproves

Figure 7.6: Evaluations of Events 11-20

Chapter 8

CONCLUSION

This thesis introduces a system for rumor exploration. It consists of three essential subsystems: Tweets Crawler, Candidate Rumors Finder, and Stance Classifier.

To detect popular events and retrieve relevant tweets on Twitter, we leverage the hashtags on Twitter as queries. Moreover, we expand the query with relevant hashtags of an initial hashtag and then simplify the query by filtering out general hashtags based on the popularity. The measurement of popularity we proposed could effectively remove the general hashtags from all relevant hashtags. Combined with the powerful tool GetOldTweets-python, the generated queries could help us acquire relevant tweets.

Besides, we also implement the subsystem Candidate Rumors Finder to recognize the candidate rumors embedded in tweets. In the Candidate Rumors Finder, essential and popular claims could be precisely extracted from tweets, and then they could be effectively clustered into different groups corresponding to their similarity. In each group, a representative claim would be selected as the candidate rumor. Finally, the relevant information can also be acquired from both Google and News API based on queries generated from candidate rumors. After evaluation, the candidate rumors extracted from this subsystem are very similar to that on the Snopes website, and the average distance is 0.37.

Moreover, by integrating the stance classification function in our system, the stances of related claims and authorized information to candidate rumors could be detected. Then based on aggregated stances information from different sources, the users could conclusively infer the same conclusion for candidate rumors shown on Snopes website in the most cases.

In summary, our system could effectively retrieve relevant tweets based on the general hashtags of events, then extract the useful claims as candidate rumors from tweets of twenty out of thirty-one events, and based on them to search more related external information from Google search. Moreover, stances of different information could be correctly classified and then used as the references for users to detect the rumors. In the end, according to the results of experiments, we could demonstrate that stances of different information could be used to detect rumors and our system is reliable.

For the future work, we plan to develop a new rumor detection approach based on the work proposed by Zubiaga et al[47]. Zubiaga et al. suggested a novel approach that leveraged the context information to detect rumors. While it exploited the context information, the performance of approach in early time is not satisfied enough due to the absence of prior information. To compensate for this deficiency, we will introduce as prior information the past posts and probability that the account is a bot. The new rumor detection approach could be an auxiliary function in our system to provide more information for users to make final judgments.

BIBLIOGRAPHY

[1] Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. Stance detection with bidirectional conditional encoding. *CoRR*, abs/1606.05464, 2016.

[2] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 675–684, New York, NY, USA, 2011. ACM.

[3] Cheng Chang, Yihong Zhang, Claudia Szabo, and Quan Z. Sheng. Extreme user and political rumor detection on twitter. In Jinyan Li, Xue Li, Shuliang Wang, Jianxin Li, and Quan Z. Sheng, editors, *Advanced Data Mining and Applications*, pages 751–763, Cham, 2016. Springer International Publishing.

[4] Teh-Chuan Chen and Kuo-Liang Chung. An efficient randomized algorithm for detecting circles. *Computer Vision and Image Understanding*, 83(2):172 – 191, 2001.

[5] Alton Y. K. Chua, Cheng-Ying Tee, Augustine Pang, and Ee-Peng Lim. The retransmission of rumor-related tweets: Characteristics of source and message. In *Proceedings of the 7th 2016 International Conference on Social Media & Society*, SMSociety '16, pages 22:1–22:10, New York, NY, USA, 2016. ACM.

[6] David Crystal et al. *Internet linguistics: A student guide*. Routledge, 2011.

[7] Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. Semeval-2017 task 8: Rumoureval: Determining rumour veracity and support for rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 69–76. Association for Computational Linguistics, 2017.

[8] Georgios Giasemidis, Colin Singleton, Ioannis Agrafiotis, Jason R. C. Nurse, Alan Pilgrim, Chris Willis, and Danica Vukadinovic Greetham. Determining the veracity of rumours on twitter. *CoRR*, abs/1611.06314, 2016.

[9] Sardar Hamidian and Mona Diab. Rumor detection and classification for twitter data. In *SOTICS 2015 : The Fifth International Conference on Social Media Technologies, Communication, and Informatics*, 2015.

[10] Sardar Hamidian and Mona T Diab. Rumor identification and belief investigation on twitter. In *Proceedings of NAACL-HLT 2016*, 2016.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[12] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[13] Elena Kochkina, Maria Liakata, and Isabelle Augenstein. Turing at semeval-2017 task 8: Sequential approach to rumour stance classification with branch-lstm. *CoRR*, abs/1704.07221, 2017.

[14] Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. PHEME dataset for Rumour Detection and Veracity Classification. 6 2018.

[15] Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. A dependency parser for tweets. In *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP*, pages 1001–1012, 2014.

[16] Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. Rumor detection over varying time windows. *PLOS ONE*, 12(1):1–19, 01 2017.

[17] Jey Han Lau, Nigel Collier, and Timothy Baldwin. On-line trend analysis with topic models: #twitter trends detection topic model online. In *COLING*, 2012.

[18] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 1867–1870, New York, NY, USA, 2015. ACM.

[19] Kuang Lu, Hui Fang, and Diego Roa. Concept based tie-breaking and maximal marginal relevance retrieval in microblog retrieval. In *TREC*, 2014.

[20] Michal Lukasik, P. K. Srijith, Duy Vu, Kalina Bontcheva, Arkaitz Zubiaga, and Trevor Cohn. Hawkes processes for continuous time sequence classification: an application to rumour stance classification in twitter. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 393–398. Association for Computational Linguistics, 2016.

[21] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai 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*, CIKM '15, pages 1751–1754, New York, NY, USA, 2015. ACM.

[22] Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. Sentiment classification using word sub-sequences and dependency sub-trees. In *PAKDD*, volume 5, pages 301–311. Springer, 2005.

[23] Richard McCreadie, Craig Macdonald, and Iadh Ounis. Crowdsourced rumour identification during emergencies. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 965–970, New York, NY, USA, 2015. ACM.

[24] Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794. Association for Computational Linguistics, 2010.

[25] Onook Oh, Manish Agrawal, and H. Raghav Rao. Community intelligence and social media services: A rumor theoretic analysis of tweets during social crises. *MIS Q.*, 37(2):407–426, June 2013.

[26] Olutobi Owoputi, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *In Proceedings of NAACL*, 2013.

[27] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. *CoRR*, abs/1703.02507, 2017.

[28] Swit Phuvipadawat and Tsuyoshi Murata. Breaking news detection and tracking in twitter. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, WI-IAT '10, pages 120–123, Washington, DC, USA, 2010. IEEE Computer Society.

[29] Vahed Qazvinian, Emily Rosengren, Dragomir R. Radev, and Qiaozhu Mei. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1589–1599, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[30] Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. Twitterstand: News in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 42–51, New York, NY, USA, 2009. ACM.

[31] Philip J Stone, Dexter C Dunphy, and Marshall S Smith. The general inquirer: A computer approach to content analysis. 1966.

[32] Laura Tolosi, Andrey Tagarev, and Georgi Georgiev. An analysis of event-agnostic features for rumour classification in twitter, 2016.

[33] Laura Tolosi, Andrey Tagarev, and Georgi Georgiev. An analysis of event-agnostic features for rumour classification in twitter. In *Social Media in the Newsroom, Papers from the 2016 ICWSM Workshop, Cologne, Germany, May 17, 2016*, 2016.

[34] Elizabeth Closs Traugott. On the rise of epistemic meanings in english: An example of subjectification in semantic change. *Language*, pages 31–55, 1989.

[35] Soroush Vosoughi. *Automatic detection and verification of rumors on Twitter.* PhD thesis, Massachusetts Institute of Technology, 2015.

[36] Xinyue Wang, Laurissa Tokarchuk, Felix Cuadrado, and Stefan Poslad. *Adaptive Identification of Hashtags for Real-Time Event Data Collection*, pages 1–22. Springer International Publishing, Cham, 2015.

[37] Yue Wang, Hao Wu, and Hui Fang. An exploration of tie-breaking for microblog retrieval. In Maarten de Rijke, Tom Kenter, Arjen P. de Vries, ChengXiang Zhai, Franciska de Jong, Kira Radinsky, and Katja Hofmann, editors, *Advances in Information Retrieval*, pages 713–719, Cham, 2014. Springer International Publishing.

[38] 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, April 2015.

[39] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, MDS '12, pages 13:1–13:7, New York, NY, USA, 2012. ACM.

[40] Peilin Yang, Srikanth Thiagarajan, and Jimmy Lin. Robust, scalable, real-time event time series aggregation at twitter. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 595–599, New York, NY, USA, 2018. ACM.

[41] Peilin Yang, Hongning Wang, Hui Fang, and Deng Cai. Opinions matter: A general approach to user profile modeling for contextual suggestion. *Inf. Retr.*, 18(6):586–610, December 2015.

[42] Renxian Zhang, Dehong Gao, and Wenjie Li. What are tweeters doing: Recognizing speech acts in twitter. *Analyzing Microtext*, 11:05, 2011.

[43] Renxian Zhang, Dehong Gao, and Wenjie Li. Towards scalable speech act recognition in twitter: tackling insufficient training data. In *Proceedings of the Workshop on Semantic Analysis in Social Media*, pages 18–27. Association for Computational Linguistics, 2012.

[44] Zili Zhang, Julie Zhang, and Hengyun Li. Predictors of the authenticity of internet health rumours. 32:195–205, 09 2015.

[45] Zhe Zhao, Paul Resnick, and Qiaozhu Mei. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 1395–1405, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.

[46] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. Detection and resolution of rumours in social media: A survey. *CoRR*, abs/1704.00656, 2017.

[47] Arkaitz Zubiaga, Maria Liakata, and Rob Procter. Learning reporting dynamics during breaking news for rumour detection in social media. *CoRR*, abs/1610.07363, 2016.