Automatic Detection and Verification of Rumors on Twitter

by

Soroush Vosoughi

S.B., Massachusetts Institute of Technology (2008)
M.S., Massachusetts Institute of Technology (2010)

Submitted to the Program in Media Arts and Sciences,
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2015

Author ______________________________

Program in Media Arts and Sciences,
May 7, 2015

Certified by ______________________________

Deb K. Roy
Associate Professor
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by ______________________________

Pattie Maes
Academic Head
Program in Media Arts and Sciences

Automatic Detection and Verification of Rumors on Twitter

by

Soroush Vosoughi

Submitted to the Program in Media Arts and Sciences,
on May 7, 2015, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

The spread of malicious or accidental misinformation in social media, especially in time-sensitive situations such as real-world emergencies can have harmful effects on individuals and society. This thesis develops models for automated detection and verification of rumors (unverified information) that propagate through Twitter. Detection of rumors about an event is achieved through classifying and clustering assertions made about that event. Assertions are classified through a speech-act classifier for Twitter developed for this thesis. The classifier utilizes a combination of semantic and syntactic features to identify assertions with 91% accuracy. To predict the veracity of rumors, we identify salient features of rumors by examining three aspects of information spread: linguistic style used to express rumors, characteristics of people involved in propagating information, and network propagation dynamics. The predicted veracity of a time series of these features extracted from a rumor (a collection of tweets) is generated using Hidden Markov Models. The verification algorithm was tested on 209 rumors representing 938,806 tweets collected from real-world events including the 2013 Boston Marathon bombings, the 2014 Ferguson unrest and the 2014 Ebola epidemic, and many other rumors reported on popular websites that document public rumors. The algorithm is able to predict the veracity of rumors with an accuracy of 75%. The ability to track rumors and predict their outcomes may have practical applications for news consumers, financial markets, journalists, and emergency services, and more generally to help minimize the impact of false information on Twitter.

Thesis Supervisor: Deb K. Roy
Title: Associate Professor, Program in Media Arts and Sciences

Automatic Detection and Verification of Rumors on Twitter

by

Soroush Vosoughi

The following people served as readers for this thesis:

Acknowledgments

This thesis is dedicated to the memory of Officer Sean A. Collier of the MIT Police, who gave his life defending the MIT community.

Soroush Vosoughi

I have never been good with words, which is why I find myself in such a delicate conundrum, to give everyone the thanks they deserve for helping me with such a Sisyphean undertaking, the completion of a PhD at MIT.

First and foremost, I would like to thank my advisor, Prof. Deb Roy. I have been doing research in Deb's group since spring of 2005, as a freshman at MIT. Throughout the years, Deb has provided opportunities for me to work on a vast array of interesting projects; from robots to models of language acquisition, and most recently social networks and complex systems.

I also wish to express my gratitude to my committee, Prof. Sinan Aral and Dr. Allen Gorin, for their support, guidance and feedback throughout this whole process.

Next, I would like to give very special thanks to Kai-yuh Hsiao, who as a Ph.D. student in Deb's group hired me as a young and inexperienced freshman to work on a very awesome robotics platform called *Trisk*. Kai-yuh has been a great mentor and friend throughout my more than ten years at MIT and has been a continuing source of inspiration to me. Without Kai-yuh taking a chance with me those many years ago, I would most likely not be where I am today.

Through out my years at MIT, I have met many great friends and worked with amazing colleagues. I would like to thank all the members of the Cognitive Machines group and the Laboratory for Social Machines for making my years at MIT, without a doubt, the best years of my life. Here is a list of my colleagues in random order: Brandon Roy, Jeff Orkin, Philip DeCamp, Karina Lundahl, Hilke Reckman,Rony Kubat, Matt Miller, George Shaw, Joe Wood, Tynan Smith, Kaname Tomite, Stefanie Tellex, Aithne Sheng-Ying Pao, Kleovoulos ”Leo” Tsourides, Sophia Yuditskaya, Thananat Jitapunkul, Mutsumi Sullivan, Yuko Barnaby, Angela Brewster, Michael Fleischman, Tanya Schalchlin, Jethran Guinness, Kai-yuh Hsiao, Philipp Robbel, Nick Mavridis, Peter Gorniak, David Wang, Peter Lao, Preeta Bansal, Sophie Chou, James Kondo, Perng-hwa ”Pau” Kung, Prashanth Vijayaraghavan, Neo Mohsenvand, Heather Pierce, William Powers, Martin Saveski, Russell Stevens and Ivan Sysoev.

Very special thanks go to Dr. Brandon Roy from the Cognitive Machine group and Mostafa ”Neo” Mohsenvand from the Laboratory for Social Machines. Brandon has been a great friend and colleague whose friendship and advise I value very much. Neo, who in addition to having a great technical mind is a great expert in visualization, helped with most of the visualizations in my thesis defense, allowing my work to shine.

My greatest love goes to my girlfriend Julia who is working on her Ph.D. at the *whitehead Institute for Biomedical Research* at MIT. She has been a source of support, happiness and love in the last year since we met.

Last, but certainly not least, I would like to thank my dad, mom, brother and sister, whose continued support and encouragement saw me through more than ten years at MIT.

Soroush Vosoughi

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The reports of my death have been greatly exaggerated.

Mark Twain

In the last decade the Internet has become a major player as a source for news. A study by the Pew Research Center has identified the Internet as the most important resource for the news for people under the age of 30 in the US and the second most important overall after television [11]. More recently, the emergence and rise in popularity of social media and networking services such as Twitter, Facebook and Reddit have greatly affected the news reporting and journalism landscapes. While social media is mostly used for everyday chatter, it is also used to share news and other important information [38, 66]. Now more than ever, people turn to social media as their source of news [54, 96, 52]; this is especially true for breaking-news situations, where people crave rapid updates on developing events in real time. As Kwak et al. (2010) have shown, over 85% of all *trending topics*[1] on Twitter are news [52]. Moreover, the ubiquity, accessibility, speed and ease-of-use of social media have made them invaluable sources of first-hand information. Twitter for example has proven to be very useful in emergency and disaster situations, particularly for response and recovery [100]. However, the same factors that make social media a great resource

[1]Trending topics are those topics being discussed more than others on Twitter.

for dissemination of breaking-news, combined with the relative lack of oversight of such services, make social media fertile ground for the creation and spread of unsubstantiated and unverified information about events happening in the world.

This unprecedented shift from traditional news media, where there is a clear distinction between journalists and news consumers, to social media, where news is crowd-sourced and anyone can be a reporter, has presented many challenges for various sectors of society, such as journalists, emergency services and news consumers. Journalists now have to compete with millions of people online for breaking-news. Often time this leads journalists to fail to strike a balance between the need to be first and the need to be correct, resulting in an increasing number of traditional news sources reporting unsubstantiated information in the rush to be first [12, 13]. Emergency services have to deal with the consequences and the fallout of rumors and witch-hunts on social media, and finally, news consumers have the incredibly hard task of sifting through posts in order to separate substantiated and trustworthy posts from rumors and unjustified assumptions. A case in point of this phenomenon is the social media's response to the Boston Marathon bombings. As the events of the bombings unfolded, people turned to social media services like Twitter and Reddit to learn about the situation on the ground as it was happening. Many people tuned into police scanners and posted transcripts of police conversations on these sites. As much as this was a great resource for the people living in the greater Boston, enabling them to stay up-to-date on the situation as it was unfolding, it led to several unfortunate instances of false rumors being spread, and innocent people being implicated in witch-hunts [51, 56, 99]. Another example of such phenomenon is the 2010 earthquake in Chile where rumors propagated in social media created chaos and confusion amongst the news consumers [64].

What if there was a tool that could not only detect rumors as they spread on Twitter but also predict the veracity of these rumors? This was the genesis of *Hearsift* and *Rumor Gauge*, tools for detection and verification of rumors respectively and the focus of this thesis.

1.1 Terminology

This thesis focuses on rumors that spread on Twitter. As such, we need to define several Twitter specific terms before we proceed any further. These terms will be used through out this thesis.

1.1.1 Twitter

Twitter is a micro-blogging platform has become a major social media platform with hundreds of millions of users. Twitter is a social network where users can publish and exchange short messages of up to 140 characters long, also known as tweets. The ubiquity, accessibility, speed and ease-of-use of Twitter have made it an invaluable communication tool. People turn to Twitter for a variety of purposes, from everyday chatter to reading about breaking news [38].

1.1.2 Retweet

A retweet is a repost or forward of a tweet by another user. It is indicated by the characters *RT*.

1.1.3 Favorite

Favorites are used by users when they like a tweet. By favoriting a tweet a user can let the original poster know that you liked their tweet. The total number of times a tweet has been favorited is visible to everyone.

1.1.4 Verified User

A verified Twitter user is a user that Twitter has confirmed to be the real. Verification is done by Twitter to establish authenticity of identities of key individuals and brands. The verified status of a user is visible to everyone.

1.1.5 Followers

The followers of a user are other people who receive the user's tweets and updates. When a user is followed by someone, it will show up in their followers list. The total number of followers a user has is visible to everyone.

1.1.6 Followees

The followees of a user are other people who the user followers. The total number of followees a user has is also visible to everyone.

1.1.7 Follower Graph

The graph of users on Twitter and their follower relationship. The nodes in the graph are users and the directional edges represent follower relationship between users.

1.2 What is a Rumor?

Before we proceed with the explanation of our algorithms for rumor detection and verification, we need to provide a clear definition of rumors. We define a rumor to an unverified assertion that starts from one or more sources and spreads over time from node to node in a network. On Twitter, a rumor is a collection of tweets, all asserting the same unverified statement (however the tweets could be, and almost assuredly, are worded differently from each other), propagating through the communications network (in this case Twitter), in a multitude of cascades.

A rumor can end in three ways: it can be resolved as either true (factual), false (non-factual) or remain unresolved. There are usually several rumors about the same topic, any number of which can be true or false. The resolution of one or more rumors automatically resolves all other rumors about the same topic. For example, take the number of perpetrators in the Boston Marathon bombings; there could be several rumors about this topic:

1. *Only one person was responsible for this act.*
2. *This was the work of at least 2 or more people.*
3. *There are only 2 perpetrators.*
4. *It was at least a team of 5 that did this.*

Once rumor number 3 was confirmed as true, it automatically resolved the other rumors as well. (In this case, rumors 1 and 4 resolved to be false and rumor 2 resolved to be true.) For the purposes of this thesis, we only consider rumors that spread on Twitter.

1.3 Approach and Contributions

This thesis develops models for detection and verification of rumors (i.e. unverified information) that propagate on Twitter. Detection of rumors about an event is achieved through classifying and clustering assertions made about that event. Assertions are classified through a state-of-the-art speech-act classifier for Twitter developed for this thesis. The classifier is a logistic regression that utilizes a combination of semantic and syntactic features and can identify assertions with 91% accuracy. For verification, we identified salient characteristics of rumors by examining three aspects of diffusion: linguistics, the users involved, and the temporal propagation dynamics. We then identified key differences in each of the three characteristics in the spread of true and false rumors. A time series of these features extracted for a rumor can be classified as predictive of the veracity of that rumor using Hidden Markov Models. The verification algorithm was trained and evaluated on 209 rumors collected from real-world events: the 2013 Boston Marathon bombings, the 2014 Ferguson unrest and the 2014 Ebola pandemic, plus many other rumors reported on Snopes.com and FactCheck.org (websites documenting rumors). The system can predict the veracity of rumors with an accuracy of 75% before verification by trusted channels (trustworthy major governmental or news organizations). The ability to track rumors and predict their outcomes can have immediate real-world relevance for news consumers, financial markets, journalists, and emergency services, and help minimize the impact of false information on Twitter.

Automatic detection and verification of rumors in Twitter are very difficult tasks. Analogous in many ways to speech recognition, they both require extracting weak signals from very noisy environments. Additionally, it is near impossible to get perfect accuracy in both domains. So in addition to accuracy, an insightful way of measuring the performance of our system is to measure the *bandwidth reduction* of information afforded by our system. This is a theme that we will come back to throughout this thesis. Bandwidth reduction is an important measurement because it can help demonstrate the usefulness and utility of our system in real-world situations. For example, a journalist trying to sort out false and true information from millions of tweets about a real-world event (as was the case with the Boston Marathon bombings), has a Sisyphean task. However, as will be shown in great detail later in this thesis, our system can make the task much less daunting and more manageable for our hypothetical journalist by greatly reducing the amount of information he or she has to sort through.

1.3.1 System Overview

Figure 1-1 shows the general pipeline of our system. As can be seen, the system has two major parts, *Hearsift* (rumor detection) and *Rumor Gauge* (rumor verification). The input to *Hearsift* is the raw tweets about an event of interest, with the output being clusters of tweets with each cluster containing tweets that have propagated through Twitter and that make similar assertions about the event in question (e.g., in the case of the Boston Marathon bombings, the tweets making the assertion that there is a third bomb at Harvard square would all be clustered together). From here on these clusters will be called rumors. These rumors are the input to the *Rumor Gauge* algorithm, the output of which are a veracity curve for each rumor indicating the likelihood that the rumor is false or true over time (the algorithm generated a veracity score at every time-step). It should be noted that the system presented here is modular, meaning that each of the two parts can be replaced by other similar systems. For example, *Hearsift* can be replaced by other rumor detection systems, without effecting the internal workings of *Rumor Gauge* and vice-versa.

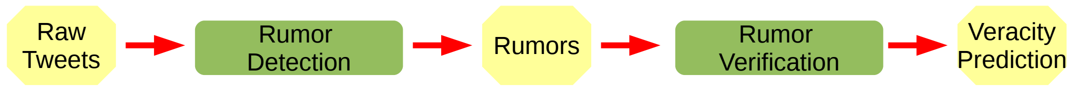


Figure 1-1: The pipeline of the rumor detection and verification system. The inputs and outputs are shown in yellow and the detection and verification subsystems in green. The rumor detection subsystem is named *Hearsift* and the the rumor verification subsystem is named *Rumor Gauge*.

1.4 Overview of Chapters

The rest of this document is organized as follows:

- Chapter 2 review related work.
- Chapter 3 describes the rumor detection subsystem and its evaluation in detail.
- Chapter 4 describes the rumor verification subsystem and its evaluation in detail.
- Chapter 5 discusses hypothetical real-world applications for the systems presented in this thesis.
- Chapter 6 concludes with ongoing and future work and contributions.

Chapter 2

Related Work

This thesis develops a system for detection and verification of rumors about real-world events that propagate on Twitter. Related work includes work on the role of Twitter in real-world emergencies, work from the field of natural language processing about capturing the syntactic and semantic structure of language, work from the field of network science about the diffusion and propagation of information in social networks, and the relatively new work on veracity prediction on Twitter and other domains.

2.1 Role of Twitter in Real-World Emergencies

In addition to being a medium for conversation and idle chatter, Twitter is also used as a source of new for many people [38, 1, 66]. A study by Kwak et al. [52] shows that the majority of trending topics on Twitter are news related. Several bodies of work have shown that Twitter can be used to detect and locate breaking news [86, 88], and to track epidemics [55]. Moreover, the use of Twitter during real-world emergencies has also been studied. These studies have shown the effectiveness of Twitter for reporting breaking news and response and recovery efforts during floods [100, 101, 95], earthquake [47, 26], forest fires [22], and hurricanes [37]. One particular study about wildfires in California [78] outlines the great value Twitter has as a medium to report breaking news more rapidly than mainstream media outlets. Inspired by the close correlation between Twitter activity and real-world events (especially in the case of earthquakes) a new term, *Twicalli scale*[1], was created by researchers for measuring the Twitter impact of real-world events.

2.2 Natural Language Processing

Since this thesis works with Twitter data, we will limit our literature review to relevant natural language processing techniques developed for the Twitter domain. There has been extensive work done on computational methods for analysing the linguistic content of tweets. These works have almost entirely focused on the semantics (e.g., sentiment classification [73]) and the syntactic (e.g., part-of-speech tagging [71]) aspects of tweets. Particularly, sentiment analysis and classification of text has been well studied for Twitter. Most of the work on Twitter sentiment classification either focus on different machine learning techniques (e.g., [104] [104], [40] [40]), novel features (e.g., [21] [21], [49] [49], [85] [85]), new data collection and labelling techniques (e.g., [34] [34]) or the application of sentiment classification to analyse the attitude of people about certain topics on Twitter (e.g., [24] [24], [7] [7]). These are just some examples of the extensive research already done on Twitter sentiment classification and analysis.

Sentiment classification is one of the many language processing techniques that are developed in this thesis, however, more sophisticated techniques such as speech-act classification and the quantification of formality and sophistication of the language used in tweets are also developed in this thesis.

There has been extensive research done on speech act (also known as dialogue act) classification in computational linguistics, e.g., [97, 31, 2, 39]. There are two major annotated corpora used for speech act annotation which all of the research listed use. The Switchboard-DAMSL [42] or SWBD, and the Meeting Recorder Dialog Act [23] or MRDA. SWBD provides an annotation scheme for labelling dialog or speech acts from phone con-

[1]http://mstrohm.wordpress.com/2010/01/15/ measuring-earthquakes-on-twitter-the-twicalli-scale

versations. It also contains labelled speech act data collected from telephone conversations. Similarly, the MRDA corpus provides an annotation scheme and labelled speech act data for face-to-face meetings with multiple parties.

Unfortunately, these annotation schemes and annotated datasets (and some of the methodologies used to analyse them) do not map well to Twitter, given the noisy and unconventional nature of the language used on the platform. As far as we know, there is no publicly available dataset of labelled speech acts for Twitter. Moreover, the only published work on Twitter speech act classification that we are aware of is the work of Zhang et al. on supervised [108] and semi-supervised [109] speech act classification. Our work in this paper is closely related to their supervised speech act classifier. However, Zhang et al. limit their features to a set of words, n-gram phrases and symbols. They do not take into account the rich syntactic structure of the tweets in their classification as their use of syntax is limited to punctuation and a few Twitter specific characters. In their paper, they claim that the noisy nature of Twitter makes the use of syntactic sub-trees, as was done by Jeong et al. [39] in their work on speech act recognition in Emails, impractical.

However, recent advances in Twitter parsers [48] and part-of-speech taggers [71] have made it possible to extract the syntactic structure of tweets with relative ease, without having to normalize the texts as was suggested by Kaufmann and Kalita [46]. In this thesis, we utilized these new tools to create a set of novel syntactic and semantic features. These features are used to train a supervised speech act classifier, using a manually annotated dataset of a few thousand tweets. This combined set of semantic and syntactic features help us achieve state-of-the-art performance for Twitter speech-act classification. This Twitter speech-act classifier is the basis for the rumor detection system presented in this thesis.

As far as we know, there has been very work on measuring the sophistication and formality of language used on Twitter. The most relevant research is work that attempt to detect non-literal text (text that is not supposed to be taken at face value) such as sarcasm [50], satire [9] and hostility (flames) [93] through a combination of semantic and sentiment analytic techniques.

2.3 Modelling Cascades in Networks

There has been extensive work done on modelling the spread of information in networks. The research in this area has mainly focused on modelling various diffusion and cascade structures [17, 35], the spread of "epidemics" [74, 68, 33, 55, 44], knowledge [17], behavior [14] and propaganda [82]. Work has also been done on identifying influential players in spreading information through a network [105, 4, 110, 3] and identifying sources of information [91].

In a work more directly related to our research direction, Mendoza et al, have looked at the difference in propagation behaviour of false rumors and true news on Twitter [64]. Additionally, Friggeri et al. [30] and Jin et al. [41] have analysed the cascade and propagation structures of rumors on social networks. Specifically, Jin et al. analyzed the spread of rumors surrounding the Ebola pandemic and found that rumors can spread just like true news. In all of these cases the properties of the actual entity that is being spread–be it a message, knowledge, or a virus– is never analysed or taken into consideration in the models. In contrast, our work will be looking at the content of the messages being spread in addition to the propagation behaviour of these messages and any information that might be available about the agents involved in the propagation.

2.4 Predicting the Veracity of Information

The field of veracity prediction on social media is a relatively new one. There have so far been only a handful of works that address this problem. Most relevant are the works of Castillo et al. [10] and Kwon et al. [53]. These works deal with propagation of rumors and misinformation on Twitter. Castillo et al. study the propagation of rumors during real-world emergencies while Kwon et al. study the propagation of urban legends (such as bigfoot) on Twitter. The works of Castillo et al. and Kwon et al. propose a combination of linguistics and propagation features that can be used to approximate credibility of information on Twitter. However, Kwon et al.'s work does not deal with rumors surrounding real-world events and Castillo et al.'s work only approximates users' subjective perceptions of credibility on Twitter (i.e. whether users believe the tweets they are reading); they do not focus on objective credibility of messages.

There has also been research done on verification of information on domains other than Twitter. Yang et al. [107] have done work similar to Castillo's work on Sina Weibo, China's leading micro-blogging service. The Washington Post's *TruthTeller*[2] which attempts to fact check political speech in real time utilizes various natural language processing techniques to retrieve relevant information from text (or transcribed speech) and compares the information against a database of known-fact.

[2]http://truthteller.washingtonpost.com/

Chapter 3

Rumor Detection

The general pipeline of the rumor detection and verification system is shown in 1-1. As can be seen in that figure, the system is composed of two main subsystems for rumor detection and rumor verification. This chapter will describe in detail the rumor detection subsystem. An overview of the rumor detection subsystem, henceforth referred to as *Hearsift* can be seen in Figure 3-1. The input to *Hearsift* is a collection of tweets about an event specified by the user through a boolean query (*Boston AND Bombing* in this illustration). *Hearsift* consists of two major parts, an assertion detector and a hierarchical clustering module. Raw tweets about an event feed directly into the assertion detector, which filters the tweets for only those containing assertions. The output of the assertion detector feeds directly into the hierarchical clustering module, the output of which is a collection of clusters. These clusters contain messages that have propagated through Twitter in a multitude of cascades, which we call a rumor. The first part of this chapter describes the assertion detection module and the second part describes the hierarchical clustering module.

3.1 Assertion Detection

An assertion is an utterance that commits the speaker to the truth of the expressed proposition. Figure 3-2 shows two tweets about the Boston Marathon bombings. The tweet shown

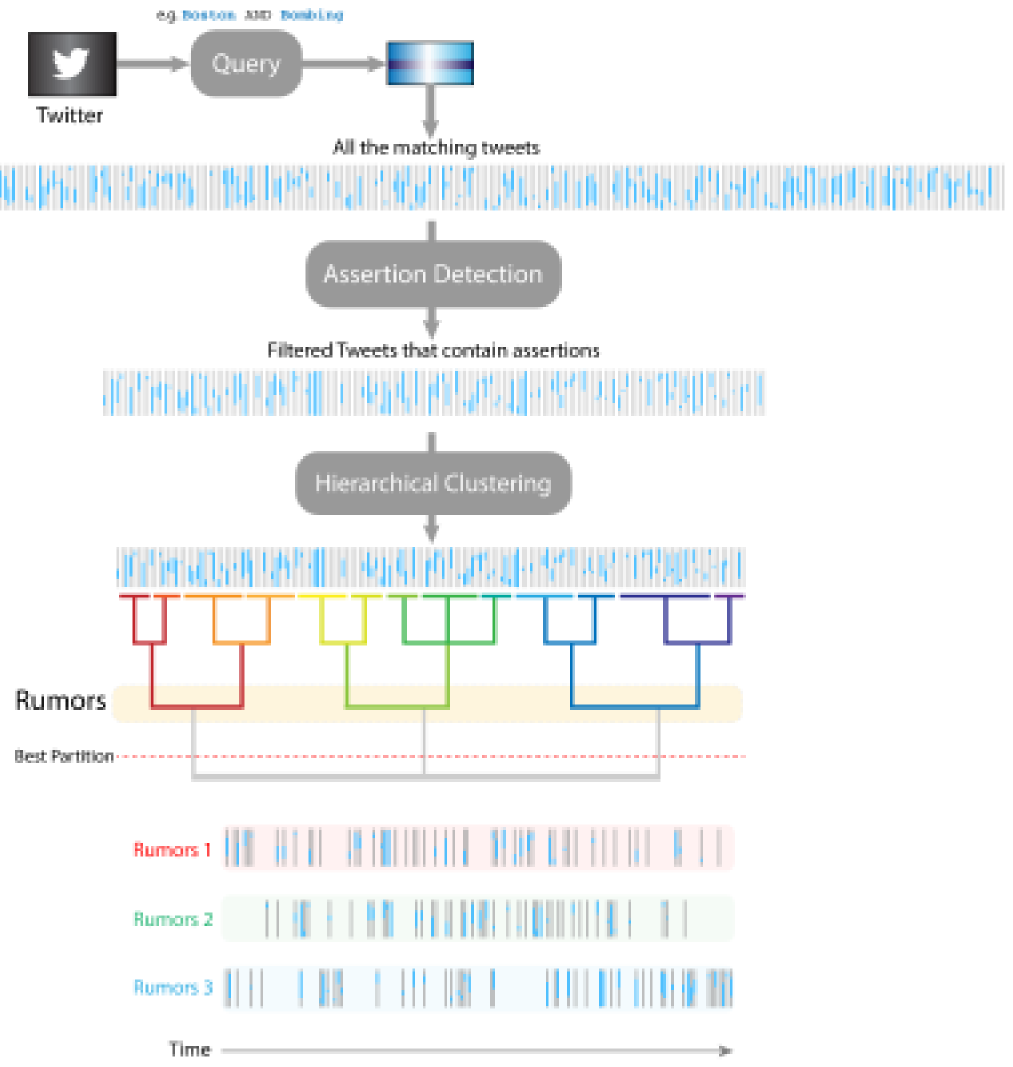
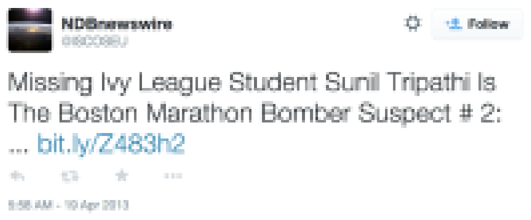


Figure 3-1: The pipeline of *Hearsift*, the rumor detection subsystem. The input to the subsystem is a collection of tweets about an event specified by the user through a boolean query (*Boston AND Bombing* in this illustration). *Hearsift* consists of two major parts, an assertion detector and a hierarchical clustering module. Raw tweets about an event feed directly into the assertion detector, which filters the tweets for only those containing assertions. The output of the assertion detector feeds directly into the hierarchical clustering module, the output of which is a collection of rumors.

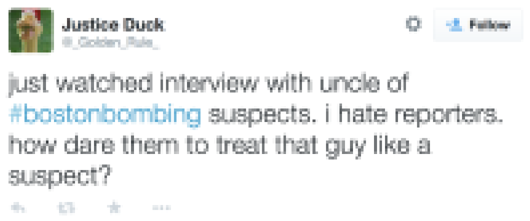
in Figure 3-2a contains an assertion while the tweet shown in Figure 3-2b does not. More generally, assertions are a class of speech-acts. Speech-acts have performative function in language and communication. For instance, we perform speech acts when we offer an apology, greeting, request, complaint, invitation, compliment, or refusal (to name a few). In order to detect assertions in tweets, we require a speech-act classifier for Twitter. Unfortunately, although there has been extensive work done on computational methods for analysing the linguistic content of tweets, these works have almost entirely focused on the semantics (e.g., sentiment classification [73]) and the syntactic (e.g., part-of-speech tagging [71]) aspects of tweets. There has been very little work done on classifying the pragmatics of tweets. Pragmatics looks beyond the literal meaning of an utterance and considers how context and intention contribute to meaning. Speech-acts fall under the definition of pragmatics.

Given the absence of a suitable speech-act classifier for Twitter, a general purpose speech-act classifier for Twitter was developed for this thesis [102]. This classifier can identify several different speech-acts in tweets, including assertions.



Missing Ivy League Student Sunil Tripathi Is The Boston Marathon Bomber Suspect # 2: ... bit.ly/Z483h2

(a) An assertion.



just watched interview with uncle of #bostonbombing suspects. i hate reporters. how dare them to treat that guy like a suspect?

(b) Not an assertion.

Figure 3-2: Two example tweets. Tweet (a) contains an assertion, while tweet (b) does not.

3.1.1 Tweet Acts: A Speech Act Classifier for Twitter

Speech acts are a way to conceptualize speech as action. This holds true for communication on any platform, including social media platforms such as Twitter. In this section of the thesis, we explore speech act recognition on Twitter by treating it as a multi-class classification problem. We created a taxonomy of six speech acts for Twitter and propose a set of semantic and syntactic features. We trained and tested a logistic regression classifier using a data set of manually labelled tweets. Our method achieved a state-of-the-art performance with an average F1 score of more than 0.70. We also explored classifiers with three different granularities (Twitter-wide, type-specific and topic-specific) in order to find the right balance between generalization and over-fitting for our task. Figure 3-3 shows an overview of the Twitter speech-act classifier.

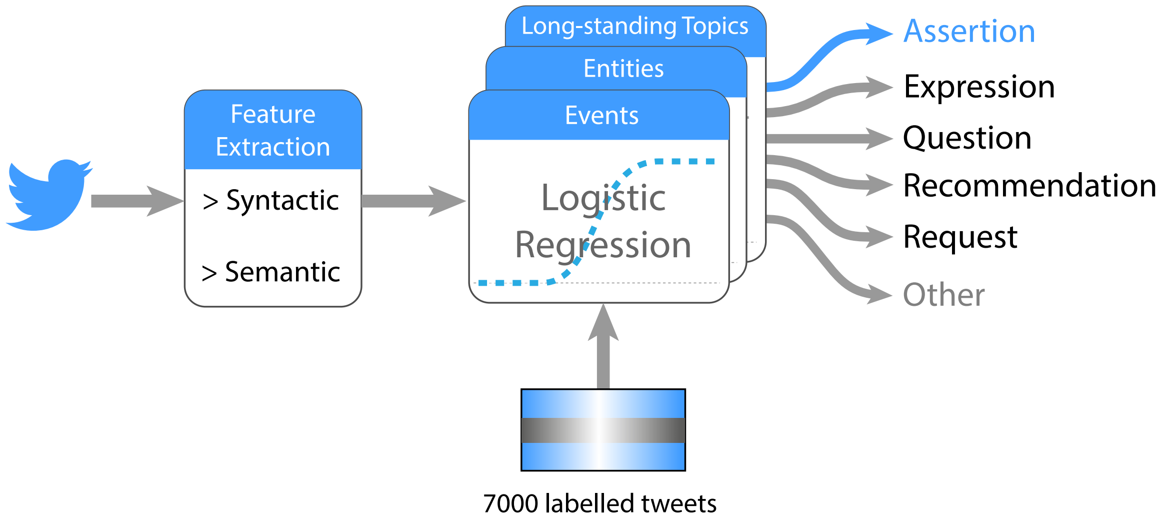


Figure 3-3: The pipeline of the Twitter speech-act classifier which can classify six categories of speech-acts, including assertions which are used for rumor detection.

Note that in addition to allowing us to detect assertions for the purposes of rumor detection, this classifier can be used in other tasks to better understand the meaning and intention behind tweets and uncover the rich interactions between the users of Twitter. Knowing the speech acts behind a tweet can help improve analysis of tweets and give us a better understanding of the state of mind of the users. Additionally, speech acts can help improve various language processing algorithms such as sentiment classification, topic modelling and assertion tracking. For example, knowing that a tweet is ”expressing a feeling” can help in sentiment analysis. Similarly, knowing that a tweet is ”making a statement” can help in tracking the assertions being made about events and people. Finally, knowing the distribution of speech acts of tweets about a particular topic can reveal a lot about the general attitude of users about that topic (e.g., are they confused and are asking a lot of questions? Are they outraged and demanding action? Etc).

3.1.2 Problem Statement

Speech act recognition is a multi-class classification problem. As with any other *supervised* classification problem, we need a large labelled dataset. In order to create such a dataset first we need to create a taxonomy of speech acts for Twitter by identifying and defining a set of commonly occurring speech acts. Next, we need to manually annotate a large collection of tweets using our taxonomy. For our Twitter speech act classifier we assume that there is only one speech act associated with each tweet. This is a valid assumption to make as a starting point given the short nature of communications on Twitter (limited to 140 characters), though we recognize that tweets may sometimes contain multiple acts.

Our primary task is to use the expertly annotated dataset to analyse and select various syntactic and semantic features derived from tweets that are predictive of their corresponding speech acts. Using our labelled dataset and robust features we can train standard, off-the-shelf classifiers (such as SVMs, Naive Bayes, etc) for our speech act recognition task.

A Taxonomy of Twitter Speech Acts

Using Searle's speech act taxonomy [89] as the basis and taking into account the taxonomy used by Zhang et al. [108] for their supervised Twitter speech act classifier, we established a list of six speech act categories that are commonly seen on Twitter. Below we provide the definition and examples (Table 3.1) for each of these speech acts:

- ***Assertion:*** Similar to Searle's *assertive* type, an assertion is a tweet that commits the speaker to the truth of the expressed proposition. So an assertion can be assessed by the "truth value" of its proposition. As Searle puts it, "The simplest test of an assertive is this: you can literally characterize it (inter alia) as true or false." [90].

Examples of assertions include: insisting, stating, hypothesizing, etc.

- ***Recommendation:*** Based on Searle's definition of advise which falls under his *directive* speech act category (directives are acts with the point of getting the hearer to do something). Tweets that recommend (or condemn) things (such as links) or give advise about a situation fall under this category.

- ***Expression:*** Based on Searle's *expressive* type, expressions are tweets that express the speaker's attitudes and emotions towards something. Examples include: celebrating, deploring, liking, thanking, etc.

- ***Question:*** As with recommendation, question falls under the *directive* type of Searle's speech acts. Tweets that are asking for information or confirmation lie under this category.

- ***Request:*** Also a *directive* type, requests are tweets that attempt to get the hearer to do or stop doing something.

- ***Miscellaneous:*** There are several speech act types from Searle's taxonomy that we did not include in our taxonomy. These types include *commissives* that commit the speaker to some future action (such as promising, pledging, taking an oath, etc) and *declaratives* that change the world according to the proposition that was declared (such as firing, declaring war, baptising, etc). As with Zhang et al. [108], we could find relatively few examples of these speech acts on Twitter, not enough to warrant a separate category. Therefore we grouped all remaining speech act types into one *miscellaneous* category.

Type and Topic Specific Classification

Given the diversity of topics talked about on Twitter [57, 111], we wanted to explore topic and type dependent speech act classifiers. Previous research on various natural language

Speech Act	Example Tweet
Assertion	authorities say that the 2 boston bomb suspects are brothers are legal permanent residents of chechen origin - @nbcnews
Recommendation	If you follow this man for updates and his opinions on #Ferguson I recommend you unfollow him immediately.
Expression	Mila Kunis and Ashton Kutcher are so adorable
Question	Anybody hear if @gehrig38 is well enough to attend tonight? #red-sox
Request	rt @craigyh999: 3 days until i run the london marathon in aid of the childrens hopsice @sschospices . please please sponsor me here
Miscellaneous	We'll continue to post information from #Ferguson throughout the day on our live-blog

Table 3.1: Example tweets for each speech act type.

processing algorithms (such as sentiment classification), has shown topic, type and category specific approaches to be superior to more general, one-size-fits-all methods [72, 69, 108].

We used Zhao et al.'s [111] definitions for topic and type. A *topic* is a subject discussed in one or more tweets (e.g., Boston Marathon bombings, Red Sox, global warming, etc). The *type* characterizes the nature of the topic. Zhao et al. have identified three topic types on Twitter, these are:

- ***Entity-oriented topics***: topics about entities such as celebrities (e.g., Ashton Kutcher), brand names (e.g., Pepsi), sports teams (e.g., Red sox), etc.
- ***Event-oriented topics***: topics about events in the world, most commonly about breaking-news (e.g., Boston Marathon bombings).
- ***Long-standing topics***: topics about subjects that are commonly discussed in every-day talk, such as music, global warming, cooking, travelling, etc.

Although several different categorization schemes for topics types on Twitter have been proposed by others [20, 94, 57], we decided to use Zhao et al.'s scheme because it included fewer types (only three) whilst still maintaining a logical division of topics on Twitter. It was important for us to not have too fine-grained division of topics since for a supervised classification task we need a sizeable amount of data per category.

3.1.3 Data Collection and Datasets

We selected two topics for each of the three topic types described in the last section for a total of six topics (see Table 3.2 for list of topics). We collected a few thousand tweets from the Twitter public API for each of these topics using topic-specific queries (e.g., #fergusonriots, #redsox, etc). We then asked three undergraduate annotators to independently annotate each of the tweets with one of the speech act categories described earlier.

We measured the inter-annotator agreement using *Fleiss' kappa*, which calculates the degree of agreement in classification over that which would be expected by chance [28]. The *kappa* score for the three annotators was 0.68, which means that there were disagreements in classification for a good number of the tweets. This is to be expected since our annotators were not expert linguists and because of the noisy and unconventional language used on Twitter. Since the quality of annotation for a supervised classifier is of utmost importance, we decided to only use tweets that were labelled the same by all three annotators, which was around 62% of all tweets. Table 3.2 shows the numbers of tweets per topic that all annotators agreed on.

Type	**Topic**	**# Tweets**
Entity	Red Sox	1418
	Ashton Kutcher	1223
Event	Boston bombings	1517
	Ferguson unrest	1306
Long-standing	Cooking	1098
	Travelling	1001
	Total	7563

Table 3.2: Number of agreed-upon annotated tweets per topic.

The distribution of speech acts for each of the six topics and three types is shown in

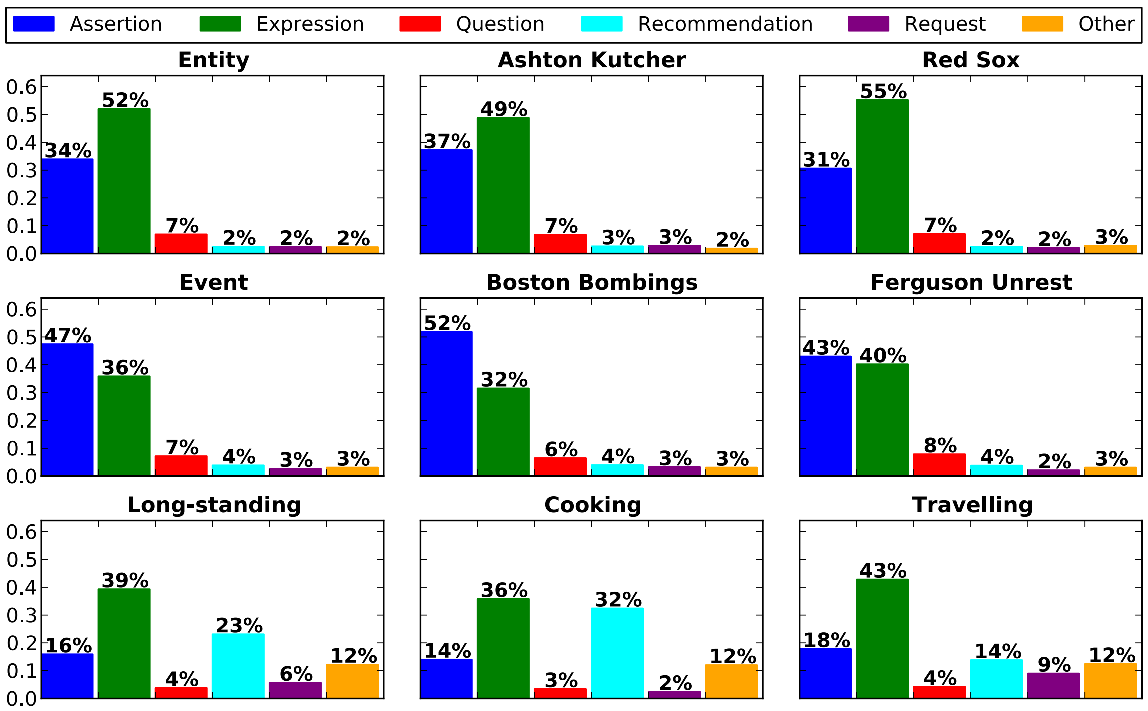


Figure 3-4: Distribution of speech acts for all six topics and three types.

Figure 3-4. There is much greater similarity between the distribution of speech acts of topics of the same type (e.g, Ashton Kutcher and Red Sox) compared to topics of different types. Though each topic type seems to have its own distinct distribution, *Entity* and *Event* types have much closer resemblance to each other than *Long-standing*. Assertions and expressions dominate in *Entity* and *Event* types with questions beings a distant third, while in *Long-standing*, recommendations are much more dominant with assertions being less so. This agrees with Zhao et al.'s [111] findings that tweets about *Long-standings* topics tend to be more opinionated which would result in more recommendations and expressions and fewer assertions.

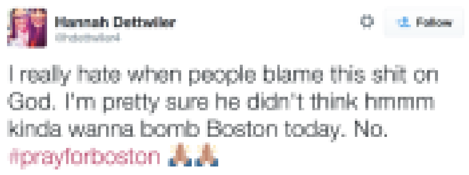
The great variance across types and the small variance within types suggests that a type-specific classifier might be the correct granularity for Twitter speech act classification (with topic-specific being too narrow and Twitter-wide being too general). We will explore this in greater detail in the next sections of this paper.

3.1.4 Features

We studied many features before settling on the features below. Our features can be divided into two general categories: *Semantic* and *Syntactic*. Some of these features were motivated by various works on speech act classification [39, 108, 109]. Overall we selected 3313 binary features, composed of 1647 semantic and 1666 syntactic features.

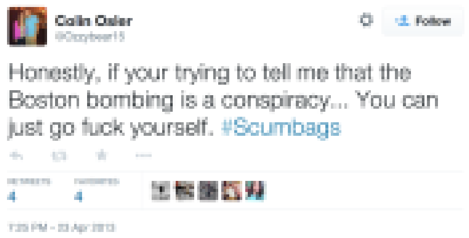
Semantic Features

Below we explain the semantic features in great detail. Examples of each semantic feature can be seen in Figure 3-5.


Hannah Dettwiler

I really hate when people blame this shit on God. I'm pretty sure he didn't think hmmm kinda wanna bomb Boston today. No. #prayforboston

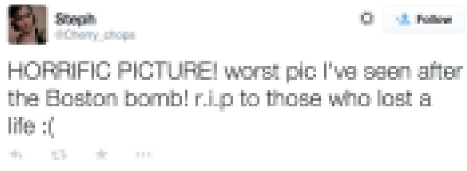
(a) An example of an opinion word being used in a tweet (the word *hate*).


Colin Osler

Honestly, if your trying to tell me that the Boston bombing is a conspiracy... You can just go fuck yourself. #Scumbags

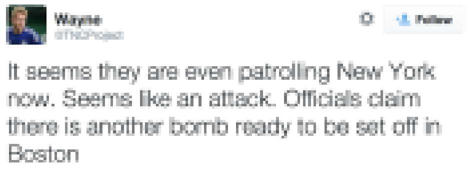
4 4

(b) An example of vulgarity used in a tweet (the word *f**k*).


Steph

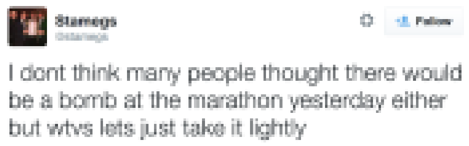
HORRIFIC PICTURE! worst pic I've seen after the Boston bomb! r.i.p to those who lost a life :(

(c) An example of an emoticon used in a tweet (*(:(*)).


Wayne

It seems they are even patrolling New York now. Seems like an attack. Officials claim there is another bomb ready to be set off in Boston

(d) An example of a speech act verb used in a tweet (the word *claim*).


Stamegs

I dont think many people thought there would be a bomb at the marathon yesterday either but wtvs lets just take it lightly

(e) An example of an informative n-gram used in a tweet (the phrase *I don't think*).

Figure 3-5: Example tweets, each containing an example of the semantic features used in the Twitter speech-act classifier.

Opinion Words

We used the ”Harvard General Inquirer” lexicon[1] [98], which is a dataset used commonly in sentiment classification tasks, to identify 2442 strong, negative and positive opinion words (such as *robust*, *terrible*, *untrustworthy*, etc). The intuition here is that these opinion words tend to signal certain speech acts such as expressions and recommendations. One binary feature indicates whether any of these words appear in a tweet.

Vulgar Words

Similar to opinion words, vulgar words can either signal great emotions or an informality mostly seen in expressions than any other kind of speech act (least seen in assertions). We used an online collection of vulgar words[2] to collect a total of 349 vulgar words. A binary feature indicates the appearance or lack thereof of any of these words.

Emoticons

Emoticons have become ubiquitous in online communication and so cannot be ignored. Like vulgar words, emoticons can also signal emotions or informality. We used an online collection of text-based emoticons[3] to collect a total of 362 emoticons. A binary feature indicates the appearance or lack thereof of any of these emoticons.

Speech Act Verbs

There are certain verbs (such as *ask*, *demand*, *promise*, *report*, etc) that typically signal certain speech acts. Wierzbicka [106] has compiled a total of 229 English speech act verbs divided into 37 groups. Since this is a collection of verbs, it is crucially important to only consider the verbs in a tweet and not any other word class (since some of these words can appear in multiple part-of-speech categories). In order to do this, we used Owoputi et

[1]http://www.wjh.harvard.edu/ inquirer/
[2]http://www.noswearing.com/dictionary
[3]http://pc.net/emoticons/

al.'s[4] [71] Twitter part-of-speech tagger to identify all the verbs in a tweet, which were then stemmed using *Porter Stemming* [77]. The stemmed verbs were then compared to the 229 speech act verbs (which were also stemmed using Porter Stemming). Thus, we have 229 binary features coding the appearance or lack thereof of each of these verbs.

N-grams

In addition to the verbs mentioned, there are certain phrases and non-verb words that can signal certain speech acts. For example, the phrase *"I think"* signals an expression, the phrase *"could you please"* signals a request and the phrase *"is it true"* signals a question. Similarly, the non-verb word *"should"* can signal a recommendation and *"why"* can signal a question.

These words and phrases are called n-grams (an n-gram is a contiguous sequence of n words). Given the relatively short sentences on Twitter, we decided to only consider unigram, bigram and trigram phrases. We generated a list of all of the unigrams, bigrams and trigrams that appear at least five times in our tweets for a total of 6738 n-grams (4068 unigrams, 1878 bigrams and 792 trigrams). Instead of manually examining and selecting n-grams from the list, we decided to create an automatic process for the selection of n-grams so that future extensions of this work can be implemented with greater ease.

The selection starts by removing all n-grams that contained topic-specific terms (such as *Boston*, *Red Sox*, etc). Next we removed all n-grams which contained proper nouns (using Owoputi et al.'s [71] part-of-speech tagger). This was an important step since we did not want words or phrases that refer to specific named-entities (such as people, events, places, etc). The next step in the process was the selection of n-grams that are most predictive of the speech act of their corresponding tweets. This was achieved by the calculation of the entropy of each n-gram with respect to the speech act distribution of the tweets that contained them. Entropy is a measure of uncertainty; the more random a source the larger the entropy. So if an n-gram is used evenly in tweets with different speech acts then it will

[4]http://www.ark.cs.cmu.edu/TweetNLP/

have a very high entropy and vice versa. Figure 3-6 shows the speech act distribution of two example n-grams with high and low entropies. As can be seen in the figure, n-grams with lower entropies are more predictive of speech acts. For example, if we knew that a tweet contained the phrase *"affected by"*, we could guess with high confidence that the phrase is an expression. On the other hand, we would not have a confident guess if the phrase was "will be".

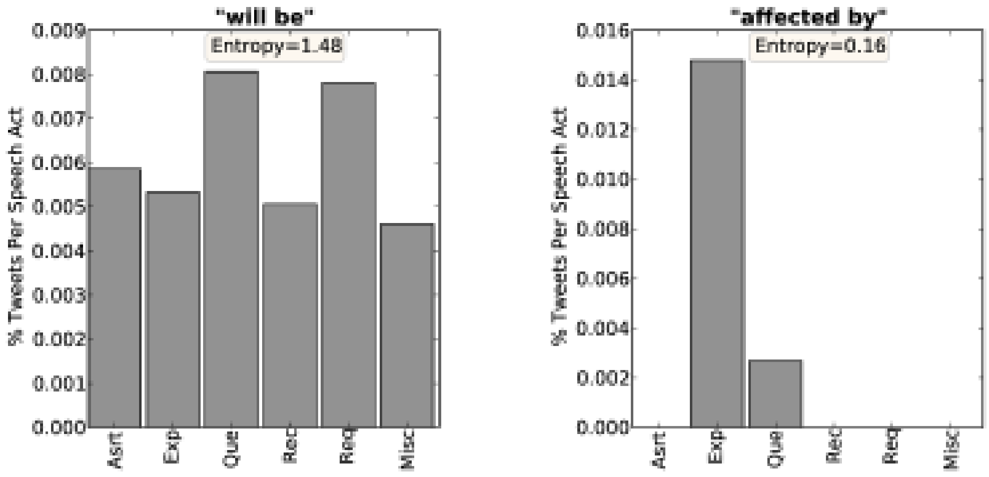


Figure 3-6: Speech act distribution of two example bigrams and their entropies.

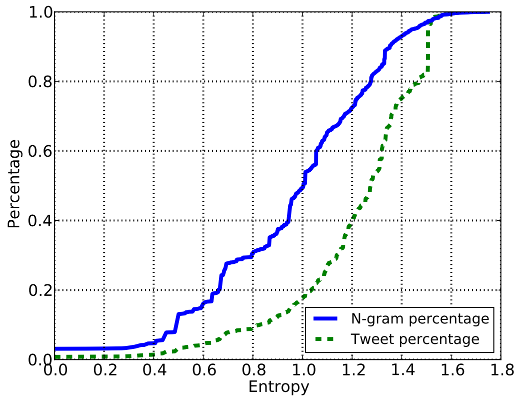
We calculated the entropy of each n-gram using Equation (3.1). Here $H(X)$ is the entropy of the speech act distribution, X, of an n-gram. $P(x_i)$ is the probability that the tweets containing that n-gram are labelled with the speech act x_i, with i looping over all speech acts.

$$H(X) = -\sum_{i \in speech-acts} P(x_i) log P(x_i) \tag{3.1}$$

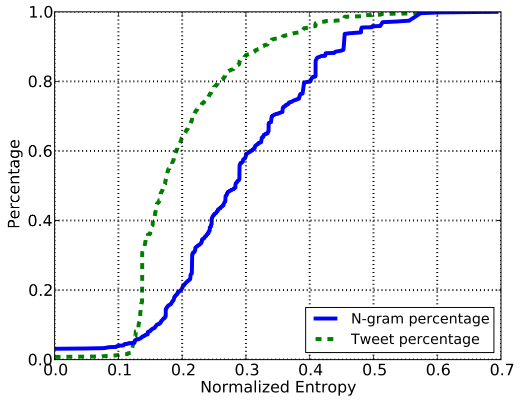
We wanted to pick the n-grams with the lowest entropy. In order to find a logical cut-off point we sorted the n-grams by their entropy and plotted them (Figure 3-7a). The blue (solid) line in Figure 3-7a corresponds to the percentage of n-grams that would be selected for different entropy cut-off values. However, what we found is that many n-grams with low entropies are used very rarely. The green (dashed) line in Figure 3-7a shows the percentage of total tweets that would contain one of the n-grams for each entropy cut-off value. This made it very difficult to find a good cut-off value for entropy since a high cut-off point would result in too many n-gram features and a low cut-off point would result in too few commonly seen n-grams.

In order to account for this, we normalized the entropy of each n-gram by the log of the count of the tweets that contained that n-grams (see Equation (3.2)). This means that the more an n-gram is used the lower its normalized entropy or H_N would be. Figure 3-7b shows the plot of H_N. We decided that a good cut-off value for normalized entropy would be 0.2 which would select 21% of the best (lowest H_N) n-grams that are collectively seen in 64% of our tweets. The top 21% of n-grams is made up of 849 unigrams, 453 bigrams and 113 trigrams for a total of 1415 n-grams. There is a binary feature for each of these n-grams indicating their presence or lack thereof.

$$H_N(X) = \frac{H(X)}{log(\sum_i x_i)} \tag{3.2}$$



(a) Entropy.

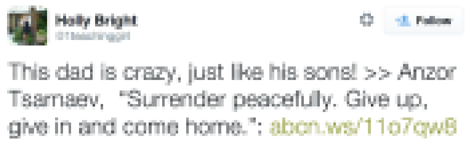


(b) Normalized entropy.

Figure 3-7: Percentage of tweets and n-grams for different (a) entropy and (b) normalized entropy cut-off values.

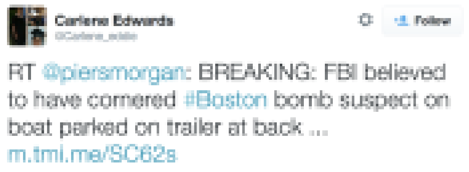
Syntactic Features

Below we explain the syntactic features in great detail. Examples of each syntactic feature can be seen in Figure 3-8.



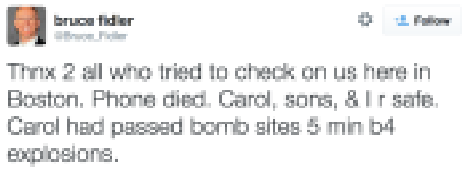
This dad is crazy, just like his sons! >> Anzor Tsarnaev, "Surrender peacefully. Give up, give in and come home.": abcn.ws/11o7qw8

(a) An example of punctuation being used in a tweet (the symbol *!*).



RT @piersmorgan: BREAKING: FBI believed to have cornered #Boston bomb suspect on boat parked on trailer at back ... m.tmi.me/SC62s

(b) An example of Twitter specific characters used in a tweet (the symbols *RT* and *@*).



Thnx 2 all who tried to check on us here in Boston. Phone died. Carol, sons, & I r safe. Carol had passed bomb sites 5 min b4 explosions.

(c) An example of abbreviation being used in a tweet (*thanx*, *b4*, *r*).

Figure 3-8: Example tweets, each containing an example of the syntactic features used in the Twitter speech-act classifier.

Punctuations

Certain punctuations can be predictive of the speech act in a tweet. Specifically, the punctuation *?* can signal a question or request while *!* can signal an expression or recommendation. We have two binary features indicating the appearance or lack thereof of these symbols.

Twitter-specific Characters

There are certain Twitter-specific characters that can signal speech acts. These characters are *#*, *@*, and *RT*. *#*, also known as a hashtag marker, is usually used to tag a tweet with a specific topic. (For example, #redsox tags tweets about the Red Sox baseball team.) Hashtags tend to be used more in expressions and recommendations and less in assertions.

@ is followed by a Twitter user-name and is usually used to reply to someone. It is often used with questions, recommendations, and requests (targeted towards someone). Finally, *RT* is used to indicate a retweet (which is reposting a message). Assertions tend to be retweeted more heavily than other speech acts. (For example a statement by authorities about a breaking news usually is an assertion that gets retweeted a lot.) There are three binary features indicating the presence of these symbols.

As Zhang et al. [108] found, the position of these characters is also important to consider since Twitter-specific characters used in the initial position of a tweet is more predictive than in other positions. This is the case since these characters have different semantics depending on their position. Therefore, we have three additional binary features indicating whether these symbols appear in the initial position.

Abbreviations

Abbreviations are seen with great frequency in online communication. The use of abbreviations (such as *b4* for *before*, *jk* for *just kidding* and *irl* for *in real life*) can signal informal speech which in turn can signal certain speech acts such as expression. We collected 944 such abbreviations from an online dictionary[5] and Crystal's book on language used on the internet [18]. We have a binary future indicating the presence of any of the 944 abbreviations.

Dependency Sub-trees

As discussed earlier, we believe that much can be gained from the inclusion of sophisticated syntactic features such as dependency sub-trees in our speech act classifier. The use of such features has shown great promise in related tasks such as speech act classification in emails [39] and document sentiment classification [63, 67]. We used Kong et al.'s [48] Twitter dependency parser for English (called the *TweeboParser*) to generate dependency trees for our tweets. Dependency trees capture the relationship between words in a sentence. Each

[5]http://www.netlingo.com/category/acronyms.php

node in a dependency tree is a word with edges between words capturing the relationship between the words (a word either modifies or is modified by other words). In contrast to other syntactic trees such as *constituency trees*, there is a one-to-one correspondence between words in a sentence and the nodes in the tree (so there are only as many nodes as there are words). Figure 3-9 shows the dependency tree of an example tweet.

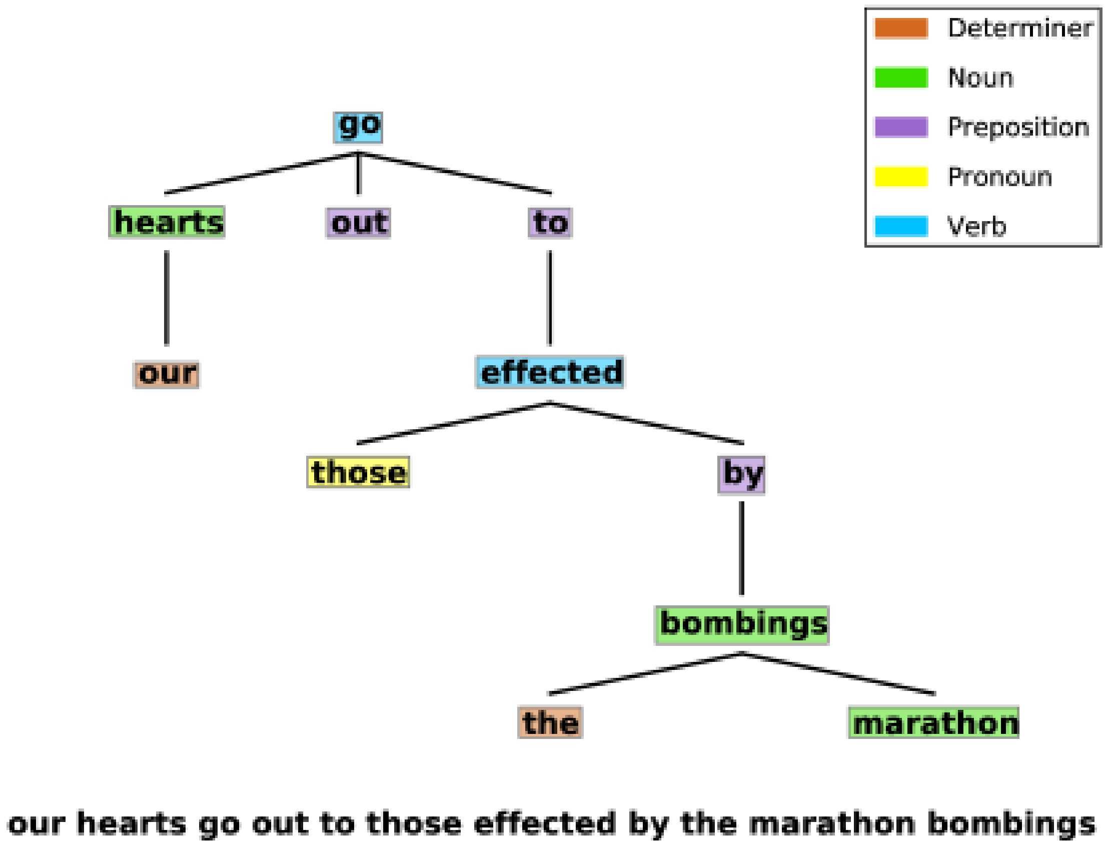


Figure 3-9: The dependency tree and the part of speech tags of a sample tweet.

We extracted sub-trees of length one and two (the length refers to the number of edges) from each dependency tree. In our example tree in Figure 3-9, there are ten sub-trees of length-one (such as *go–hearts*, *effected–those*, etc) and 12 sub-trees of length-two (such as *by–effected–those*, *bombings–the–marathon*). Overall we collected 5484 sub-trees that appeared at least five times (4250 length-one sub-trees and 1234 length-two sub-trees). We then used a selection process identical to the one used for n-grams: removing sub-trees that contain proper nouns or topic-specific words (like the *bombings–the–marathon* sub-tree) and calculating the normalized entropy for each sub-tree. As with n-grams, we sorted the sub-trees by their normalized entropy and plotted them with respect to how much data they capture. This was done in order to pick a good cut-off for the normalized entropy that selects predictive sub-trees but does not over-fit the data by selecting too many sub-trees. Figure 3-10 shows this plot. Through examining the plot we decided that a good cut-off value for normalized entropy would be 0.2 which would select 30% of the best sub-trees that are collectively seen in 38% of our tweets. The top 30% of the sub-trees is made up of 1109 length-one sub-trees and 546 length-two sub-trees for a total of 1655 sub-trees. There is a binary feature for each of these sub-trees indicating their appearance.

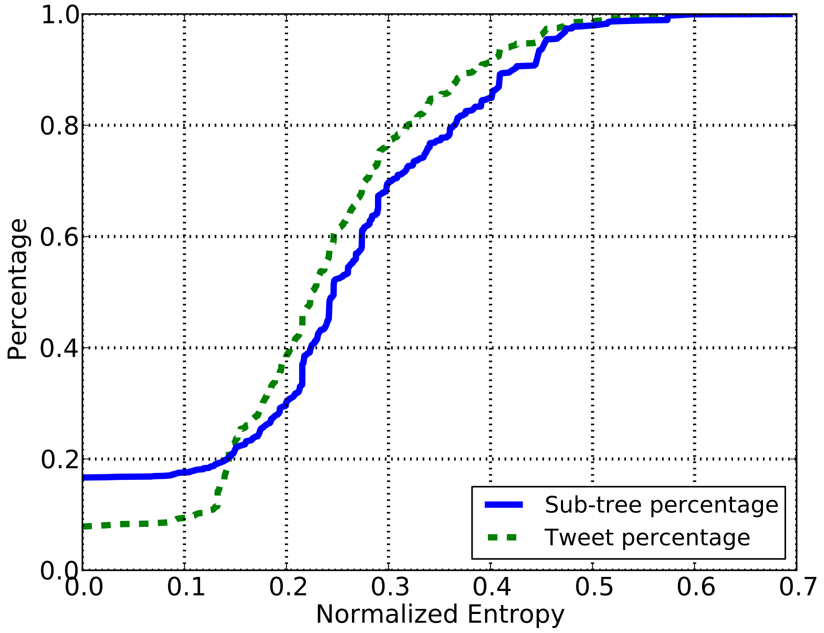


Figure 3-10: Percentage of tweets and sub-trees for different normalized entropy cut-off values.

Part-of-speech

Finally, we used the part-of-speech tags generated by the dependency tree parser to identify the use of adjectives and interjections (such as *yikes*, *dang*, etc). Interjections are mostly used to convey emotion and thus can signal expressions. Similarly adjectives can signal expressions or recommendations. We have two binary features indicating the usage of these two parts-of-speech.

3.1.5 Supervised Speech Act Classifier

We used the Python Scikit-learn toolkit [75] to train four different classifiers on our 3313 binary features using the following methods: *naive bayes (NB)*, *decision tree (DT)*, *logistic regression (LR. Also known as max entropy)* and *linear support vector machine (SVM)*. Additionally, given the highly uneven distribution of speech acts, it was particularly important to compare the performance of our classifiers against a *baseline (BL)* classifier to assess relative performance gains with our features. The baseline classifier always selects the most likely speech act from the prior speech act distribution. We trained classifiers across three granularities: *Twitter-wide*, *Type-specific*, and *Topic-specific*. All of our classifiers are evaluated using 20-fold cross validation. We used a high number of folds since we have limited training data, especially when training fine-grained topic-specific classifiers.

Twitter-wide classifier

Table 3.3 shows the performance of our five classifiers trained and evaluated on all of the data. We report the F1 score for each class, which is the harmonic mean of recall and precision for that class (see Equation (3.3)). The average F1 score we report is the weighted average according to the size of each class. As shown in Table 3.3, all of the classifiers significantly outperform the baseline classifier with logistic regression being the overall best performing classifier with a weighted average F1 score of .70. With the exception of the *recommendation* class, logistic regression beat the other classifiers across all other speech acts. Thus we picked logistic regression as our classier and the rest of the results reported will be for LR only.

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3.3}$$

	As	Ex	Qu	Rc	Rq	Mis	Avg
BL	0.	.59	0.	0.	0.	0.	.24
DT	.57	.68	.79	.32	0.	.29	.58
NB	.72	.76	.71	**.40**	0.	.41	.66
SVM	.71	.80	.86	.35	.13	.43	.69
LR	**.73**	**.80**	**.87**	.30	**.16**	**.45**	**.70**

Table 3.3: F1 scores for each speech act category. The best scores for each category are highlighted.

Type-specific classifier

Next, we trained and evaluated logistic regression classifiers for each of our three topic types. Table 3.4 shows the performance of each classier. Overall *entity* and *event* classifiers outperform the *long-standing* topic classifier. This could be attributed to the prior distribution of speech acts in the training data (as well its poor performance on the *assertion* class). As shown in earlier parts of this paper, *long-standin*g topics have a much more even distribution of speech acts compared to *entity* and *event* topics that are mostly dominated by assertions and expressions with questions in a distant third (see Figure 3-4).

Furthermore, it is interesting to note that there is not a single type-specific classifier that performs best for all speech acts, with the entity-classifier doing best on the *expression* and *recommendation* categories, event-classifier doing best on the *assertion* and *request* categories, and the long-standing-classifier doing best on the *miscellaneous* category, with all doing extremely well on the *question* category.

	As	Ex	Qu	Rc	Rq	Mis	Avg
BL_{ent}	0.	.68	0.	0.	0.	0.	.35
LR_{ent}	.75	**.94**	**.98**	**.53**	0.	.14	**.79**
BL_{evt}	.65	0.	0.	0.	0.	0.	.31
LR_{evt}	**.86**	.80	.97	.24	**.33**	.20	.74
BL_{lst}	0.	.56	0.	0.	0.	0.	.21
LR_{lst}	.51	.78	.98	.35	0.	**.42**	.61
Avg_{all}	.71	.84	.98	.37	.11	.25	.71

Table 3.4: F1 scores for each speech act category for all three topic types and their average. (ent=entity, evt=event, lst=long-standing.)

Topic-specific classifier

Finally, we trained and evaluated logistic regression classifiers for each of our six topics. Table 3.5 shows the performance of each classifier.

3.1.6 Analysis

The topic-specific classifiers' average performance was better than that of the type-specific classifiers (.74 and .71 respectively) which was in turn marginally better than the performance of the Twitter-wide classifier (.70). This is in spite of the fact that the Twitter-wide classifier was trained on a dataset which was on average three times larger than the datasets used for the type-specific classifiers, which were themselves about two times larger than the datasets used for the topic-specific classifiers. This confirms our earlier hypothesis that the more granular type and topic specific classifiers would be superior to a more general Twitter-wide classifier. This also agrees with previous works on different natural language processing algorithms that show topic and type specific approaches outperform more general ones [72, 69, 108].

Even though the topic-specific classifiers have the best performance, their granularity is too fine-grained. This means that every time we want speech acts classified for tweets about a new topic, a new training set needs to be annotated and a new classifier trained. This makes topic-specific classifiers somewhat impractical. We believe that type-specific

	As	Ex	Qu	Rc	Rq	Mis	Avg
BL_{ash}	0.	.60	0.	0.	0.	0.	.25
LR_{ash}	.80	.90	.97	.52	0.	.16	.77
BL_{rs}	0.	.62	0.	0.	0.	0.	.27
LR_{rs}	.73	**.95**	.97	**.62**	0.	.24	**.81**
Avg_{ent}	.77	.93	.97	.57	0.	.20	.79
BL_{bos}	.60	0.	0.	0.	0.	0.	.25
LR_{bos}	**.87**	.82	.99	.53	0.	.22	.76
BL_{fer}	.67	0.	0.	0.	0.	0.	.34
LR_{fer}	.80	.86	.99	.37	**.15**	.14	.75
Avg_{evt}	.84	.84	.99	.45	.08	.18	.76
BL_{ck}	0.	.56	0.	0.	0.	0.	.21
LR_{ck}	.58	.85	**.99**	.53	0.	.39	.69
BL_{tr}	0.	.59	0.	0.	0.	0.	.24
LR_{tr}	.59	.82	.98	.43	0.	**.42**	.68
Avg_{lst}	. 59	.84	.99	. 48	0.	.41	.69
Avg_{all}	.73	.87	.98	.57	.03	.26	.74

Table 3.5: F1 scores for each speech act category for all six topics and their average. (ash=Ashton Kutcher, rs=Red Sox, bos=Boston Marathon bombings, fer=Ferguson riots, ck=cooking, tr=travelling.)

classifiers are the correct granularity. Not only do they outperform the Twitter-wide classifier, they also are limited to three classifiers and any new topic can be assigned to one of the three.

Next, we wanted to measure the contributions of our semantic and syntactic features. To achieve that, we trained two versions of our Twitter-wide logistic regression classifier, one using only semantic features and the other using syntactic features. As shown in Table 3.6, the semantic and syntactic classifiers' performance was fairly similar, both being on average significantly worse than the combined classifier. It is interesting to note that neither the semantic nor the syntactic classifiers perform better on all the speech act categories. The semantic classifier performs best on the *assertion*, *recommendation* and *miscellaneous* categories while the syntactic classifier performs best on the *expression* and *question* categories. Also with the notable exception of the *question* category, the combined classifier outperform the semantic and syntactic classifiers on all other categories, which strongly suggests that both feature categories contribute to the classification of speech acts. It is also interesting to note that for the *request* category, both the semantic and the syntactic classifiers had an F1 score of 0 (which corresponds to zero recall and precision), while the combined classifier had a F1 score of .16.

	As	Ex	Qu	Rc	Rq	Mis	Avg
Sem	.71	.80	.62	.22	0.	.23	.64
Syn	.59	**.81**	**.94**	.12	0.	0.	.62
All	**.73**	.80	.87	**.30**	**.16**	**.45**	**.70**

Table 3.6: F1 scores for each speech act category for semantic and syntactic features.

We can not directly compare the performance of our classifier to the only other supervised Twitter speech act classifier by Zhang et al. [108], since we used different training datasets and used a different taxonomy for Twitter speech acts (differing both in definition and number). However, we can still make a qualitative comparison of the two classifiers, especially since they were both trained on similarly sized datasets (a few thousand tweets each). Our general (Twitter-wide) speech act classifier had a weighted average F1 score of .70 compared to their weighted average score of .64. This is despite the fact that our classification problem had six classes while theirs problem had five.

3.1.7 Detecting Assertions

For the purposes of rumor detection, the job of the Twitter speech-act classifier developed is to detect assertions, since it is assertions that form the basis of our rumors. Given that we are looking at rumors about real-world events, we use the event-specific classifier. Furthermore, since we only care about the assertion category, we can turn the speech-act classifier into a binary classifier for identifying assertions by treating all other classes as one *other* class. As shown in Table 3.4, the *F1* score for classifying assertions using the event-specific classifier is .86. In order to better understand the performance of the event-specific classifier for identifying assertions we look at the *receiver operating characteristic (ROC)*, or the ROC curve of the classifier. An ROC curve is a plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the *true positive rate (TPR)* against the *false positive rate (FPR)* at various threshold settings. The definitions for TPR and FPR are shown in Equation (3.4). In the equation, *TP* stands for true positive, *TN* for true negative, *FP* for false positive, and *FN* for false negative.

$$TPR = \frac{TP}{TP+FN} \quad FPR = \frac{FP}{FP+TN} \tag{3.4}$$

Using the TPR and FPR we can plot the ROC curve of the assertion classifier. Figure 3-11 shows this ROC curve. According to the ROC curve, at a false positive rate of .28, the true positive rate would be 1.0. This means that in order to get all the tweets containing assertions correctly classified, we would have to tolerate 28% of non-assertion containing tweets mistakenly classified as containing assertions. Depending on the application that system is being used for one might be able to tolerate such a noise in order to capture all assertion containing tweets. However, if the application requires much more precision, then a different point on the operating curve can be picked. For example, at only 15% false positive rate, the tool will correctly identify 90% of all tweets containing assertions. This parameter is under the control of the user who can tune it to different values depending on the application.

Going back to the theme of bandwidth reduction discussed in the introduction chapter of this thesis, we would like to estimate the bandwidth reduction afforded by the assertion classifier. Almost half of the tweets about real-world events do not contain assertions (as shown in Figure 3-4), meaning that the bandwidth reduction afforded by the assertion classifier is around 50%. Case in point, for the Boston Marathon bombings event, there were around 20 million tweets about the event with about 10 million of those containing assertions.

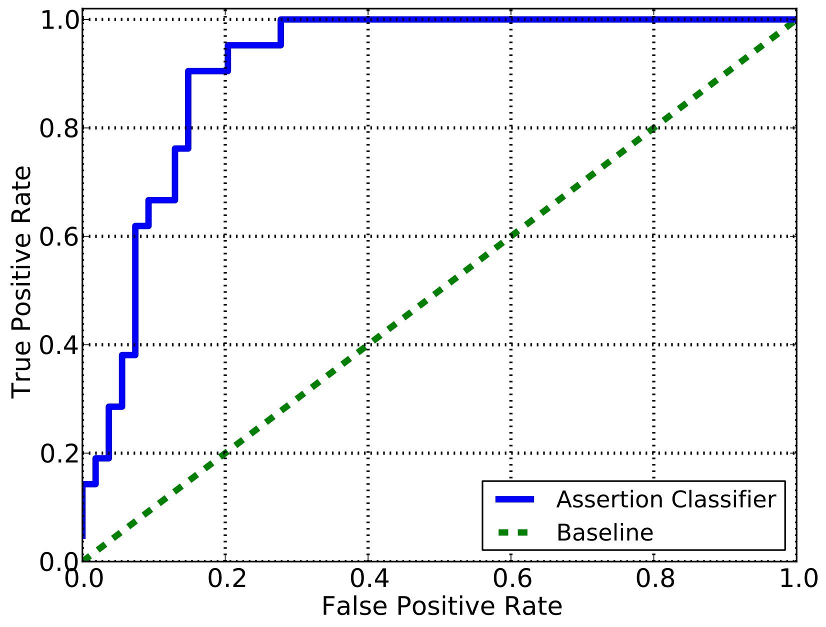


Figure 3-11: The receiver operating characteristic (ROC) curve of the event-specific assertion classifier.

3.2 Clustering of Assertions

The second module of *Hearsift*, shown in Figure 3-1, is clustering module. This module takes as input the output of the assertion detector, which is tweets containing assertions. The output of the clustering module is a collection of clusters, each containing tweets with similar assertions. We call each collection of assertions that have propagated through Twitter a rumor; therefore, the output of the clustering module is a collection of rumors. Below, we describe the clustering method used for this purpose.

3.2.1 Hierarchical Clustering of Tweets

In order to cluster tweets containing assertions, we used Hierarchical Agglomerative Clustering (HAC). Generally speaking, hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. There are two strategies for hierarchical clustering [60]:

- *Agglomerative*: This is a "bottom up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

- *Divisive*: This is a "top down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

The complexity of agglomerative clustering is polynomial at $O(n^3)$, while the complexity of divisive clustering is exponential at $O(2^n)$. Given the relatively large size of our datasets, we decided to use agglomerative clustering, given its lower complexity.

Similarity Function

For agglomerative clustering, there needs to be a way to decide which clusters should be combined. This is achieved through the use of a metric that measures the distance between pairs of observations, or tweets in our case. The similarity function that we used for HAC of tweets is *TF-IDF* combined with *cosine similarit*y. TF-IDF, or Term FrequencyInverse Document Frequency, is a method of converting text into numbers so that it can be represented meaningfully by a vector [80]. TF-IDF is the product of two statistics, *TF* or Term Frequency and *IDF* or Inverse Document Frequency.

Term Frequency measures the number of times a term (word) occurs in a document. Since each document will be of different size, we need to normalize the document based on its size. We do this by dividing the Term Frequency by the total number of terms. TF considers all terms as equally important, however, certain terms that occur too frequently should have little effect (for example, the term *"the"*). And conversely, terms that occur less in a document can be more relevant. Therefore, in order to weigh down the effects of the terms that occur too frequently and weigh up the effects of less frequently occurring terms, an Inverse Document Frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. Generally speaking, the Inverse Document Frequency is a measure of how much information a word provides, that is, whether the term is common or rare across all documents.

The exact formula for calculating TF is shown in Equation (3.5). Here, *t* is the term being processed, *d* is the document, and the function *f(t,d)* measure the raw frequency of *t* in d.

$$TF(t,d) = 0.5 + \frac{0.5 \times f(t,d)}{max\{f(w,d) : w \in d\}} \tag{3.5}$$

The formula for calculating IDF is shown in Equation (3.6). Here, *N* is the total number of documents in the corpus, and $|\{d \in D : t \in d\}|$ is the number of documents where the term *t* appears.

$$IDF(t,D) = \log \frac{N}{1 + |\{d \in D : t \in d\}|} \tag{3.6}$$

Using the definitions of TF and IDF, the TF-IDF is then calculated as shown in Equation (3.7).

$$TFIDF(t,d,D) = TF(t,d) \times IDF(t,D) \tag{3.7}$$

Using TF-IDF, we derive a vector for each tweet. The set of tweets in our collection is then viewed as a set of vectors in a vector space with each term having its own axis. We measure the similarity between two tweets using the formula shown in Equation (3.8). Here, $d1 \cdot d2$ is the dot product of two documents, and $||d1|| \times ||d2||$ is the product of the magnitude of the two documents.

$$Similarity(d1,d2) = \frac{d1 \cdot d2}{||d1|| \times ||d2||} \tag{3.8}$$

Hierarchical Agglomerative Clustering

Using the similarity function that was described, we can use hierarchical agglomerative clustering (HAC) to cluster similar assertions together. The output of HAC can best be described as a dendrogram. A dendrogram is a tree diagram that can be used to illustrate the arrangement of the clusters produced by HAC. Figure 3-12 shows a sample dendrogram depicting HAC. As can be seen in the figure, at the very first level all tweets belong to their own cluster, so there are as many clusters as there are tweets. At the very root of the tree, is a single cluster, containing all the tweets. It is up to the user to pick the level at which the clusters are to be used (this is called the *best partition* in the figure). These clusters are what we call rumors. A partition higher in the tree (further from the root) would yield more rumors, with each rumor containing fewer number of tweets. Conversely, a partition lower in the tree would yield less rumors, with each containing greater number of tweets. Depending on the application and the particular event that is being looked at, different *best partitions* can be picked by the user. For example, if the event in question is a very local event, meaning that there are not many tweets about the event, then perhaps a partition higher in the tree would be more useful and vice-versa.

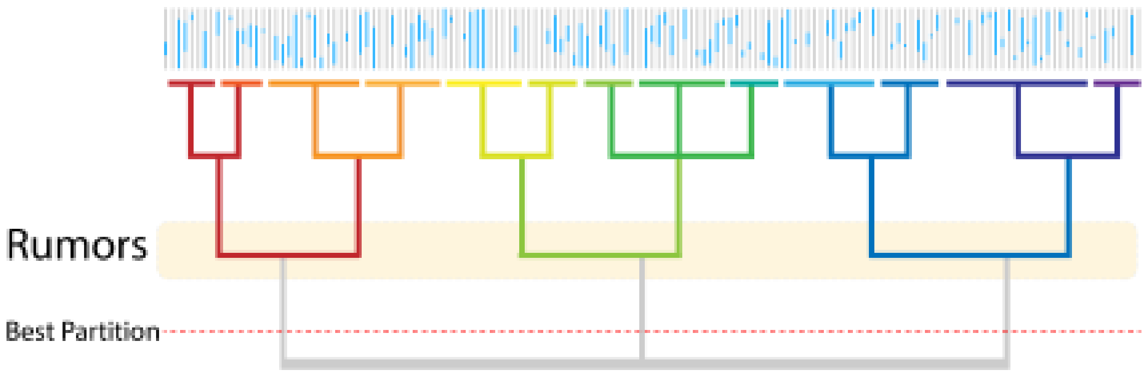


Figure 3-12: A sample dendrogram depicting hierarchical agglomerative clustering. The algorithm starts by assign each tweet its own cluster. At every step, the most similar clusters are merged (two clusters at a time). At the very root of the tree is a single cluster, containing all the tweets. The user has to pick the level at which the clusters are to be used (called the *best partition* in the picture). These clusters are what we call rumors.

Once again, going back to the theme of bandwidth reduction, we would like to estimate the bandwidth reduction afforded by hierarchical agglomerative clustering. Of course, the bandwidth reduction would depend on the level of the partition, but generally speaking, the number of clusters or rumors is somewhere between tens to hundreds of rumors, depending on the size of the event. For example, for the Boston Marathon bombings event, the 10 million tweets containing assertions yielded somewhere between 100 and 1000 rumors (with different partitions). This is a reduction by four orders of magnitude.

Chapter 4

Rumor Verification

In addition to the rumor detection module described in the previous chapter, the other major module of the rumor detection and verification system shown in 1-1 is the rumor verification module. This chapter will describe in detail this system. An overview of the rumor detection subsystem, henceforth referred to as *Rumor Gauge* can be seen in Figure 4-1.

The input to *Rumor Gauge* is a collection of rumors. A rumor about an event is a collection of tweets that have spread through Twitter, all making similar assertions about the event in question. For example, a rumor that spread on Twitter about the Boston Marathon bombings was that there were bombs in Harvard square. There were thousands of tweets making the same statement, each maybe worded differently. All of these tweets bundled together would constitute a rumor. It should be noted that a rumor can end up being true or false. The input to *Rumor Gauge* could be the output of the rumor detection system-*Hearsift*- explained in the previous chapter or it could be the output of any other event/rumor detection system, including manual rumor identification systems (i.e., rumors identified manually through inspection).

Rumor Gauge extracts time-series features about the linguistic content of the rumors, the identity of the users involved in the propagation of the rumors and the propagation dynamics of the rumors. These features are then passed to a Hidden Markov Model (HMM) trained on 209 manually annotated rumors which predicts the veracity of the rumors. For each rumor, the output of the system is a veracity curve that shows the predicted veracity of the rumor over time. The purpose of *Rumor Gauge* is to correctly predict the veracity of rumors before verification by trusted channels (trustworthy major governmental or news organizations). *Rumor Gauge* can do that for 75% rumors it was evaluated on. This chapter explains in detail how this was achieved.

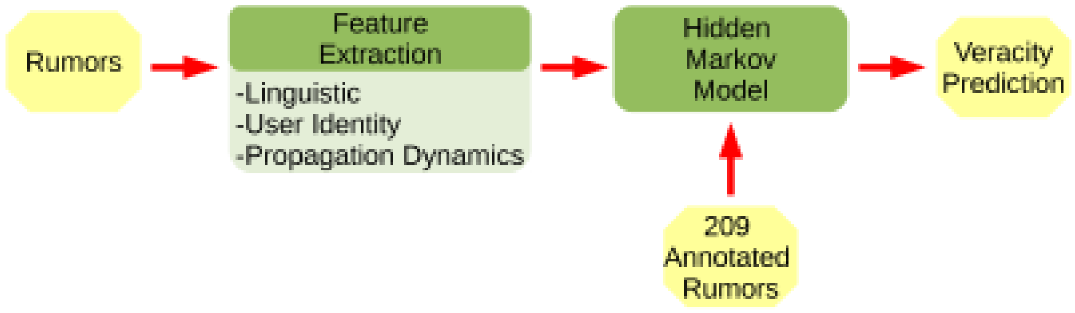


Figure 4-1: The pipeline of *Rumor Gauge*, the rumor verification subsystem. The input to this subsystem is rumors on Twitter about an event (rumor is defined as a collection of tweets containing similar assertions that have spread over Twitter). *Rumor Gauge* extracts time-series features about the linguistic content of the rumors, the identity of the users involved in the propagation of the rumors and the propagation dynamics of the rumors. These features are then passed to a Hidden Markov Model (HMM) trained on 209 annotated rumors which predicts the veracity of the rumors.

4.1 Methodology

As explained earlier, the purpose of *Rumor Gauge* is to correctly predict the veracity of rumors before verification by trusted channels, where trusted channels are defined as trustworthy major governmental or news organizations. The intuition behind *Rumor Gauge* is that even before trusted verification, there are signals, however weak, that are predictive of the veracity of rumors. Based on previous research on spread of gossips in networks [8, 29], and the study of gossips and rumors from the fields of psychology and sociology

[92, 84], we have identified salient characteristics of rumors by examining three aspects of diffusion: linguistics, the users involved, and the temporal propagation dynamics. Each of these aspects is composed of several features. These features are explained in detail later in this chapter.

Since rumors are temporal in nature (i.e., the tweets that make up a rumor are tweeted at different times), time series of these features are extracted. These temporal features are extracted from 209 manually annotated rumors and used to train HMMs for true and false rumors. It should be noted that features are extracted only from data before trusted verification. When a new rumor is detected and passed to *Rumor Gauge*, the same temporal features are extracted at every time-step as the rumor spreads. At every time-step the temporal features are passed the HMMs for true and false rumors. Each HMM measures the fitness of the data to its model and returns a probability score. These probability scores are then compared to predict the veracity of the rumor. As described earlier, the goal is to get a correct veracity prediction for a rumor before trusted verification. Figure 4-2 shows an overview of our method.

4.2 Data Collection and Datasets

Our model was trained and evaluated on 209 rumors collected from real-world events: the 2013 Boston Marathon bombings, the 2014 Ferguson unrest, and the 2014 Ebola epidemic, plus many other rumors reported on Snopes.com and FactCheck.org (websites documenting rumors). These rumors were manually selected and annotated. Table 4.1 shows the distribution of the rumors. Out of the 209 rumors, 113 (54%) were false and 96 (46%) were true. Below we provide a brief description of each of the events or sources of the 209 rumors:

- *2014 Boston Marathon bombings*: The 2013 Boston Marathon bombings were a series of attacks and incidents which began on April 15, 2013, when two pressure cooker bombs exploded during the Boston Marathon at 2:49 pm EDT, killing 3 peo-

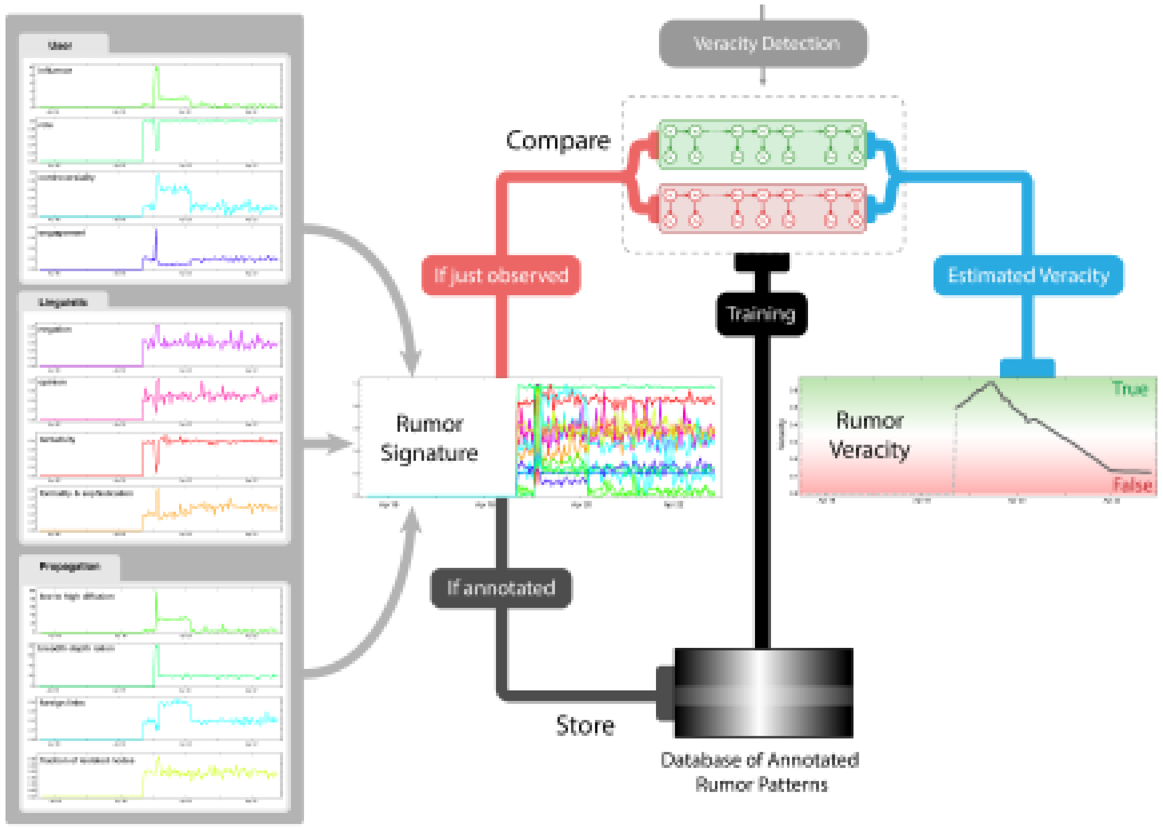


Figure 4-2: An overview of the approach used for predicting the veracity of rumors. As a rumor spreads, at every time-step several time-series features are extracted from the rumor and passed to HMMs trained on false and true rumors. Each HMM measures the fitness of the data to its model and returns a probability score. These probability scores are then compared to predict the veracity of the rumor for each time step. Over time the system generates a veracity curve.

ple and injuring an estimated 264 others. The events after the bombing led to an MIT police officer being killed, a manhunt for the suspects, and the lockdown of the city of Boston and neighbouring towns. [1]

- *2014 Ferguson unrest*: The 2014 Ferguson unrest was a series of protest that began the day after Michael Brown was fatal shot by Darren Wilson, a policeman, on August 9, 2014, in Ferguson, Missouri. [2]
- *2014 Ebola epidemic*: The 2014 Ebola epidemic is the first Ebola outbreak to each epidemic proportions. It originated in several West African countries, causing significant mortality, with reported case fatality rates of up to 70%. Imported cases in the United States and Spain led to secondary infections of medical workers but did not spread further. [3]
- *Snopes.com*: Snopes.com is a website that documents Internet rumors, urban legends, and other stories of unknown or questionable origin. It is a well-known resource for validating and debunking rumors. [4]
- *FactCheck.org*: FactCheck.org is a website that documents inaccurate and misleading claims. The website also documents and verifies online rumors in its *Ask FactCheck* section. [5]

The rumors selected from these events and sources were manually annotated. This entailed not only identifying the rumors, but also creating boolean queries using terms describing each rumor that could be used to select tweets that talked about that rumor. For example, if the rumor in question was that *"there is a bomb at Harvard square"*, the query *"bomb AND Harvard"* could be used to pick out tweets talking about that rumor. Additionally, the *trusted verification* time of each rumor was also manually annotated.

[1]http://en.wikipedia.org/wiki/Boston_Marathon_bombings
[2]http://en.wikipedia.org/wiki/Ferguson_unrest
[3]http://en.wikipedia.org/wiki/Ebola_virus_epidemic_in_West_Africa
[4]www.snopes.com
[5]www.factcheck.org

Source/Event	**False Rumors**	**True Rumors**	**Total**
2013 Boston Marathon Bombings	16	6	22
2014 Ebola Pandemic	11	9	20
2014 Ferguson Unrest	10	7	17
Snopes.com & Factcheck.org	76	74	150
All	113	96	209

Table 4.1: Distribution of manually annotated rumors used for training and evaluating the rumor verification system.

We used sites such as *Wikipedia*, *Snopes.com*, and *FactCheck.org* that aggregate and cite trustworthy external sources (major governmental or news organization) for verification of rumors. A rumor was annotated as true or false if at least three trustworthy sources confirmed it as such. The earliest confirmation time was taken as the trusted verification time of that rumor. From this point on in the document unless otherwise stated, a rumor refers to the tweets that have propagated over Twitter until the trusted verification time. So when we talk about the duration of a rumor we mean the time from its very first tweet to the trusted verification time of that rumor.

The count distribution of the 209 rumors can be seen in Figure 4-3. It was made certain that all of the rumors have at least 1000 tweets. Any rumor that was identified with less than 1000 tweets was discarded. Table 4.2 shows the rounded average number of tweets for each class of rumors, for each event. Note that false rumors on average have more tweets than true rumors. This is mostly due to the fact that false rumors generally take longer to be verified by trusted sources, compared to true rumors (we will explore this fact shortly).

Source/Event	**All Rumors**	**False Rumors**	**True Rumors**
2013 Boston Marathon Bombings	9334	11002	4887
2014 Ebola Pandemic	2835	3136	2467
2014 Ferguson Unrest	3011	3274	2635
Snopes.com & Factcheck.org	2170	2421	1912
All	3056	3782	2203

Table 4.2: The rounded average number of tweets for each class of rumors, for each event.

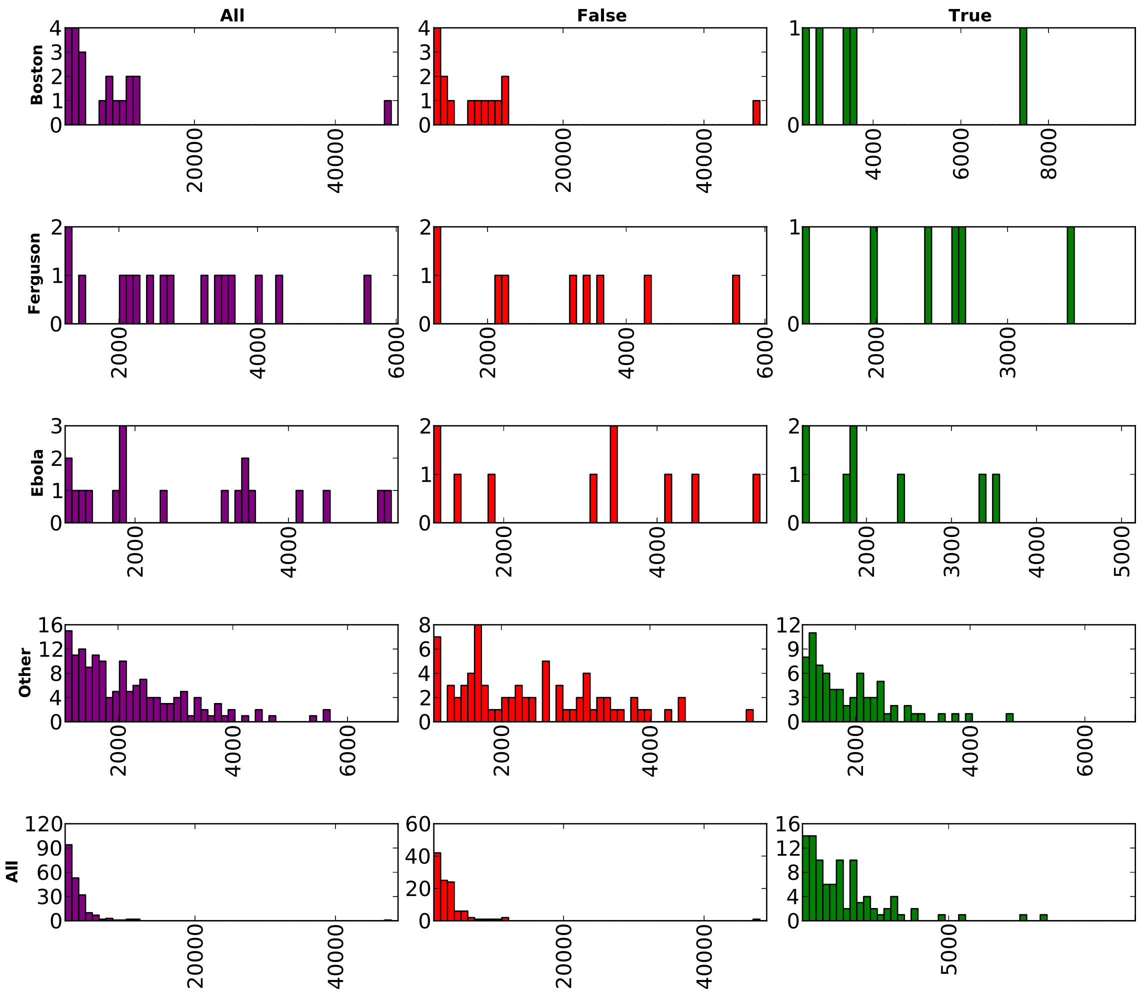


Figure 4-3: The count distribution of the 209 rumors used for training and evaluating *Rumor Gauge*. The x-axis corresponds to the number of tweets (binned) and the y-axis is the number of rumors that fall in the bins. Note that none of the rumors have less than 1000 tweets.

The duration distribution of the 209 rumors can be seen in Figure 4-4. Table 4.3 shows the average duration of rumors in hours for each class of rumors, for each event. Note that false rumors on average are longer than true rumors. One reason for this could be that proving a negative (i.e. verifying a false rumor) is a much harder and more time consuming task than proving a positive (i.e. verifying a true rumor).

Source/Event	**All Rumors**	**False Rumors**	**True Rumors**
2013 Boston Marathon Bombings	17.3	19.7	10.9
2014 Ebola Pandemic	87.1	131.8	32.4
2014 Ferguson Unrest	30.5	28.1	33.9
Snopes.com & Factcheck.org	53.9	71.6	30.0
All	51.3	66.7	29.2

Table 4.3: The average duration of rumors (in hours) for each class of rumors, for each event.

4.3 Features

A rumor can be described as a temporal communication network, where each node corresponds a communicating user, the edges correspond to communication between nodes and the temporal aspect captures the propagation of messages through the network. The intuition is that there is measurable differences between the temporal communication network corresponding to false and true rumors. In order to capture these difference, we need to identify characteristics of rumors. It makes sense that these characteristics would be related to either the nodes (i.e. users) in the network, the edges (i.e. messages) in the network or the temporal behaviour of the network (i.e. propagation).

Using this insight, we identified salient characteristics of rumors by examining three aspects of diffusion: linguistics, the users involved, and the temporal propagation dynamics. Using insights gained from related work in the related fields of meme-tracking [58, 81], diffusion and virality in social networks [62, 30, 35], measuring influence in networks [4, 110, 3], and information credibility estimation[10, 53], we composed a list of interesting features for each of the three categories. Overall, we studied 15 linguistic, 12 user-based

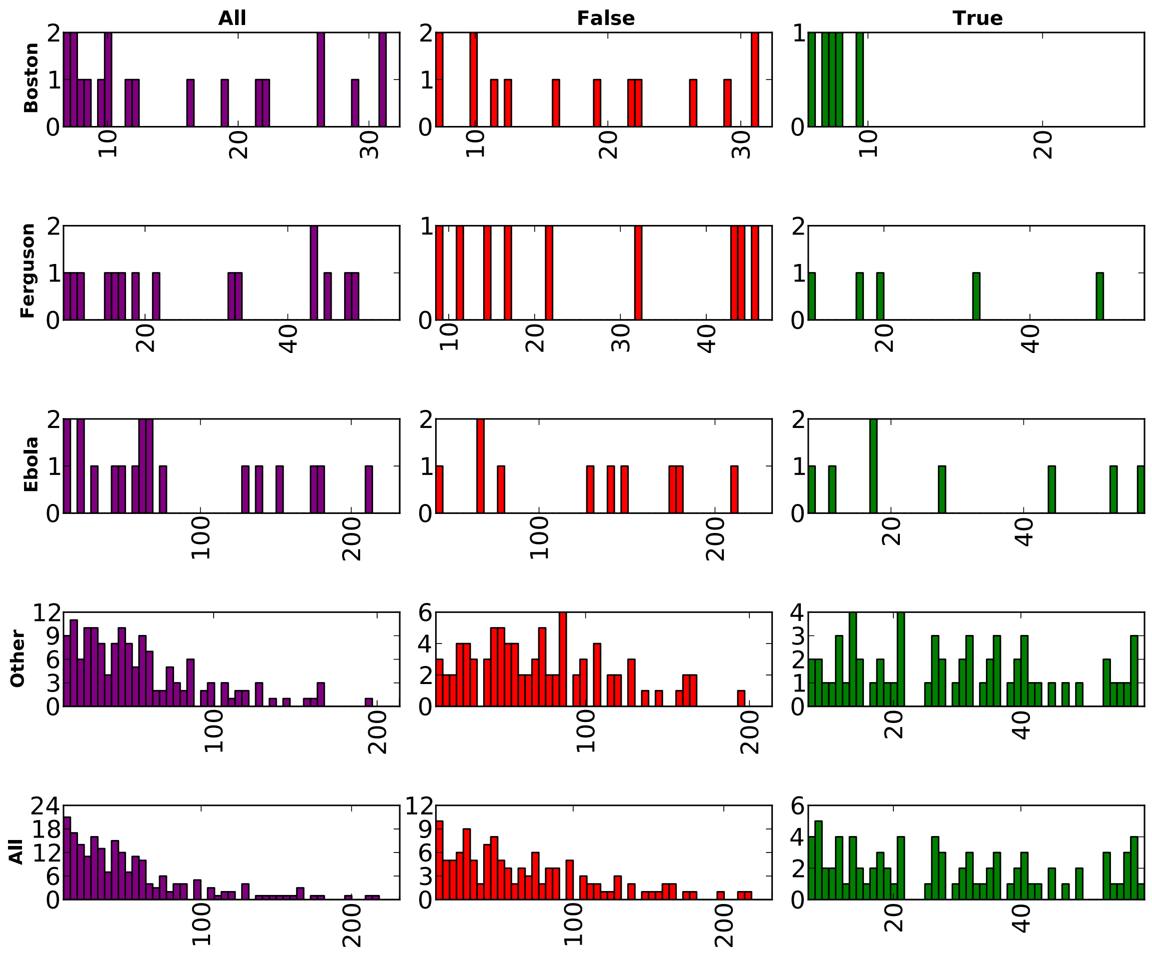


Figure 4-4: The duration distribution of the 209 rumors used for training and evaluating *Rumor Gauge*. The x-axis corresponds to the duration in hours (binned) and the y-axis is the number of rumors that fall in the bins. Note that false rumors have on average a longer duration than true rumors.

and 10 propagation features, for a total of 37 features. The contribution of each of these feature to the prediction of veracity was studied and not all were found to be significant. To measure the contribution of each feature, we rank them by the probability of the *Wald chisquare* test [36]. The null hypothesis is that there is no significant association between a feature and the outcome after taking into account the other features in the model. Small p-values indicate statistical significance, meaning that the null hypothesis should be rejected, which suggests that there is non-zero association for that feature. After removing all features that did not significantly contribute to the outcome of our models, we were left with 4 linguistic, 6 user-based and 7 propagation features, for a total of 17 features. The contributions of each feature will be explored in the "Evaluation" section of this chapter. Below, each of the 17 features from the three categories will be explained in detail.

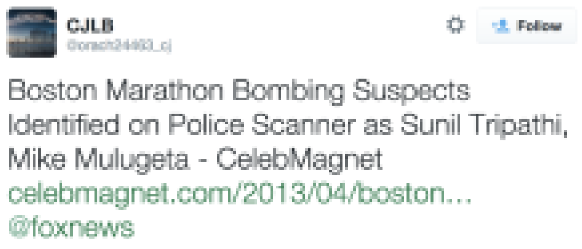
4.3.1 Linguistic

The linguistic features capture the characteristics of the text of the tweets in a rumor. A total of 4 linguistic features were found to significantly contribute to the outcome of our models. In the descending order of contribution these features are: ratio of tweet containing negations, average formality & sophistication of the tweets, ratio of tweets containing opinion & insight, and ratio of inferring & tentative tweets. We will now describe each of these features in detail.

Ratio of Tweets Containing Negation

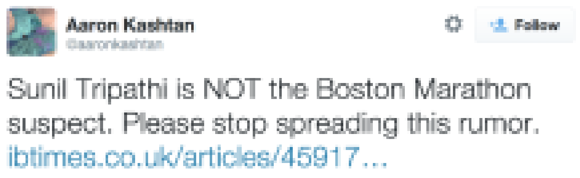
This feature measures the ratio of assertions containing negation over the total number of assertions in a rumor. Figure 4-5 shows two example tweets from the same rumor, both containing assertions. The tweet shown in Figure 4-5b however, contains a negation while the tweet shown in Figure 4-5a does not.

Equation (4.1) shows how the negation ratio value is calculated. In that equation, NR corresponds to the negation ratio, R to the rumor, $\sum^R A$ to the total number of assertions in the rumor R, and $\sum^R N(A)$ to the total number of assertion in the rumor R that contain


CJLB
@orach24493_cj

Boston Marathon Bombing Suspects Identified on Police Scanner as Sunil Tripathi, Mike Mulugeta - CelebMagnet celebmagnet.com/2013/04/boston… @foxnews

(a) An assertion not containing negation.


Aaron Kashtan
@aaronkashtan

Sunil Tripathi is NOT the Boston Marathon suspect. Please stop spreading this rumor. ibtimes.co.uk/articles/45917…

(b) An assertion containing negation.

Figure 4-5: Two example tweets from the same rumor, both containing assertions. Tweet (b) however contains a negation, while tweet (a) does not.

negation.

$$NR(R) = \frac{\sum^{R} N(A)}{\sum^{R} A} \tag{4.1}$$

The function $N(A)$ uses a negation detector to identify negations in assertions. The negation detector uses the Stanford NLP parser [16] to generate typed dependencies of tweets. For example, the sentence, *"Sunil Tripathi is NOT the Boston Marathon suspect."* from the tweet in Figure 4-5b has the following typed dependencies:

nn(Tripathi-2, Sunil-1)
nsubj(suspect-8, Tripathi-2)
cop(suspect-8, is-3)
neg(suspect-8, NOT-4)
det(suspect-8, the-5)
nn(suspect-8, Boston-6)
nn(suspect-8, Marathon-7)
root(ROOT-0, suspect-8)

From the typed dependencies we can see that the sentence contains a negation (shown in bold red) and that it is applied to the noun *suspect*.

In addition to detecting negations using syntactic analysis, a more nuanced approach to finding negations would be to use relational semantic databases like WordNet[6] [65] and ConceptNet[7] [59] to identify antonyms of terms in tweets. For example, the syntactic

[6]http://wordnet.princeton.edu
[7]http://conceptnet5.media.mit.edu/

approach for identifying negations can not correctly detect that *innocent* is, from a semantic standpoint, a negation of *guilty*. This is an area that can be explored further in future extensions to this system.

Average Formality & Sophistication of Tweets

This feature measures the sophistication and formality (or rather the informality) of tweets in a rumor. There are five indicators of formality & sophistication of a tweet, those are:

- *Vulgarity*: The presence of vulgar words in the tweet.
- *Abbreviations*: The presence of abbreviations (such as *b4* for *before*, *jk* for *just kidding* and *irl* for *in real life*) in the tweet.
- *Emoticons*: The presence of emoticons in the tweet.
- *Average word complexity*: Average length of words in the tweet.
- *Sentence complexity*: The grammatical complexity of the tweet.

The first three factors estimate the formality (or informality) of a tweet, while the last two factors estimate the sophistication. Each tweet is checked against collections of vulgar words, abbreviations and emoticons. These collections were assembled using online dictionaries[8][9][10] (and in the case of abbreviations, also Crystal's book on language used on the internet [18]). There are a total of 349 vulgar words, 362 emoticons, and 944 abbreviations. You can see example tweets containing vulgarity and emoticons in Figure 3-5 and an example of a tweet containing abbreviations in Figure 3-8.

The average word complexity of a tweet is estimated by counting the number of characters in each word in the tweet and dividing by the total number of words in that tweet. Equation (4.2) shows how this is done. Here, $WC(t)$ refers to the word complexity score

[8]http://www.noswearing.com/dictionary
[9]http://pc.net/emoticons/
[10]http://www.netlingo.com/category/acronyms.php

of tweet t, $NW(t)$ counts the number of words in the tweet t, and $NC(w)$ counts the number of characters in the word w. For example, the tweet, *"There is another bomb at Harvard"* has 6 words, containing $5, 2, 7, 4, 2, 7$ characters respectively. The average word complexity of the tweet is therefore: $\frac{5+2+7+4+2+7}{6} = 4.5$.

$$WC(t) = \frac{\sum_{i=0}^{NW(t)} NC(w_i)}{NW(t)} \tag{4.2}$$

The sentence complexity of a tweet is estimated by the depth of its dependency parse tree. We used Kong et al.'s [48] Twitter dependency parser for English to generate dependency trees. A sample dependency tree can be seen in Figure 3-9 in the last chapter. In that example, the tweet,*"our hearts go out to those effected by the marathon bombings"*, has a depth of 5.

Ratio of Tweets Containing Opinion & Insight

We collected a list of *opinion* and *insight* words from the Linguistic Inquiry and Word Count (LIWC)[11]. The LIWC dictionary provides psychologically meaningful categories for the words in its collection [76]. One of these categories is *opinion & insight* words. This includes words like, *know*, *consider*, *think*, etc. Each tweet is checked against the *opinion & insight* words from LIWC.

Ratio of Inferring & Tentative Tweets

Another category in the LIWC is the *inferring tentative* words. This includes words like, *perhaps*, *guess*, *may be*, etc. Each tweet is checked against the *inferring & tentative* words from LIWC.

[11]http://www.liwc.net/descriptiontable1.php

4.3.2 User Identities

The user features capture the characteristics of the users involved in spreading a rumor. A total of 6 user features were found to significantly contribute to the outcome of our models. In the descending order of contribution these features are: controversiality, originality, credibility, influence, role, and engagement. We will now describe each of these features in detail.

Controversiality

The controversiality of a user is measured by analysing the replies to the user's tweets. The replies to the last 1000 tweets of a user are collected. These replies are then run through a sentiment classifier which classifies them as either positive, negative, or neutral. There are numerous off-the-shelf Twitter sentiment analyser, however, for the purposes of this thesis we developed an state-of-the-art Twitter sentiment classifier [103]. This classifier is explained in detail in Appendix A.

The number of all positive and negative replies are counted and used to calculate a controversiality score for the user. The formula for this is shown in equation (4.3). In that equation, p is the total number of positive replies and n is the total number of negative replies.

$$Controversiality = (p+n)^{min(\frac{p}{n},\frac{n}{p})} \tag{4.3}$$

Figure 4-6 illustrates how the controversiality score is distributed for different number of positive and negative replies. Note that this is an example illustration so the number of positive and negative replies don't exceed 100, which is not necessarily the case for the users in our dataset. As you can see in the figure, controversiality of a user is dependent on two factors: the number of replies to the user, and the ratio of replies with different sentiments. The higher the number and the closer to 1.0 the ratio, the higher the controversiality score. This makes intuitive sense, since a controversial user would be someone whose tweets generate a lot of replies, with around half agreeing with (liking) and half disagreeing with (disliking) the user's tweets.

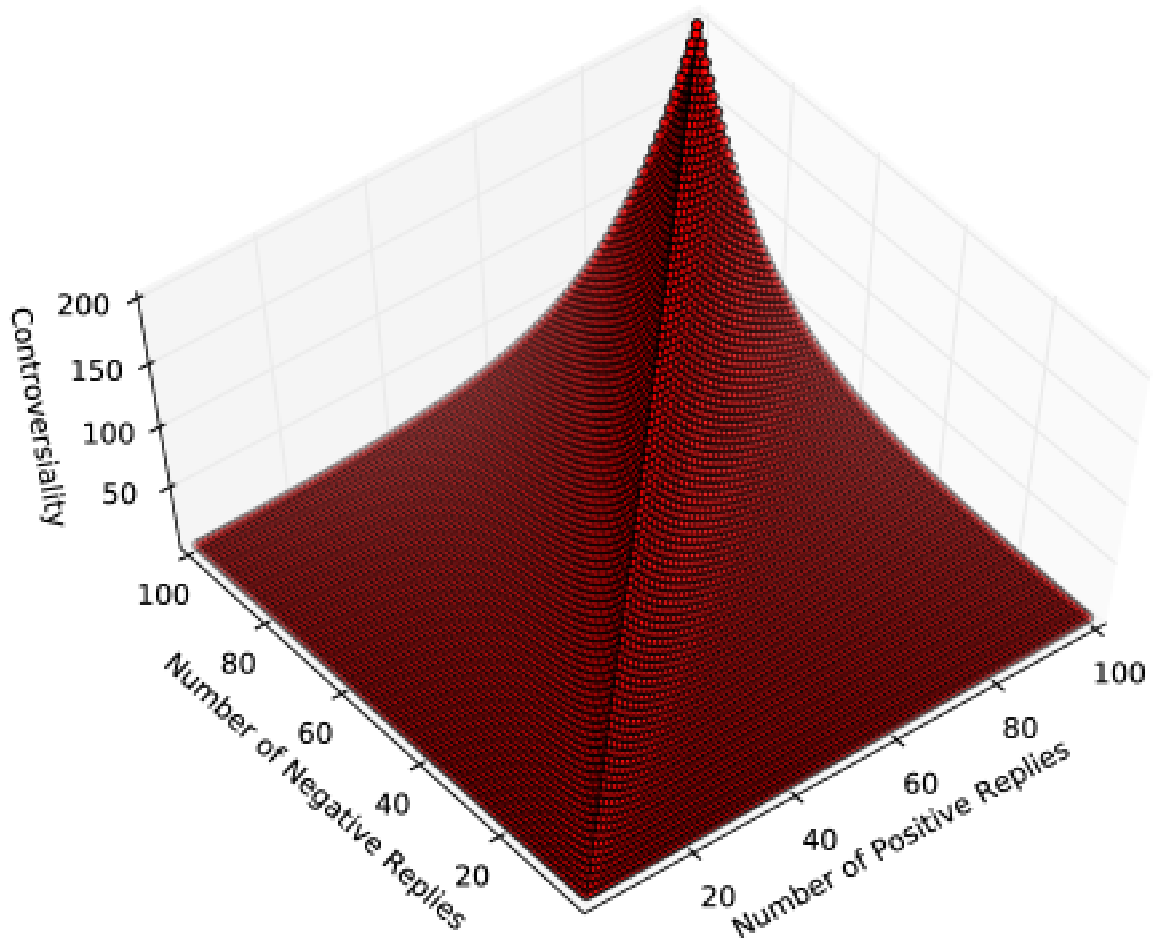


Figure 4-6: Controversiality values for different number of positive and negative replies. This is an example to illustrate the distribution of controversiality scores as a function of positive and negative replies. Note that this is an example illustration so the number of positive and negative replies don't exceed 100, which is not necessarily the case for the users in our dataset.

Originality

Originality is a measure of how original a user's communications are on Twitter. As shown in Equation (4.4), originality is calculated by the ratio of the number of original tweets a user has produced, and the number of times the user's has just retweet someone else's original tweet. The greater this ratio, the more inventive and original a user is. Conversely, a lower ratio indicates an unoriginal and parrot like behaviour by the user (i.e. just repeating what others say).

$$Originality = \frac{\#Tweets}{\#Retweets} \tag{4.4}$$

Credibility

The credibility of a user is measured by whether the user's account has been verified by Twitter or not. This feature can either have the value 1 or 0.

$$Credibility = \begin{cases} 1 & \text{if verified} \\ 0 & \text{otherwise} \end{cases}$$

(4.5)

Influence

Influence is measured simply by the number of followers of a user. Presumably, the more followers a user has, the more influential he or she is.

$$Influence = \#Followers \tag{4.6}$$

Role

Role measures the ratio of followers and followees of a user, as shown in Equation (4.7). A user with a high score is a broadcaster; conversely, a user with a low score is a receiver.

$$Role = \frac{\#Followers}{\#Followees} \tag{4.7}$$

Engagement

Engagement measures how active a user has been on Twitter ever since joining. Equation (4.8) shows how this is calculated.

$$Engagement = \frac{\#Tweets + \#Retweets + \#Replies + \#Favourites}{AccountAge} \tag{4.8}$$

4.3.3 Propagation Dynamics

The propagation features capture the temporal diffusion dynamics of a rumor. A total of 7 propagation features were found to significantly contribute to the outcome of our models. In the descending order of contribution these features are: fraction of low-to-high diffusion, fraction of nodes in largest connected component (LCC), average depth to breadth ratio, ratio of new users, ratio of original tweets, fraction of tweets containing outside links, and the fraction of isolated nodes. All of these features are derived from a rumor's diffusion graph. Before we describe these features in detail, we need to explain how to diffusion graph was created.

Time-Inferred Diffusion

A rather straight-forward way to capture the diffusion of tweets is through analysing the retweet path of those tweet. Since each tweet and retweet is labelled with a time-stamp, one can track the temporal diffusion of messages on Twitter. However, the Twitter API does not provide the true retweet path of a tweet. Figure 4-7 shows the retweet tree that the Twitter API provides. As you can see, all retweets point to the original tweet. This does not capture the true retweet tree since in many cases a user retweets another user's retweet, and not the original tweet. But as you can see in Figure 4-7, all credit is given to the user that tweeted the original tweet, no matter who retweeted who.

Fortunately, we can infer the true retweet path of a tweet by using the Twitter's follower

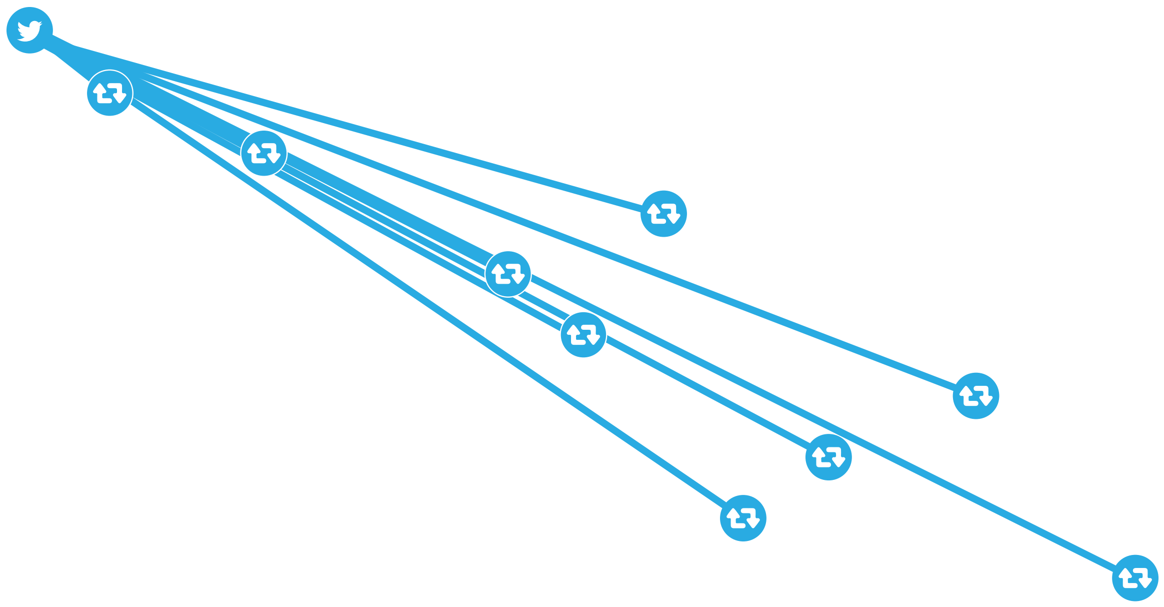


Figure 4-7: The retweet tree of a tweet as provided by the Twitter API. Each node represents a user and the x-axis is time. The bird on the top right represents an original tweet and the arrows represent retweets.

graph. Figure 4-8 shows how this is achieved. The top panel in the figure shows the retweet path provided by the Twitter's API. The middle panel shows that the bottom user is a follower of the middle user but not that of the top user (the user who tweeted the original tweet). Finally, the third panel shows that using this information, and the fact that the bottom user retweeted after the middle user, it can be inferred that the bottom person retweeted must have retweeted the middle person and not the top person. This method of reconstructing the true retweet graph is called time-inferred diffusion as is motivated by work by Geol et al. [35].

Using this method, we can convert our hypothetical retweet tree shown in Figure 4-7 to a more accurate representation of the true retweet tree, shown in Figure 4-9. Note that a rumor is composed of many retweet trees. Figure 4-10 is a simplified illustration of what the diffusion of a rumor might look like. Note that the diffusion is composed of several time-inferred diffusion trees. Using these time-inferred diffusion trees, we can trace a rumor's diffusion through Twitter and extract informative features about the nature of the rumor's diffusion. Next, we will explain these features in detail.

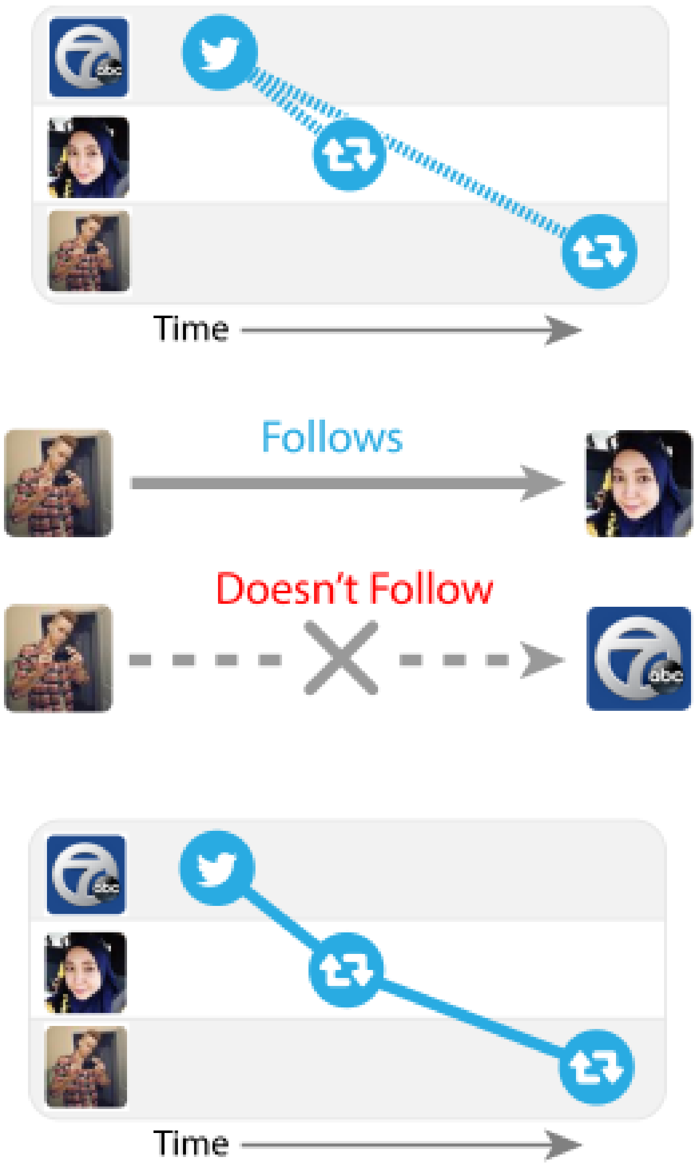


Figure 4-8: Using Twitter's follower graph to infer the correct retweet path of a tweet. The top panel shows the retweet path provided by the Twitter's API. The middle panel shows that the bottom user is a follower of the middle user but not that of the top user (the user who tweeted the original tweet). The third panel shows that using this information, and the fact that the bottom user retweeted after the middle user, we can infer that the bottom person retweeted the middle person and not the top person.

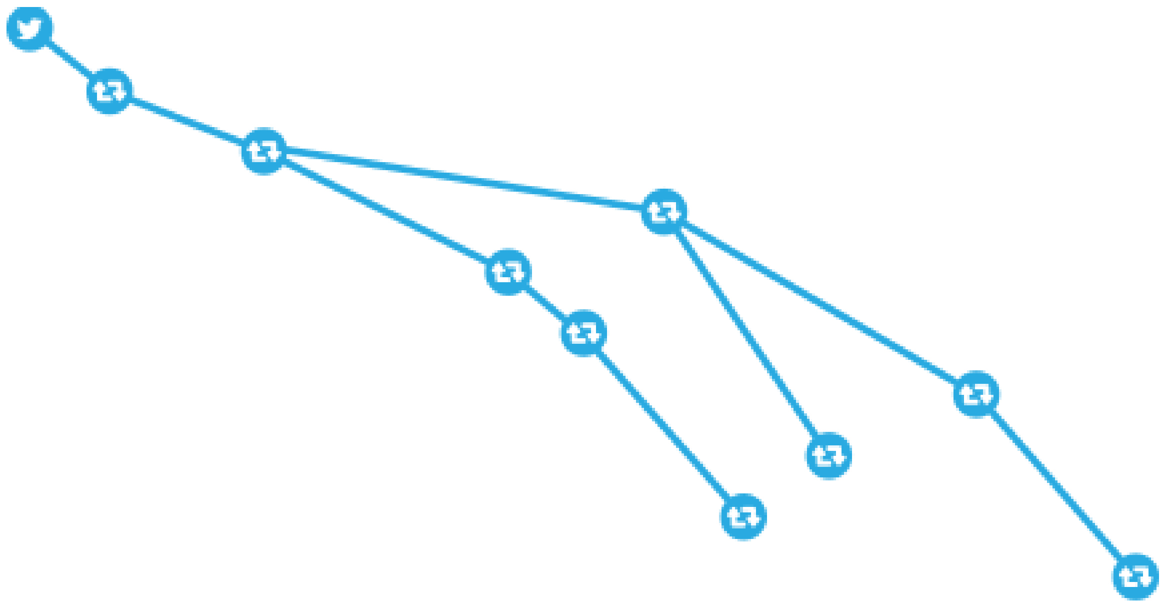


Figure 4-9: The time-inferred retweet tree of the tree shown in Figure 4-7. Same as in that figure, each node represents a user and the x-axis is time. The bird on the top right represents an original tweet and the arrows represent retweets.

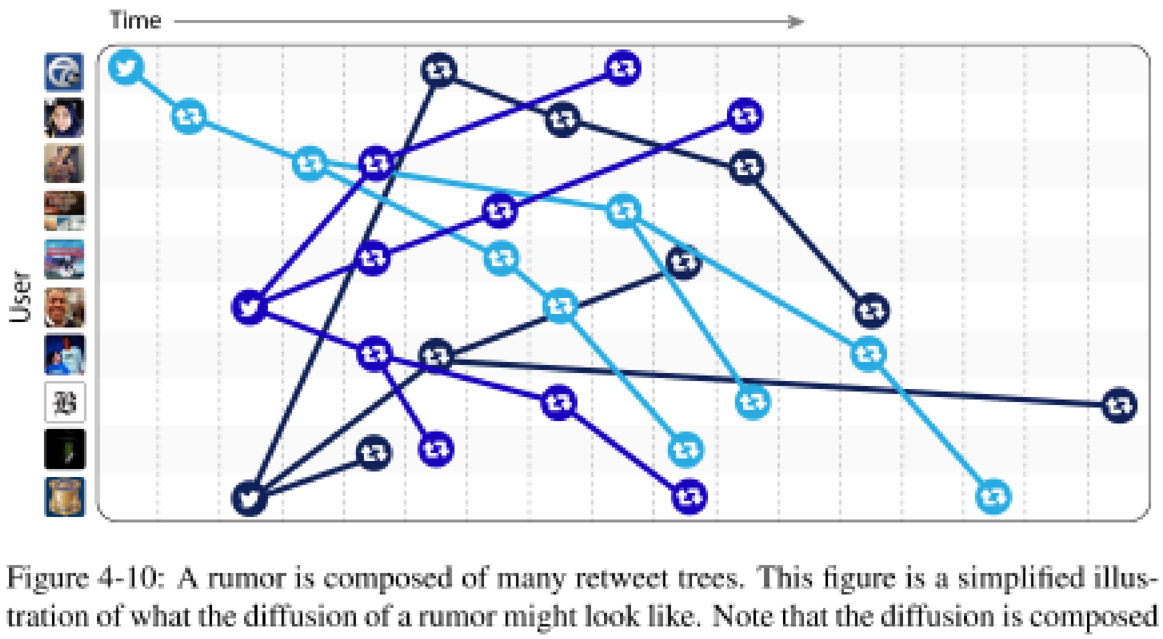


Figure 4-10: A rumor is composed of many retweet trees. This figure is a simplified illustration of what the diffusion of a rumor might look like. Note that the diffusion is composed of several time-inferred diffusion trees.

Fraction of Low-to-High Diffusion

Each edge in the propagation graph of a rumor (such as the one shown in Figure 4-9) corresponds to a diffusion event. Each diffusion event takes place between two nodes. Diffusion events are directional (in the direction of time), with the information diffusing from one node to another over time. We shall call the node that pushes the information out (i.e., influences), the *sender* and the node that receives the information (i.e., the one being influenced), the *receiver*.

The *fraction of low-to-high diffusion* feature measures the fraction of diffusion events where the diffusion was from a sender with lower influence to a receiver with higher influence (see Equation (4.9)). Influence of a user corresponds to the user's number of followers (as defined in section 4.3.2 of this thesis). To illustrate this further, Figure 4-11 shows an enhanced version of the diffusion tree shown in Figure 4-9 where the size of the nodes correspond to the influence of the users. Here, we can more clearly see the concept of low-to-high diffusion. For example, the diffusion between the second and third nodes (from left).

$$\%\text{Low-High Diffusion} = \frac{\#\text{Low-high diffusions}}{\#\text{All diffusion events}} \tag{4.9}$$

Figure 4-12 shows a real example of a low-to-high diffusion from the Boston Marathon bombings. As you can see, the user on the left, with 19.8K followers, was retweeted by the person on the right, with 178K followers (roughly one order of magnitude more influential than the other user).

This feature is very informative as it captures the *eye-witness* phenomenon which is prevalent during real-world events. It also highlights the role of Twitter as a source for breaking-news through eye-witnesses on the ground. As we will explain later in this chapter, this feature is most predictive of the veracity of rumors compared to any other feature by itself.

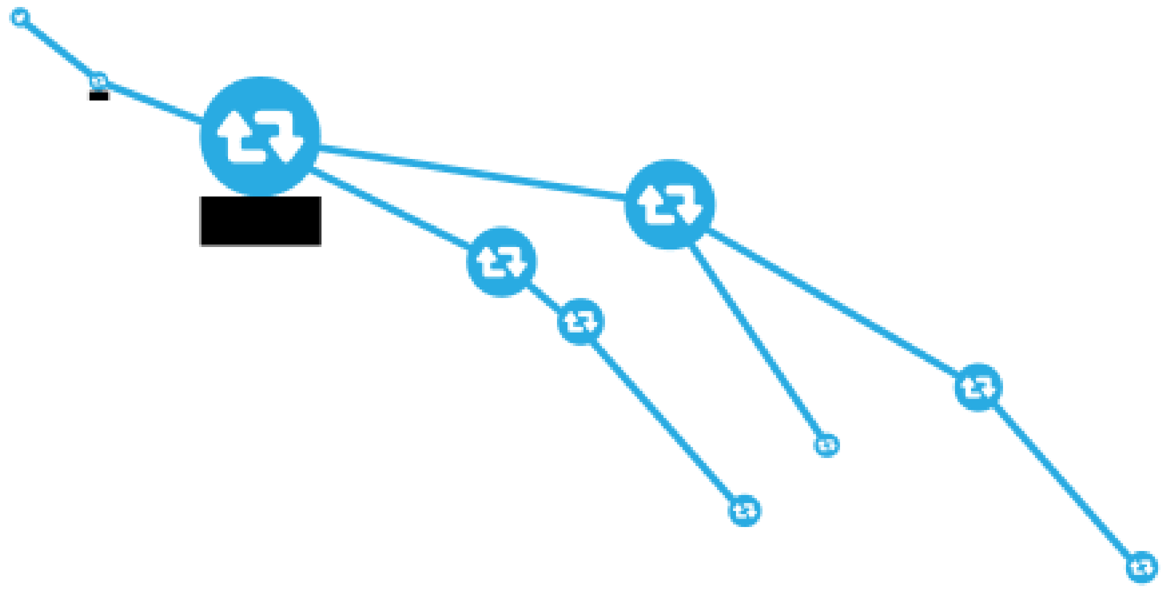


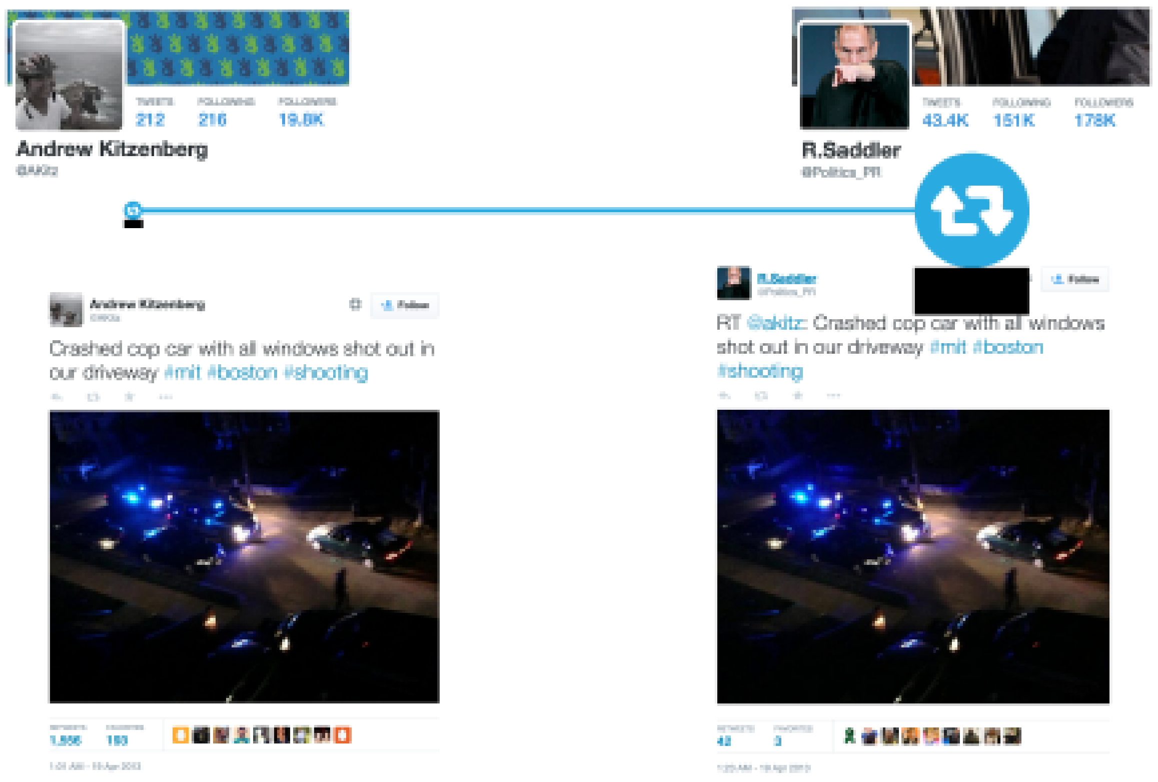
Figure 4-11: This figure illustrates an enhanced version of the diffusion tree shown in Figure 4-9 where the size of the nodes correspond to the influence of the users. Here we can more clearly see the concept of low-to-high diffusion. For example, the diffusion between the second and third nodes (from left). As with Figure 4-9, the x-axis represents time.

Fraction of Nodes in Largest Connected Component

A connected component of a graph is a subgraph where every node is reachable from any other node. The largest connected component (LCC) of a graph is the connected component with the highest number of nodes. In a Twitter diffusion graph, the LCC corresponds to an original tweet with the highest number of retweets.

The *fraction of nodes in LCC* feature measures the ratio of nodes in the LCC of a rumor's diffusion graph, over the total number of nodes in the diffusion graph (see Equation (4.10)). This feature captures the longest conversation chain of a rumor on Twitter. To illustrate this feature, figure 4-13 shows two example diffusion trees with same number of nodes (54). The tree in Figure 4-13a has a large fraction of nodes in LCC ($\frac{32}{54} = 0.6$) and the tree in Figure 4-13b has a relatively lower fraction of nodes in LCC ($\frac{9}{54} = 0.16$).

$$\%\text{Nodes in LCC} = \frac{\#\text{Nodes in LCC}}{\#\text{All Nodes}} \tag{4.10}$$



Andrew Kitzenberg
@AKitz
TWEETS 212 FOLLOWING 216 FOLLOWERS 19.8K

R.Saddler
@Politics_PR
TWEETS 43.4K FOLLOWING 151K FOLLOWERS 178K

[redacted]

Andrew Kitzenberg
@AKitz

Crashed cop car with all windows shot out in our driveway #mit #boston #shooting

RETWEETS 1,856 FAVORITES 183

R.Saddler
@Politics_PR

[redacted]

RT @akitz: Crashed cop car with all windows shot out in our driveway #mit #boston #shooting

RETWEETS 42 FAVORITES 3

Figure 4-12: This figure shows a real example of a low-to-high diffusion from the Boston Marathon bombings. As you can see, the user on the left, with 19.8K followers, was retweeted by the person on the right, with 178K followers (roughly one order of magnitude more influential than the other user).

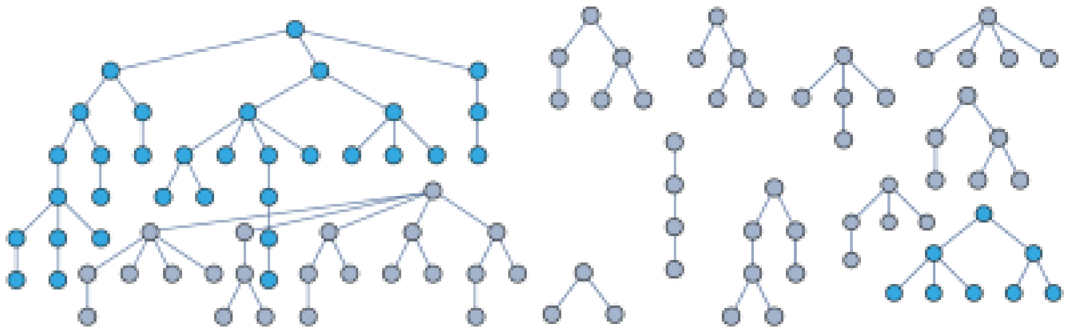

(a) A diffusion tree with a large fraction of nodes in LCC.
(b) A diffusion tree with a small fraction of nodes in LCC.

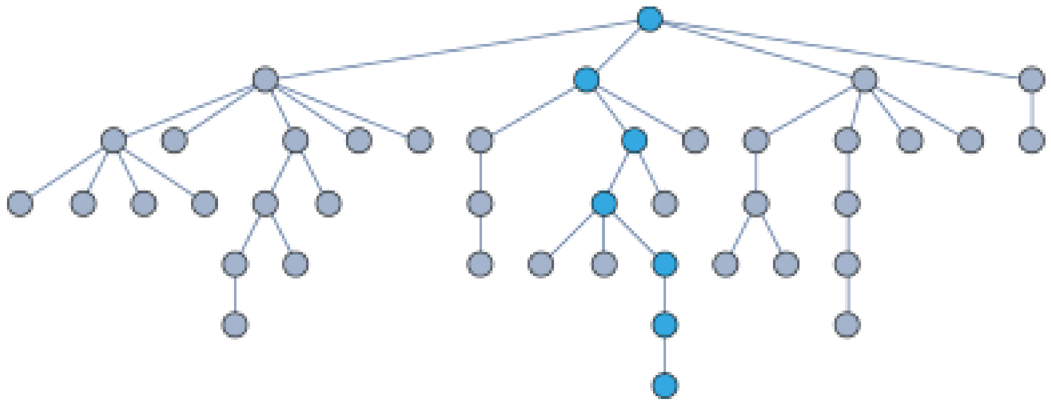
Figure 4-13: Two example diffusion trees with same number of nodes, one with a large fraction of nodes in LCC and one with a relatively lower fraction of nodes in LCC. The blue nodes represent the nodes in the LCC.

Average Depth to Breadth Ratio

The shape of a diffusion graph can reveal a lot about the nature of the diffusion. The feature, *average depth to breadth ratio* is an attempt to quantify the shape of the diffusion graphs of rumors. The depth of a diffusion tree is defined as the longest path from the root (i.e. original tweet) to the a leaf. The breadth of a tree is defined as the total number of nodes it contains. Since each rumor diffusion graph is composed of many diffusion trees (see Figure 4-10), we average the depth to breadth ratio of all the diffusion trees in a rumor. Equation (4.11) shows exactly how this feature is calculated for a rumor (here N refers to the number of diffusion trees in a rumor).

$$\text{Average \%depth-to-breadth} = \frac{\sum^{N} \frac{\text{\#Nodes in largest chain}}{\text{\#All Nodes}}}{N} \tag{4.11}$$

To illustrate this feature, figure 4-14 shows two example diffusion trees with same number of nodes (42). The tree in Figure 4-14a has a low depth to breadth ratio (i.e. is shallow) ($\frac{7}{42} = 0.16$) and the tree in Figure 4-14b has a relatively higher depth to breadth ratio (i.e. is deeper) ($\frac{12}{42} = 0.28$).


(a) A shallow diffusion tree.

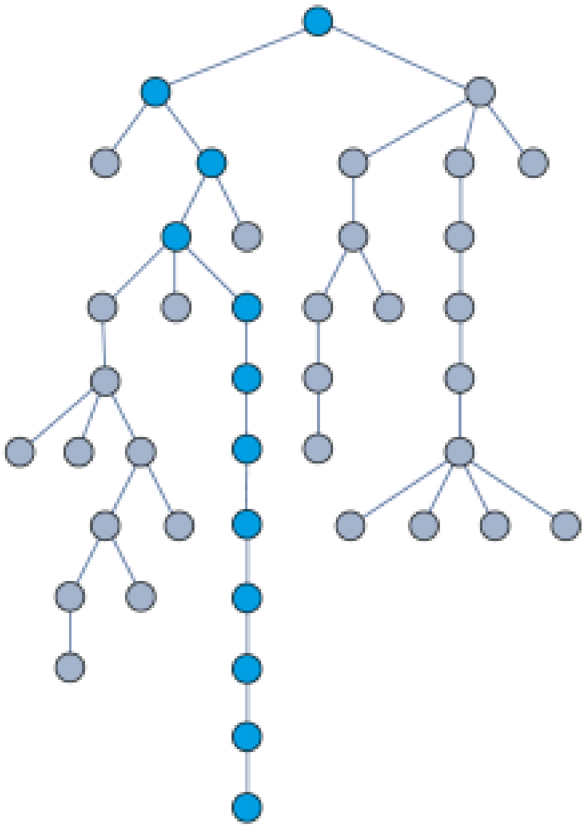

(b) A deep diffusion tree.

Figure 4-14: Two example diffusion trees with same number of nodes, one with deep chains and one with shallow chains. The blue nodes represent the nodes in the deepest chain.

Ratio of New Users

This is a measure of the diversity of users engaged in the conversation about a rumor. Recall that all of the features are temporal, meaning that each feature is calculated at every time-step (e.g., every hour), resulting in a time-series for each feature. Note that without the temporal aspect, this feature is meaningless. However, by measuring the number of new users that enter the conversation about a rumor over time, this feature becomes meaningful. Equation (4.12) shows hows this features is calculated.

$$\%\text{New Users}(t_i) = \frac{\sum \{\text{users}(t_i) \mid \text{users}(t_i) \notin \text{users}(t_0 ... t_{i-1})\}}{\sum \{\text{users}(t_i)\}} \tag{4.12}$$

Ratio of Original Tweets

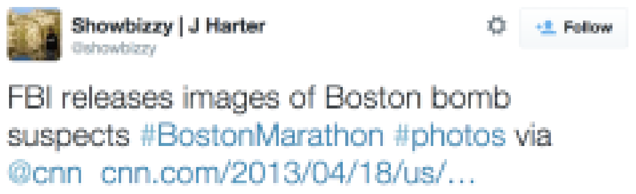
This is a simple measure of how captivating, engaging and original is the conversation about a rumor. This is measured by the ratio of new tweets and replies (i.e. not retweets) in the diffusion graph of a rumor. Equation (4.13) shows the exact formula used to calculate this feature.

$$\%\text{Original Tweets} = \frac{\#\text{Tweets} + \#\text{Replies}}{\#\text{Tweets} + \#\text{Replies} + \#\text{Retweets}} \tag{4.13}$$

Fraction of Tweets Containing Outside Links

Tweets can contain links to sources outside of Twitter. It is very common for tweets that talk about real-world emergencies and events to have links to news organizations or other social media. Figure 4-15 shows an example tweet about the Boston Marathon bombings with a link to an outside source.

When studying rumors on Twitter, it makes intuitive sense to see whether tweets that are making assertions contain links to other sources (i.e. if there are other sources corroborating the claims). Though we do not currently track the diffusion of rumors outside of Twitter, through this feature we can have a very rough approximation of the corroboration factor. Equation (4.14) shows how this feature is calculated.


Showbizzy | J Harter
@showbizzy

FBI releases images of Boston bomb suspects #BostonMarathon #photos via @cnn cnn.com/2013/04/18/us/...

Figure 4-15: A a sample tweet about the Boston Marathon bombings with a link to an outside source (CNN).

$$\%\text{Tweet-with-URL} = \frac{\#\text{Tweets containing a URL}}{\#\text{All Tweets}} \tag{4.14}$$

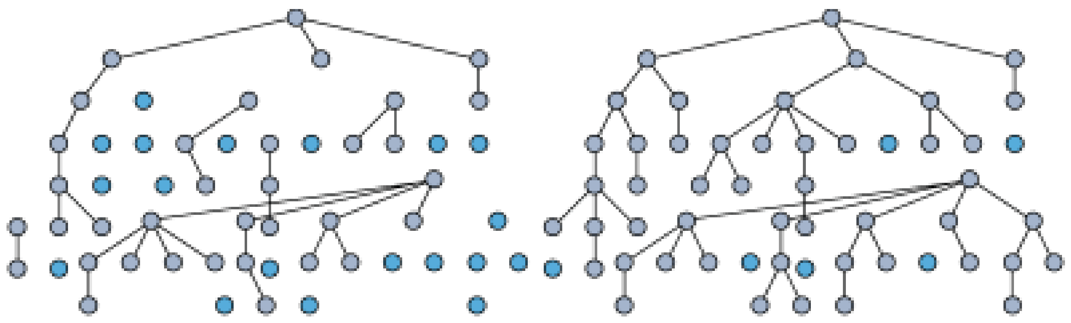
Fraction of Isolated Nodes

Not all tweets get retweeted or get a reply . According to a study by the social media analytics company *Sysomos*, about 71% tweets do not ever get a response (retweet or reply)[12]. Tweets that get no response can be indicators of a user that's uninfluential, a message that is uninteresting, or the over saturation of similar messages on Twitter. All of these factors reveal something about the nature of the tweet, and when analysed for all the tweets contained in a rumor, it can reveal something about the nature of the rumor.

A tweet that is not get any replies or retweets shows up as an isolated node (a node with degree 0, i.e., without any edges) in a rumor's diffusion graph. This feature captures the fraction of all nodes in a rumor's diffusion graph that are isolated. Equation (4.15) shows exactly how this feature is calculated. To illustrate this feature, figure 4-16 shows two example diffusion trees with same number of nodes (54). The tree in Figure 4-16a has a large fraction of isolated nodes ($\frac{19}{54} = 0.35$) and the tree in Figure 4-16b has a relatively lower fraction of isolated nodes ($\frac{6}{54} = 0.11$).

$$\%\text{Isolated Nodes} = \frac{\sum \{nodes \mid degree(nodes) = 0\}}{\sum \{nodes\}} \tag{4.15}$$

[12]http://sysomos.com/insidetwitter/engagement/



(a) A diffusion tree with a large fraction of isolated nodes. (b) A diffusion tree with a small fraction of isolated nodes.

Figure 4-16: Two example diffusion trees with same number of nodes, one with a large fraction of isolated nodes and one with a relatively lower fraction of isolated nodes. The blue nodes represent the isolated nodes.

4.4 Models

Once again, recall that all of the features are temporal, meaning that each feature is calculated at every time-step, resulting in a time-series for each feature. The temporal aspect of the features is very important for two main reasons. First, with a few notable exceptions, it is rarely the case that the magnitude (i.e., the values without the temporal dynamics) of any of these features are predictive of the veracity of rumors. It is mainly the temporal dynamics of these features that can signal the falsehood or truthfulness of rumors. Second, different rumors have vastly different footprints on Twitter. Some contain tens of thousands of tweets, retweets and replies while others might contain only a thousand responses (see Figure 4-3 for the size distribution of the 209 rumors in our dataset). The temporal dynamics of these features are mostly invariant to the size of the rumors, allowing our models to generalize to events and rumors of various sizes.

Given that our features are temporal, we selected models best suited for dealing with temporal features. Also, the models had to be invariant to time. This is crucial since in addition to being of different sizes, rumors have varying durations, ranging from a few hours to days (see Figure 4-4 for the duration distribution of the 209 rumors in our dataset). Inspired by the field of speech recognition where similar constraints apply (e.g., the speed at which people talk can be different, but that should not affect the outcome of the speech recognition), we selected two models: Dynamic Time Warping (DTW), and Hidden Mark Model (HMM). Next, we explain how these models work and how they were used in detail.

4.4.1 Dynamic Time Warping

Originally developed for speech recognition, dynamic time warping (DTW) is a time-series alignment algorithm. DTW can find an optimal non-linear alignment between two time-series [87]. The non-linear alignment makes this method time-invariant, allowing it to deal with time deformations and different speeds associated with time-series. In other words, it can match time-series that are similar in shape but out of phase, or stretched or elongated in the time axis. Figure 4-17 shows a sketch of DTW applied to two curves. The non-linear alignment aspect of DTW can be clearly seen in the figure.

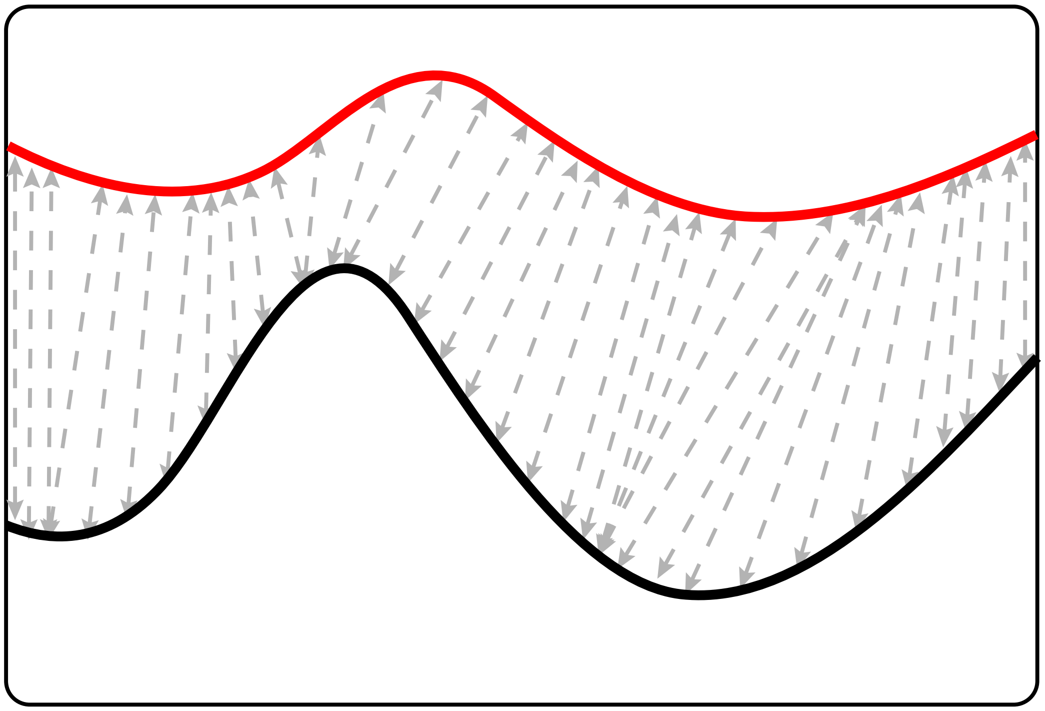


Figure 4-17: An example sketch of dynamic time warping applied to two curves. The non-linear alignment aspect of DTW can clearly be seen.

The input to the DTW algorithm is two time-series, and the algorithm, returns the minimum distance between the series (amongst other things). We set the cost measure used by the DTW to be the standard *Manhattan Distance* (i.e. *L1 norm*). The Manhattan distance between two points is the sum of the absolute differences of their Cartesian coordinates (see Equation (4.16) below).

$$||x||_1 = \sum_{i=1}^{n} |x_i| \tag{4.16}$$

We used DTW to measure the similarity between rumors. Since each rumor is composed of 17 features, we need to average over the similarity between all 17 time-series in the rumors. Equation (4.17) below shows how this is done. Here, $S(R_c, R_i)$ is the similarity between two rumors, R_i which is the input rumor, and R_c which is an annotated rumor of class c (either false or true). The similarity is the average of one minus the normalized distance, as measured by DTW, between each of the 17 time-series that make up the rumors.

$$S(R_c, R_i) = \frac{\sum_{f=1}^{17}(1 - DTW(R_c^f, R_i^f))}{17} \tag{4.17}$$

For the purposes of training a classifier, DTW can be used as the distance measure for a *Nearest Neighbors (NN)* classifier [25]. Specifically, we used $S(R_c, R_i)$ shown in Equation (4.17) as the distance measure for a NN classifier. The similarity between an input rumor and a class of rumors is shown in Equation (4.18). Here, $S(R_i, C)$ is the similarity between an input rumor, R_i and the class of rumors, C (note that there are two classes for C: false and true). N is the equivalent of the number neighbours, K in a K-NN classifier. We set N to be 10.

$$S(R_i, C) = \sum_{j=1}^{N} \frac{S(R_j^c, R_i)}{N} \tag{4.18}$$

Using the $S(R_i, C)$ function, the veracity of an input rumor, R_i can be calculated as shown in Equation (4.19) below.

$$Veracity = \begin{cases} True & \text{if } S(R_i, True) > S(R_i, False) \\ False & \text{if } S(R_i, True) < S(R_i, False) \\ Indecisive & Otherwise \end{cases} \tag{4.19}$$

Finally, since the output of the similarity function, $S(R_i, C)$, is normalized, we can approximate the confidence of the prediction using Equation (4.20) below. Note that this is not a probabilistic confidence score and is not statistically rigorous, it is meant as an approximation of the confidence of the prediction. The Hidden Markov Model explained in the next section provides a much more rigours estimation of confidence.

$$Confidence = \mid S_{true} - S_{false} \mid \tag{4.20}$$

4.4.2 Hidden Markov Model

DTW is limited by the fact that it assumes all the time-series are independent of each other. However, this is an assumption that does not hold for our features, since many of our features are in fact coupled. Moreover, the DTW model shown in the last section assigns equal weight to all 17 features. This is also an incorrect assumption since certain features are much more correlated with the veracity of rumors (this is explored in detail in the ”Evaluation” section of this chapter). Hidden Markov Models address both of these shortcomings in DTW.

Hidden Markov Models are generative and probabilistic. In an HMM, a sequence of observable variable, X, is generated by a sequence of internal hidden states, Z, which can not be directly observed. In an HMM, it is assumed that the transitions between the hidden states have the form of a Markov chain [79]. An HMM can be fully determined by three parameters. A start probability vector, Π, a transition probability matrix, A, and the emission probability of the observable variable (e.g., Gaussian, Poisson, etc), Θ_i, which is conditioned on the current hidden state (i). HMM also allows for multiple observable variables by using multivariate emission probability distributions (e.g., multivariate Gaussian distribution).

Generally speaking, there are three problems for HMMs [79]:

1. Given the model parameters and observed data, estimate the optimal sequence of hidden states.

2. Given the model parameters and observed data, calculate the likelihood of the data.

3. Given just the observed data, estimate the model parameters.

For our problem, we start with multivariate observations for false and true rumors. We want to use HMMs to model the temporal dynamics of these multivariate observations. The hidden states capture the different event that drive the dynamics of the time-series (e.g., sudden influx trustworthy which will correspond to an increase in influential and verified users). We experimented with different number of states and found that 20 states was sufficient for our purposes.

We trained two HMMs, one on observed data from false rumors and one on observed data on true rumors. The first problem we needed to solve was item number 3: given the observed data, estimate the model parameters. This was done using the standard, iterative *Expectation-Maximization (EM)* algorithm, known as the *Baum-Welch* algorithm [79]. The emission probabilities,Θ_i, were set to be multivariate Gaussian. We used a full covariance matrix (as opposed to a diagonal matrix), as to allow for correlation between different features. A diagonal matrix on the other hand would have treated each of the features as independent variables.

When we want to predict the veracity of a new rumor, we need to solve item number 2: given the model parameters and observed data, calculate the likelihood of the data. We calculate the likelihood of the new observed data for the false and true HMMs. This is achieved by using the standard *Forward-Backward* algorithm [79]. We then compare the likelihood of the new data under the false HMM and true HMM. The veracity is predicted to be true if the likelihood under the true HMM is higher and vice-versa (see Equation (4.21) below).

$$Veracity = \begin{cases} True & \text{if } P(X_{new} \mid \Pi_t, A_t, \Theta_t) > P(X_{new} \mid \Pi_f, A_f, \Theta_f) \\ False & \text{if } P(X_{new} \mid \Pi_t, A_t, \Theta_t) < P(X_{new} \mid \Pi_f, A_f, \Theta_f) \\ Indecisive & Otherwise \end{cases} \tag{4.21}$$

The confidence of the prediction can be estimated by dividing the greater likelihood probability by the smaller likelihood probability. Since likelihoods are usually extremely small, we move to logarithmic space to avoid possible floating point inaccuracies. In logarithmic space, the confidence is estimated by the absolute value of two log-likelihoods subtracted from each other, as shown in Equation (4.22) below.

$$Confidence = \mid \log(P(X_{new} \mid \Pi_t, A_t, \Theta_t) - \log(P(X_{new} \mid \Pi_f, A_f, \Theta_f) \mid \tag{4.22}$$

In the next section, we will evaluate the performance of our models and features.

4.5 Evaluation

The evaluation paradigm used for *Rumor Gauge* is shown in 4-18. The figure shows a sample rumor diffusion in Twitter. The purple vertical lines correspond to the times at which rumor was trusted verified. Recall that trusted verification is defined to be verification by trusted channels (trustworthy major governmental or news organizations). It was made certain that all of the 209 rumors in our dataset were verified by at least three sources. We used *Wikipedia*, *Snopes.com*, and *FactCheck.org* (websites that aggregate external sources) to retrieve the trusted verification sources. All trusted verifications had timestamps which we used to place the verifications in the context of the rumors' diffusion.

Given this framing, there are four main goals in the evaluation of *Rumor Gauge*:

1. Measure the accuracy at which our model can predict the veracity of a rumor before the first trusted verification (i.e., using the pre-verification signal).
2. Measure the contribution of each of the linguistic, user, and propagation categories as a whole.
3. Measure the contributions of each of the 17 features.
4. Measure the accuracy of our model as a function of latency (i.e., time elapsed since the beginning of a rumor).

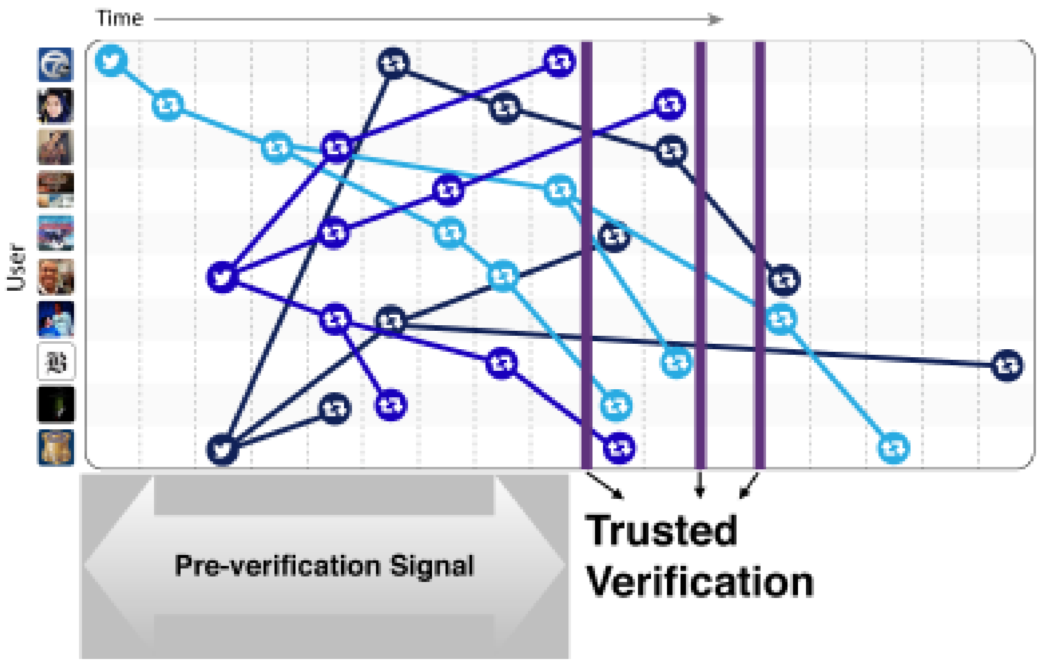


Figure 4-18: The evaluation paradigm for rumor verification. This figure illustrates a sample rumor diffusion in Twitter. The purple vertical lines correspond to the times at which the rumor was verified by trusted sources. We want to evaluate how accurately our models can predict the veracity of a rumor before the first trusted verification (i.e., using the pre-verification signal).

Note that given the relative sparsity of our dataset (209 annotated rumors), standard cross-fold validation is not the optimal way to evaluate our models. We utilized an evaluation method called jackknife evaluation [27]. 4-19 shows an example of how jackknife evaluation is achieved. At each step, a data point is held-out for testing while the other data points (208 in the case of the rumors dataset) are used to train our models. The model is tested on the held-out data point; this process is repeated for all the available data points (209 times for our dataset).

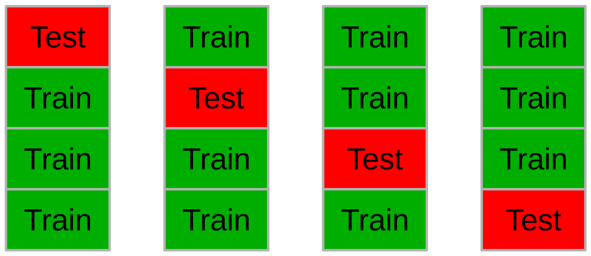


Figure 4-19: An illustration of how jackknife evaluation works. In this example, there are four data points. At each step, a data point is held-out for testing while the other three points are used for training. This process is repeated for all the data points (four times in this example).

4.5.1 Model Performance

The first evaluation task is to measure the overall performance of our models in comparison with certain baselines. Recall that out of the 209 rumors, $113(54\%)$ are false and $96(46\%)$ are true. Table 4.4 shows the performance of the DTW and HMM models compared to four baselines. These baselines are: a majority classifier, a retweet classifier, an N-gram classifier, and a classifier trained on features used by Castillo et al. [10], which we call *CAST*. The majority classifier always predicts the veracity to be of the majority class (in this case the false class). The retweet classifier predicts the veracity of rumors purely based on the number of times they have been retweeted. The N-Gram classifier is trained on the 1000 most common unigrams, bigrams and trigrams in false and true rumors. The retweet and N-Gram classifier represent simple linguistic and propagation features and the *CAST* classifier is the veracity prediction model most related to our model. As it can be seen in Table 4.4, both the DTW and HMM models greatly outperform the baseline models, with the HMM model performing the best with an overall accuracy of .75. Since HMM is the best performing model, from this point on, unless otherwise noted, the model being discussed is the HMM model.

Model	**All Rumors**	**False Rumors**	**True Rumors**
Majority	.54	**1.**	.0
Retweet	.56	.61	.50
N-Gram	.59	.61	.58
CAST	.64	.68	.60
DTW	.71	.73	.69
HMM	**.75**	.77	**.73**

Table 4.4: Accuracy of *Rumor Gauge* models and a few baselines.

Table 4.5 shows the performance of the HMM model for each of the three categories of features. As you can see, the model trained on the propagation features outperforms the linguistic and user models by a sizeable margin. In order to get a better understanding of the performance of our models, we can look at their receiver operating characteristic (ROC) curves. Recall that the principle on which ROC curves operate on is explained in detail in *Section 3.1.17* of this thesis. Figure 4-20 shows four ROCs curves, for the HMM model trained on all the features, the propagation features, the user features and the linguistic features.

Depending on the application, the user can pick different point on the ROC curve for the model to operate on. For example, the user could be a financial markets expert who needs to have a list of most of the true rumors spreading on Twitter about an event of interest, so that he could use the information to make stock trades. This user could perhaps tolerate some false rumors being mistakenly identified as true. The optimal operating point for this user would be around 0.6 on the false-positive axis (x-axis), which corresponds to .97 on the y-axis. At that point, the model would correctly identify 97% of the true rumors, but also getting a sizeable (around 60%) of the false rumors mistakenly classified as true. On the other hand, if the user was a journalist who had limited resources and wanted a list of true rumors that he or she could trust, the journalist would perhaps pick a point on the curve with the false positive rate closer to zero. For example, the journalist could perhaps pick the point $x = .16, y = .70$. At this point, the model would correctly identify 70% of the true rumors, with only getting around 16% of the false rumors mistakenly classified as true. These two examples are just to illustrate how one might use our system for real-world applications. In the next chapter we will discuss in detail several hypothetical real world applications for our system.

Feature Category	**All Rumors**	**False Rumors**	**True Rumors**
Linguistic	.64	.70	.58
User	.65	.64	.66
Propagation	**.70**	**.72**	**.66**
All	.75	.77	.73

Table 4.5: Accuracy of *Rumor Gauge* using each of the three feature categories independently.

4.5.2 Contribution of Features

In this section, we report the contribution of each of the 17 features used in our model. Recall that these 17 features were selected from a list of 37 features. The 17 that were selected all significantly contributed to the performance of our model. This was done by ranking the features by the chisquare test, with the null hypothesis being that there is no significant association between a feature and the outcome, after taking into account the other features in the model. Note that this method only indicated the strength of evidence that there is some association, not the magnitude of the association. Therefore, in order to measure the magnitude of association between each of the 17 features and the outcome of our model, we trained the model using each of the features independently. We then measured the accuracy of the model and used that value as a proxy for the magnitude of the contribution of each feature. Table 4.6 shows the contribution of each feature.

The relative contribution of the features is more clearly illustrated in Figure 4-21. It is clear from this figure that the propagation features (shown in blue) do most of the heavy lifting, with one particular feature, the *fraction of low-to-high diffusion* contributing a great

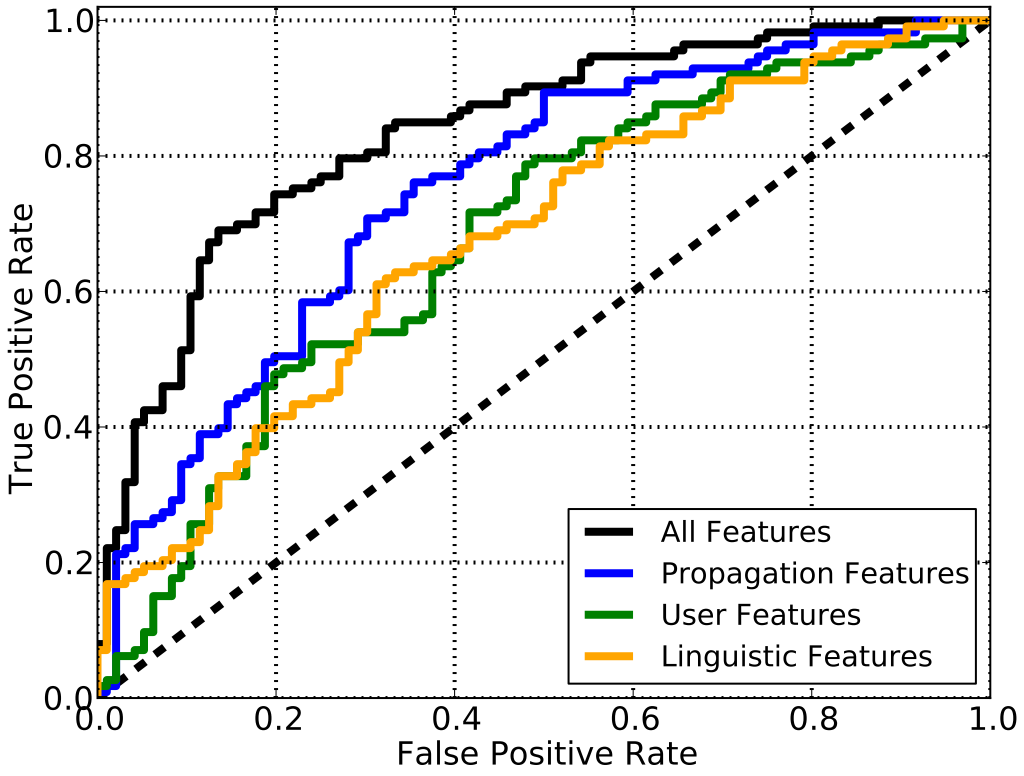


Figure 4-20: ROC curves for the HMM model trained on different sets of features.

Feature	**Contribution**
Fraction of low-to-high diffusion	.68
Average depth-to-breadth ratio	.63
Fraction of nodes in LCC	.63
Ratio of tweets containing negation	.61
User controversiality	.61
Ratio of new users	.60
Ratio of original tweets	.59
User credibility	.58
User originality	.58
Fraction of tweets with outside links	.58
Average formality & sophistication	.57
User influence	.57
Ratio of Tweets containing tentatives	.56
Ratio of tweets containing opinion	.56
User engagement	.56
User role	.56
Fraction of isolated nodes	.56

Table 4.6: The contribution of each of the 17 features to out model. Blue corresponds to propagation features, purple to user features and orange to linguistic features.

deal to the model. However, we will show in the next section that the three categories each contribute to the performance of the model at different times.

4.5.3 Accuracy vs Latency

Finally, we measured the accuracy of our model as a function of latency (i.e., time elapsed since the beginning of a rumor). The goal of this evaluation is to assess how well our system would function for time-sensitive tasks. Additionally, through this evaluation, we can determine which category of features perform best at which times during the life of a rumor. Since the 209 rumors have varying durations, we study latency as the percentage of time passed from the beginning of a rumor to the trusted verification of that rumor. So 0% latency refers to the very beginning of the rumors, and at 100% latency is the time at which they were verified by trusted sources, and 200% latency is when the time from the beginning of rumors to their trusted verification equals the amount of time passed since the

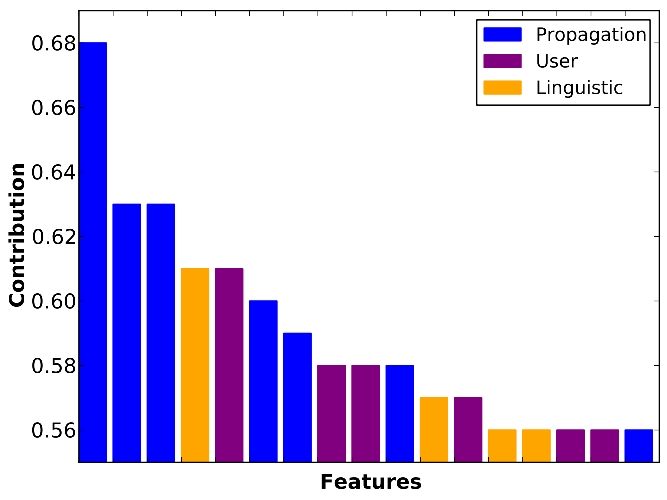


Figure 4-21: An illustration of the relative contribution of the 17 features used in our model. These features are in the same order as in Table 4.6. Blue corresponds to propagation features, purple to user features and orange to linguistic features.

trusted verification.

Figure 4-22 shows the accuracy versus latency for the model using all the features, the propagation features, the user features and the linguistic features. The dashed red line in the figure represents trusted verification of the rumors. As it can be seen, the model reaches 75% accuracy right before trusted verification. Several interesting observations can be made from this figure. First, the model barely performs better than chance before 50% latency. Second, the contribution of the different categories of features varies greatly over time. Different categories of features kick-in and plateau at different times. For example, the propagation features do not contribute much until around 65% latency. The initial performance of the model seems to be fuelled mostly by the linguistic and user features, which then plateau at around 55% latency, as the amount of information they can contribute to the model saturates. Finally, the plot shows the overall performance of the model and the performance of the model trained on propagation features only to be tightly correlated. This is not surprising since in the last section we showed that the propagation features do most of the heavy lifting in our model (see Figure 4-21 or Table 4.5). In chapter 5 of this thesis we will further discuss what the accuracy versus latency curve would mean and how it could be utilized for potential real-world applications of our system.

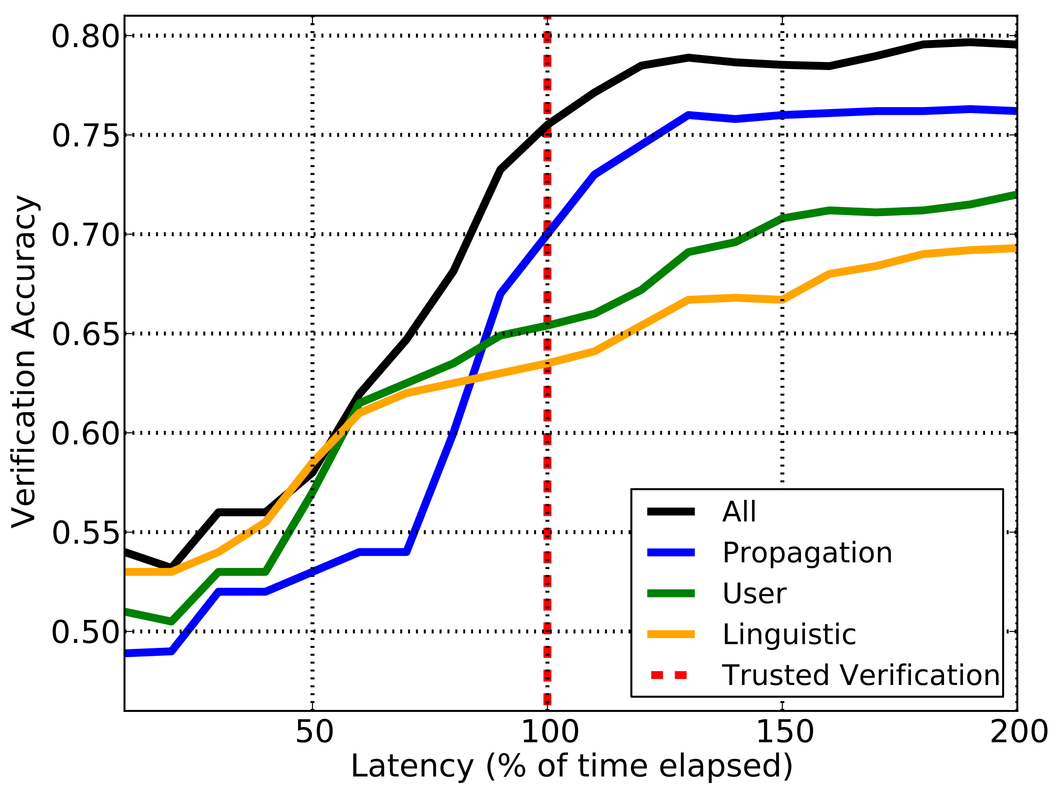


Figure 4-22: Accuracy of the model as a function of latency. All 209 rumors have been aligned by using the percentage of duration instead of hours. The dashed red line represents trusted verification of the rumors.

4.6 A Tale of Two Rumors

In this section we will describe in detail the performance of our model on two rumors, one false and one true, from our dataset of 209 rumors. The first rumor, which turned out to be true, is about the 2014 Ebola epidemic. The rumor is shown below:

Nigerian nurse suspected of having Ebola fled quarantine.

This rumor first appeared on Twitter at 8:08 am, on August 13th of 2014. The rumor quickly spread through Twitter, and it was reported to be true by several trusted sources around 9 hours after the first tweet. Figure 4-23 shows the prediction of the model for this rumor from the very start, through the trusted verification 9 hours later and an additional 10 hours after that. The y-axis shows the veracity prediction and confidence of the model. Any value above 0 means the model is predicting the veracity to be true, with higher the value the more confident the model. The background color also corresponds to the prediction of the model, with green corresponding to true and red to false (and the saturation of the background colors correspond to confidence, with higher saturation meaning higher confidence). As you can see in the figure, several hours before trusted verification the model was able to with some confidence correctly predict the veracity of the rumor. The curves in this figure also show the same behavior seen in Figure 4-22, where the linguistic and user features contribute in the beginning but then plateau soon after. Conversely,the propagation features do not contribute much to the model in the beginning and then take-off after a while, doing the majority of the heavy lifting in the model.

The second rumor, which turned out to be false, is about the 2014 Ferguson unrest. The rumor is shown below:

Officer Darren Wilson suffered fracture to his eye socket during Mike Brown attack.

This rumor first appeared on Twitter at 8:01 am, on August 19th of 2014 and spread through Twitter. It was reported to be false by several trusted sources around 18 days after the first tweet. Figure 4-24 shows the prediction of the model for this rumor from the very start, through the trusted verification 1.8 days later and an additional 1.7 days after that. Similar as with the true rumor, the model was able to with relatively high confidence correctly predict the veracity of the rumor, several hours before trusted verification. The curves here also show similar behavior to what is seen in Figure 4-22.

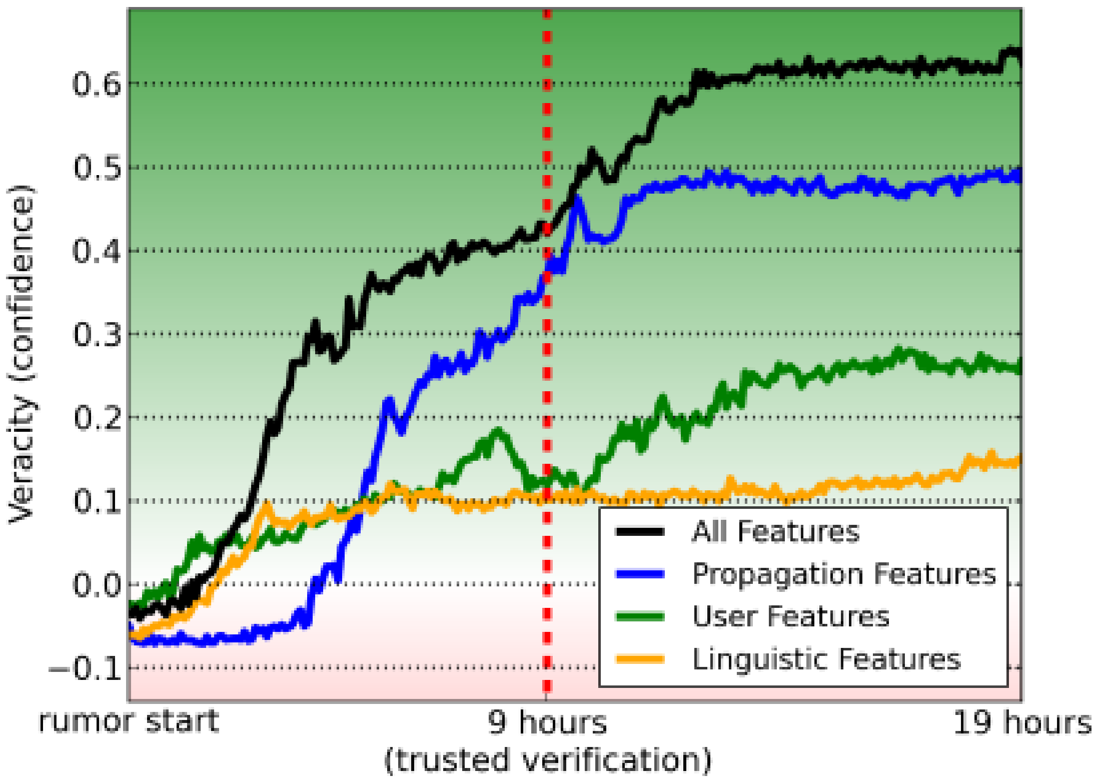


Figure 4-23: The prediction of the model as a function of time, for a true rumor about Ebola. You can see the performance of the model when trained on all the features, the propagation features, the user features and the linguistic features. The y-axis shows the veracity prediction of the model. Any value above 0 means the model is predicting the veracity to be true and vice-versa. The background color also corresponds to the prediction of the model, with green corresponding to true and red to false. The saturation of the background colors correspond to confidence, with higher saturation meaning higher confidence.

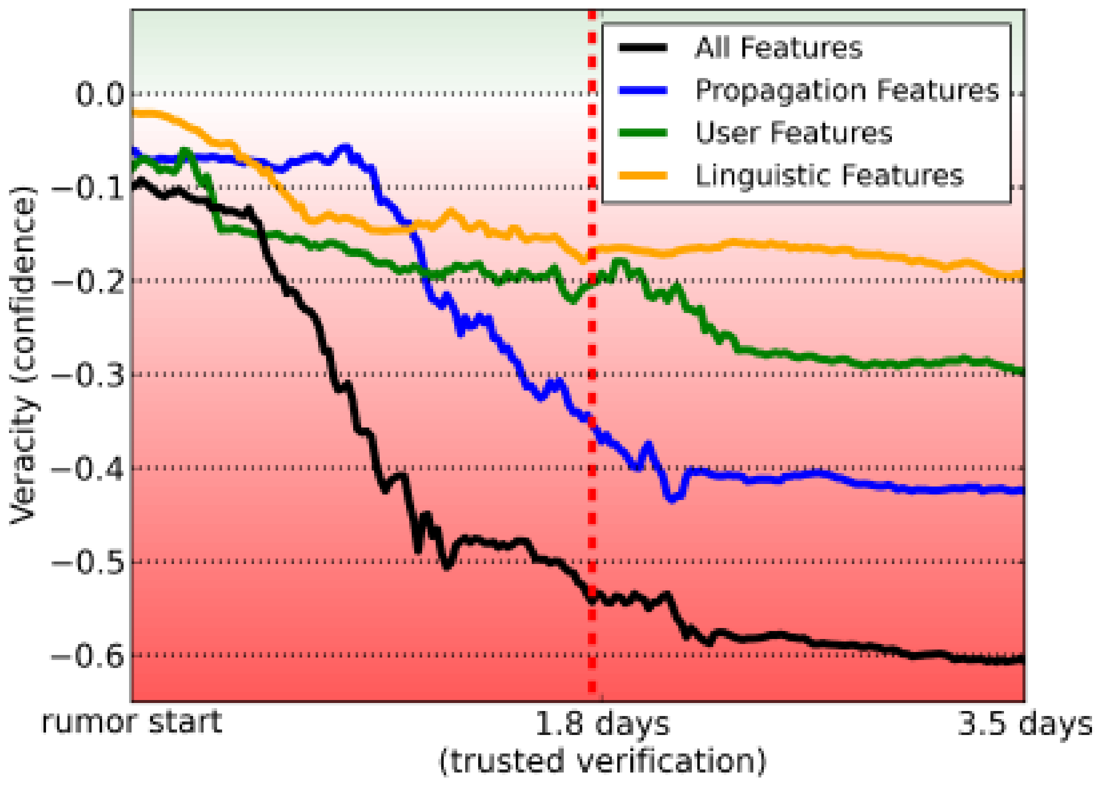


Figure 4-24: The prediction of the model as a function of time, for a false rumor about the Ferguson unrest. You can see the performance of the model when trained on all the features, the propagation features, the user features and the linguistic features. The y-axis shows the veracity prediction of the model. Any value below 0 means the model is predicting the veracity to be false and vice-versa. The background color also corresponds to the prediction of the model, with green corresponding to true and red to false. The saturation of the background colors correspond to confidence, with higher saturation meaning higher confidence.

4.7 Anatomy of Rumors in Twitter

Through our work on *Rumor Gauge* we were able to identify salient characteristics of rumors in Twitter by examining the language of rumors, the users involved in spreading rumors and the propagation dynamics of rumors. Crucially, we were able to identify key differences in each of the three characteris in the spread of false and ture rumors. In addition to allowing us to create a state-of-the-art rumor verification system, these features also shed some light on the anatomy of rumors in Twitter.

Many insights about the nature of rumors on Twitter can be gained from our work. Here we will discuss several of more interesting ones. First of all, the diffusion of information from users with low influence to users with high influence is a phenomenon which is seen much more frequently when the information is true. The reason for this is perhaps because the user with the high influence would not risk retweeting a less known user's information unless the person had very good reasons to believe the information is true. Second, the formality and sophistication of the language used to describe false rumors seems to be bimodal. The language on average tends to be either more formal and sophisticated or less formal and sophisticated than other language used about an event (this bimodality is also seen in language used in spams). Third, not surprisingly perhaps, false rumors are more likely to be spread by users who are influential but controversial, while true rumors are more likely spread by influential and credible users. Finally, from the very genesis of false rumors, there tends to be people refuting the rumors, much more so than for false rumors. Though unfortunately, their collective voice is usually muffled by the much louder voice of the people spreading the rumor.

Chapter 5

Real-World Applications of the System

The ability to track rumors and predict their outcomes can have real-world applications for news consumers, financial markets, journalists and emergency services. In this chapter we discuss hypothetical real-world applications for the rumor detection and verification system developed in this thesis. Though there are many ways to utilize these tools for real-world purposes, we will discuss three general utilities afforded by our system.

5.1 Bandwidth Reduction

The first utility afforded by our system is the bandwidth reduction of information. The term bandwidth reduction is borrowed from the field of telecommunication, where it means the reduction of the bandwidth needed to transmit a given amount of data in a given time. In the context of our system, bandwidth reduction refers to the reduction in the amount of information a user has to sift through in order to make sense of the information and misinformation spreading about a real-world event.

For instance, a law enforcement employee trying to identify all the misinformation (i.e. false rumors) spreading about a real-world emergency has to go through hundreds of thousands, if not millions, of tweets in a very short amount of time (e.g., in the case of the Boston Marathon bombings, there were more than 20 millions tweets about the event in less than 10 days). With the aid of our system, the amount of information the law enforcement employee has to study is reduced by a large percent. This reduction happens at three junctions: when the tweets not containing an assertion are filtered out, when the tweets are clustered together to form rumors, and when the rumors are verified. Going back to the case of the Boston Marathon bombings, the 20 millions tweets were reduced by about 50% at the first stage, then reduced by about four orders of magnitude in the second stage, and finally reduced by about 40% in the last stage. The 40% reduction in the verification step is inferred by the ROC curve shown in Figure 4-20, since at around 40% true positive rate, the false positive rate is very close to zero. Figure 5-1 shows a logarithmically scaled plot, illustrating the reduction of information afforded by each part of our system, for the Boston Marathon bombings dataset. The total reduction is approximately 2×10^{-5}.

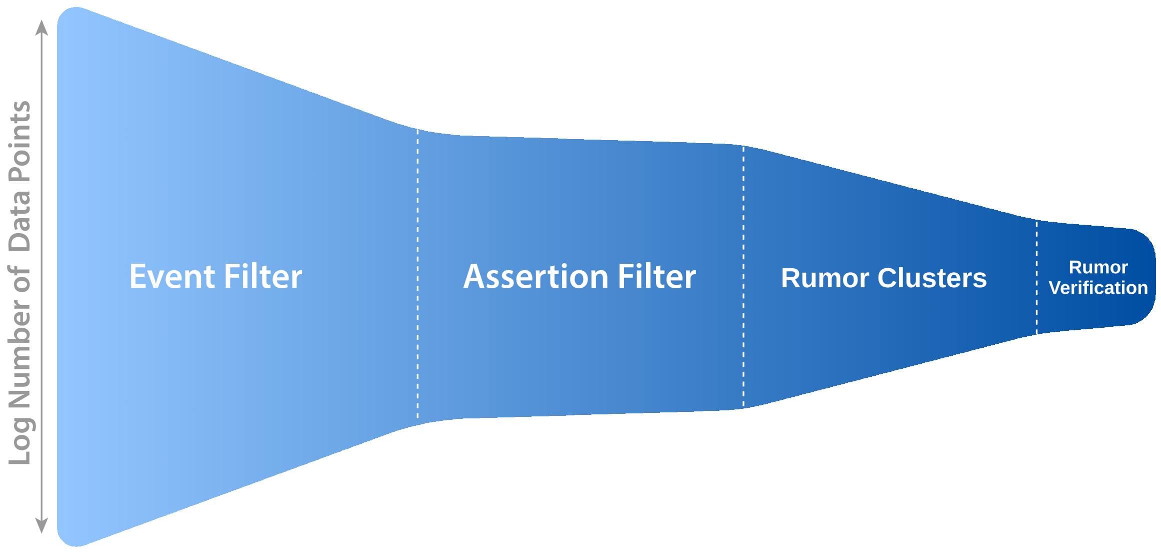


Figure 5-1: Logarithmically scaled plot of the bandwidth reduction afforded by the rumor detection and verification system for the Boston Marathon bombings dataset.

5.2 Priority Queue

The second utility afforded by our system is assigning *priority* to rumors about an event so that they could be ordered in a queue based on their priority. In a real-world situation, the rumors would be addressed based on their priority. The priorities assigned to rumors would be based on their size, predicted veracity and the confidence of the prediction. Though currently our system can not predict the future impact (i.e. reach) of a rumor, this would be a very useful variable to also consider when assigning priority. Note that the priorities would change and the queue would be reordered over time as these variables change.

For example, let's take a journalist covering a real-world emergency. The journalist plans to cover the true rumors that are spreading about the event before anyone else. She also has limited time and resources and so can only investigate 5 rumors. The journalist wants to maximize the chance that the 5 rumors she is covering, (a) are true, and (b) have the biggest footprints (i.e. are most talked about). First, *Hearsift* would detect tens to hundreds of rumors about the said event (each with different sizes). Next, *Rumor Gauage* would predict the veracity of these rumors. Each veracity prediction would have a different confidence. The true and false rumors are then ordered in different queues based on their confidence and/or size. The journalist can then address the top 5 rumors in the "true" queue.

5.3 Rumor Analytics Dashboard

Finally, our system can be used to create a *rumor analytics dashboard*. The dashboard would allow users to investigate the anatomy of rumors by digging deeper into the characteristics of their diffusion. In addition to detecting and predicting the veracity of rumors, the rumor analytics dashboard would also allow the users to see the inner workings of our system to understand how the system is verifying the rumors. For example, using the dashboard, the users can see which of the features are driving the prediction of the model. Moreover, the users would be able to see the actual data underneath the predictions (i.e., who are the controversial users that the system has detected, which tweets are lowering the formality sophistication score, etc). By allowing users to see under the hood of our system, we guide the user but leave the final decision of whether a rumor is true or false in the hands of the user.

The rumor analytics dashboard enables human-machine collaboration by utilizing what our system and humans do best. Our system can analyse and detect patterns in large amount of data, while humans can quickly adapt to new situations and tune the system accordingly.

Chapter 6

Conclusions

This thesis described a system for automatic detection and verification of rumors about real-world events on Twitter. Here we will summarize the contributions of this thesis and explore possible future directions for extending this work.

6.1 Future Directions

There are many ways to extend the works presented in this thesis, several of which have been mentioned through out this document. Here, we will discuss what we believe to be the the four most fruitful directions for future work. These directions are:

- Rumor analytics dashboard.
- Extend system to other media platforms (social and traditional).
- Predict the impact of rumors.
- Strategies for dampening the effects of rumors.

The idea behind the rumor analytics dashboard was discussed in the last chapter. The rumor analytics dashboard is the most immediately useful and achievable extension to this work as it does not require any new analytic work, just the creation of user-friendly interface.

Next, we would like to extend our system to cover other media platforms. As the title implies, all the work presented in this thesis is focused on rumors in Twitter. However, similar techniques and algorithms could potentially be applied to other online and publicly available media platforms, be it social (e.g., Reddit, Facebook, etc) or traditional (e.g., BBC, CNN, etc). Though some of the features described for this work are Twitter-specific, many of the features are platform-agnostic and can readily be extracted and processed from different platforms. We have already done some preliminary work on Reddit and the results, though too early to share, look promising.

In addition to detecting and predicting rumors, it will be very useful to predict the impact the rumors would have on individuals and society. There are many ways one can define *impact*; for example, the total reach of a rumor (i.e., how many people are exposed to the rumor) can be an estimate of impact. By predicting the impact of rumors, one can better assign priorities to rumors to be addressed (this is discussed in depth in Section 5.2 of this thesis). This is especially relevant for emergency services who might want to respond to false rumors that might have a large negative impact.

Finally, a system that can detect rumors and predict their veracity and maybe impact is indeed a very valuable and useful tool. However, in some cases the users of the system might want to dampen the effects of rumors, especially ones that are predicted to be false and impactful. This again would be something that would have the most relevance to the emergency services dealing with real-world emergencies as they are the ones that have to deal with the consequences and the fallout of rumors on social media. For example, in the case of Boston Marathon bombing, there were several unfortunate instances of innocent people being implicated in witch-hunts [51, 56, 99], one of which became the Boston Marathon bombing's largest rumor. This rumor was that a missing Brown University student, named *Sunil Tripathi*, was one of the suspects the police were looking for. This led to a great amount of confusion and heartache for the family of the accused. By having strategies to dampen the effects of such rumors, our system can move from the role of a passive observe of rumors to one that can actively reduce the amount of harm done by the rumors. Strategies for dampening rumors could be something as simple as requesting influential users to publicly rebuke the rumors, or it could be something much more sophisticated and surgical.

6.2 Contributions

The work described in this thesis describes how to create a system for detection and verification of rumors on Twitter. The key scientific contribution of this is the identification of salient characteristics of rumors on Twitter and using that to develop a computational model of rumors.

We created models for detection and verification of rumors that propagate on Twitter. The rumor detection system, called *Hearsift*, operates by classifying and clustering assertions made about an event. These assertions are classified through a state-of-the-art speech-act classifier for Twitter developed for this thesis. The classifier utilizes a combination of semantic and syntactic features and can identify assertions with 91% accuracy.

For the rumor verification system, called *Rumor Gauge*, we identified salient characteristics of rumors by examining three aspects of diffusion: linguistic, the users involved, and the temporal propagation dynamics. We then identified key differences in each of the three characteristics in the spread of true and false rumors. A time series of these features extracted for a rumor can be classified as predictive of the veracity of that rumor using Hidden Markov Models. *Rumor Gauge* was tested on 209 rumors from several real-world events. The system predicted the veracity of 75% of the rumors correctly, before verification by trusted channels (trustworthy major governmental or news organizations).

The ability to detect and verify rumors through *Hearsift* and *Rumor Gauge* respectively, can have immediate real-world relevance for news consumers, financial markets, journalists, and emergency services.

6.3 Concluding Remarks

Hearsift and *Rumor Gauge* have immediate real-world relevance for various sectors of society. However, the work presented in this thesis has been almost entirely analytical. Though from a sceintific and analytic point of view this work has been very fulfilling, we have not yet developed a complete tool to be used by interested parties. Given the strong performance of our system, and its potential to help individuals and society during real-world emergencies, we think it is essential to design, develop and user-test a tool based on this work.

Furthermore, we believe the features and techniques described in this thesis have the potential to influence other works focused on Twitter and other social media platforms. Finally, though we had full access to Twitter's historical data, except for a few features used in the rumor verification system, most of the work described in this thesis can be replicated using the Twitter public API.

Appendix A

Enhanced Twitter Sentiment Classification Using Contextual Information

A.1 Introduction

The rise in popularity and ubiquity of Twitter have made sentiment analysis of tweets an important and much covered area of research. However, the 140 character limit imposed on tweets make it hard to use standard linguistic methods for sentiment classification. On the other hand, what tweets lack in structure they make up with sheer volume and rich metadata. This metadata includes geolocation, temporal and author information. We hypothesize that sentiment is dependent on all these contextual factors. Different locations, times and authors have different emotional valences. In this paper, we explored this hypothesis by utilizing distant supervision to collects millions of labelled tweets from different locations, times and authors. We used this data to analyse the variation of tweet sentiments across different authors, times and locations. Once we explored and understood the relationship between these variables and sentiment, we used a Bayesian approach to combine these variables with more standard linguistic features such as n-grams to create a state-of-the-art

Twitter sentiment classifier.

One area of research that has attracted great attention in the last few years is that of sentiment classification of tweets. Through sentiment classification and analysis, one can get a picture of people's attitudes about particular topics on Twitter. This can be used for measuring people's attitudes towards brands, political candidates, and social issues. There have been several works that do sentiment classification on Twitter using standard sentiment classification techniques, with variations of n-gram and bag of words being the most common. There have been attempts at using more advanced syntactic features as is done in sentiment classification for other domains [83, 67], however the 140 character limit imposed on tweets makes this hard to do as each article in the Twitter training set consists of sentences of no more than several words, many of them with irregular form [85].

On the other hand, what tweets lack in structure they make up with sheer volume and rich metadata. This metadata includes geolocation, temporal and author information. We hypothesize that sentiment is dependent on all these contextual factors. Different locations, times and authors have different emotional valences. For instance, people are generally happier on weekends, more depressed at the end of summer holidays, and happier in certain hours of the day and certain states in the United States. Moreover, people have different baseline emotional valences from one another. These claims are supported for example by the annual Gallup poll that ranks states from most happy to least happy [32], or the work by Csikszentmihalyi and Hunter [19] that showed reported happiness varies significantly by day of week and time of day. We believe these factors manifest themselves in sentiments expressed in tweets and that by accounting for these factors, we can improve sentiment classification on Twitter.

In this work, we explored this hypothesis by utilizing *distant supervision* [83, 34] to collect millions of labelled tweets from different locations (within the USA), times of day, days of week, months and authors. We used this data to analyse the variation of tweet sentiments across the aforementioned categories. We then used a Bayesian approach to incorporate the relationship between these factors and tweet sentiments into standard n-gram based Twitter sentiment classification.

A.2 Approach

The main hypothesis behind this work is that the average sentiment of messages on Twitter is different in different contexts. Specifically, tweets in different spatial, temporal and authorial contexts have on average different sentiments. Basically, these factors (many of which are environmental) have an affect on the emotional states of people which in turn have an effect on the sentiments people express on Twitter and elsewhere. In this paper, we used this contextual information to better predict the sentiment of tweets.

Luckily, tweets are tagged with very rich metadata, including location, timestamp, and author information. By analysing labelled data collected from these different contexts, we calculated *prior* probabilities of negative and positive sentiments for each of the contextual categories shown below:

- The states in the USA (50 total).
- Hour of the day (HoD) (24 total).
- Day of week (DoW) (7 total).
- Month (12 total).
- Authors (57710 total).

This means that for every item in each of these categories, we calculated a probability of sentiment being positive or negative based on historical tweets. For example, if seven out of ten historical tweets made on Friday where positive then the prior probability of sentiment being positive for tweets sent out on Friday is 0.7 and the prior probability of sentiment being negative is 0.3. We then trained a Bayesian sentiment classifier using a combination of these prior probabilities and standard n-gram models. The model is described in great detail in the ”Baseline Model” and ”Contextual Model” sections of this paper.

In order to do a comprehensive analysis of sentiment of tweets across aforementioned contextual categories, a large amount of labelled data was required. We needed thousands of tweets for every item in each of the categories (e.g. thousands of tweets per hour of day, or state in the US). Therefore creating a corpus using human annotated data would have been impractical. Instead we turned to distant supervision techniques to obtain our corpus. Distant supervision allows us to have noisy but large amount of annotated tweets.

There are different methods of obtaining labelled data using distant supervision [83, 34, 5, 21], we used emoticons to label tweets as positive or negative, an approach that was introduced by Read [83] and used in multiple works [34, 21]. We collected millions of English language tweets from different times, dates, authors and US states. We used a total of six emoticons, three mapping to positive and three mapping to negative sentiment (table A.1). We identified more than 120 positive and negative ASCII emoticons and unicode emojis[1] but we decided to only use the six more common emoticons in order to avoid possible selection biases. For example, people who use obscure emoticons and emojis might have a different base sentiment from those who do not. Using the six most commonly used emoticons limits this bias. Since there are no ”neutral” emoticons, our dataset is limited to tweets with positive or negative sentiments. Accordingly, in this work we are only concerned with analysing and classifying the polarity of tweets (negative vs. positive) and not their subjectivity (neutral vs. non-neutral). Blow we will explain our data collection and corpus in greater detail.

Positive Emoticons	Negative Emoticons
:)	:(
:-)	:-(
:)	: (

Table A.1: List of emoticons.

[1]Japanese pictographs similar to ASCII emoticons

A.3 Data Collection and Datasets

We collected two datasets, one massive and labelled through distant supervision, the other small and labelled by humans. The massive dataset was used to calculate the prior probabilities for each of our contextual categories. Both datasets were used to train and test our sentiment classifier. The human labelled dataset was used as a sanity check to make sure the dataset labelled using emoticons classifier was not too noisy and that the human and emoticon labels matched for a majority of tweets.

A.3.1 Emoticon-based Labelled Dataset

We collected a total of 18 million, geo-tagged, English language tweets over three years, from January 1st, 2012 to January 1st, 2015, evenly divided across all 36 months, using Historical PowerTrack for Twitter[2] provided by GNIP[3]. We created geolocation bounding boxes[4] for each of the 50 states which were used to collect our dataset. All 18 millions tweets originated from one of the 50 states and are tagged as such. Moreover, all tweets contained one of the six emoticons in table A.1 and were labelled as either positive or negative based on the emoticon. Out of the 18 million tweets, 11.2 million (62%) were labelled as positive and 6.8 million (38%) were labelled as negative. The 18 million tweets came from $7,657,158$ distinct users.

A.3.2 Human Labelled Dataset

We randomly selected 3000 tweets from our large dataset and had all their emoticons stripped. We then had these tweets labelled as positive or negative by three human annotators. We measured the inter-annotator agreement using *Fleiss' kappa*, which calculates the degree of agreement in classification over that which would be expected by chance [28]. The *kappa* score for the three annotators was 0.82, which means that there were disagree-

[2]Historical PowerTrack for Twitter provides complete access to the full archive of Twitter public data.
[3]https://gnip.com/
[4]The bounding boxes were created using http://boundingbox.klokantech.com/

ments in sentiment for a small portion of the tweets. However, the number of tweets that were labelled the same by at least two of the three human annotator was 2908 out of of the 3000 tweets (96%). Of these 2908 tweets, 60% were labelled as positive and 40% as negative.

We then measured the agreement between the human labels and emoticon-based labels, using only tweets that were labelled the same by at least two of the three human annotator (the majority label was used as the label for the tweet). Table A.2 shows the confusion matrix between human and emoticon-based annotations. As you can see, 85% of all labels matched ($\frac{1597+822}{1597+882+281+148} = .85$).

	Human-Positive	Human-Negative
Emot-Positive	1597	281
Emot-Negative	148	882

Table A.2: Confusion matrix between human-labelled and emoticon-labelled tweets.

These results are very promising and show that using emoticon-based distant supervision to label the sentiment of tweets is an acceptable method. Though there is some noise introduced to the dataset (as evident by the 15% of tweets whose human labels did not match their emoticon labels), the sheer volume of labelled data that this method makes accessible, far outweighs the relatively small amount of noise introduced.

A.3.3 Data Preparation

Since the data is labelled using emoticons, we stripped all emoticons from the training data. This ensures that emoticons are not used as a feature in our sentiment classifier. A large portion of tweets contain links to other websites. These links are mostly not meaningful semantically and thus can not help in sentiment classification. Therefore, all links in tweets were replaced with the token ”URL”. Similarly, all mentions of usernames (which are denoted by the @ symbol) were replaced with the token ”USERNAME”, since they also can not help in sentiment classification. Tweets also contain very informal language and as such, characters in words are often repeated for emphasis (e.g., the word *good* is used with an arbitrary number of *o*’s in many tweets). Any character that was repeated more than two times was removed (e.g., *goooood* was replaced with *good*). Finally, all words in the tweets were stemmed using *Porter Stemming* [77].

A.4 Baseline Model

For our baseline sentiment classification model, we used our massive dataset to train a negative and positive n-gram language model from the negative and positive tweets. An n-gram language model is a compact probabilistic model that adequately explains the observed linguistic data. N-gram models assign probabilities to word sequences $w_1 \ldots w_\ell$. N-gram language modelling [15, 43, 61] is an effective technique that treats words as samples drawn from a distribution conditioned on other words that occur in the corpus, usually the immediately preceding $n-1$ words enabling them to capture the strong local dependencies between words. The probability of a sequence of ℓ words, written compactly as w_1^ℓ is $\Pr(w_1^\ell)$ and can be factored exactly as

$$\Pr(w_1^\ell) = \Pr(w_1) \prod_{i=2}^{\ell} \Pr(w_i | w_1^{i-1}) \tag{A.1}$$

However, parameter estimation in this full model is intractable, as the number of possible word combinations grows exponentially with sequence length. N-gram models address this with the approximation $\tilde{\Pr}(w_i | w_{i-n+1}^{i-1}) \approx \Pr(w_i | w_1^{i-1})$ using only the preceding $n-1$ words for context. A bigram model ($n = 2$) uses the preceding word for context, while a unigram model ($n = 1$) does not use any context.

As our baseline model, we built purely linguistic bigram models in Python, utilizing some components from NLTK [6]. These models used a vocabulary that was filtered to remove words occurring 5 or fewer times. Probability distributions were calculated using Witten-Bell smoothing [43]. Rather than assigning word w_i the maximum likelihood probability estimate $p_i = \frac{c_i}{N}$, where c_i is the number of observations of word w_i and N is the total number of observed tokens, Witten-Bell smoothing discounts the probability of observed words to $p_i^* = \frac{c_i}{N+T}$ where T is the total number of observed word types. The remaining Z words in the vocabulary that are unobserved (i.e. where $c_i = 0$) are given $p_i^* = \frac{T}{Z(N+T)}$. In addition to Witten-Bell smoothing, the bigram models also used "back-off" smoothing[45], in which an n-gram model falls back on an $(n-1)$-gram model for words that were unobserved in the n-gram context.

In order to classify the sentiment of a new tweet, it is evaluated in both the negative and positive bigram models. Each model returns a probability of fit for the tweet. These probabilities are then used to classify the tweet. (If the probability returned by negative model is greater then the probability returned by the positive model, the tweet is classified as negative and vice versa.) This method is very flexible, we can even define a neutral class for when the probabilities returned by the models are close to each other, if we so choose. However, in this paper we classify all tweets as either negative or positive. Another way to look at our models is through a Bayesian lens, as shown in equation A.2 below:

$$\Pr(\theta_s \mid W) = \frac{\Pr(W \mid \theta_s)\Pr(\theta_s)}{\Pr(W)} \tag{A.2}$$

Where θ_s stands for $\theta_{sentiment}$ and can be $\theta_{positive}$ or $\theta_{negative}$, corresponding to the hypothesis that the sentiment of the tweet is positive or negative respectively. W is the sequence of ℓ words, written as w_1^ℓ that make up the tweet, making $\Pr(W)$ the probability of such sequence. Since we are using a bigram model, the probability of the sequence of words w_1^ℓ is thus the same as shown in equation A.1. Therefore equation A.2 can be written as:

$$\Pr(\theta_s \mid W) = \frac{\prod_{i=2}^{\ell}\Pr(w_i \mid w_{i-1}, \theta_s)\Pr(\theta_s)}{\prod_{i=2}^{\ell}\Pr(w_i w_{i-1})} \tag{A.3}$$

As we are dealing with very small numbers, we can rewrite equation A.3 as the sum of log probabilities in order to avoid possible floating point inaccuracies, as shown in equation A.4.

$$\begin{aligned}\log(\Pr(\theta_s \mid W)) = &\sum_{i=2}^{\ell} \log(\Pr(w_i \mid w_{i-1}, \theta_s)) \\ &+ \log(\Pr(\theta_s)) - \sum_{i=2}^{\ell} \log(\Pr(w_i \mid w_{i-1}))\end{aligned} \tag{A.4}$$

This is our purely linguistic Bayesian model which serves as our baseline model.

A.5 Contextual Model

The Bayesian approach allows us to easily integrate the contextual information into our models. $\Pr(\theta_s)$ in equation A.4 is the prior probability of a tweet having the sentiment s (negative or positive). The prior probability ($\Pr(\theta_s)$) can be calculated using the contextual information of the tweets. Therefore, $\Pr(\theta_s)$ in equation A.4 is replaced by $\Pr(\theta_s|C)$, which is the probability of the hypothesis (negative or positive) given the contextual information. $\Pr(\theta_s|C)$ is the posterior probability of the following Bayesian equation:

$$\Pr(\theta_s \mid C) = \frac{\Pr(C \mid \theta_s)\Pr(\theta_s)}{\Pr(C)} \tag{A.5}$$

Where C is the set of contextual variables: $\{State, HoD, Dow, Month, Author\}$. $\Pr(\theta_s|C)$ captures the probability that a tweet is positive or negative, given the state, hour of day, day of week, month and author of the tweet. Equation A.4 can therefore be rewritten to include the contextual information:

$$
\begin{aligned}
\log(\Pr(\theta_s \mid W, C)) = &\sum_{i=2}^{\ell} \log(\Pr(w_i \mid w_{i-1}, \theta_s)) \\
&+ \log(\Pr(C \mid \theta_s)) + \log(\Pr(\theta_s)) \\
&- \log(\Pr(C)) - \sum_{i=2}^{\ell} \log(\Pr(w_i \mid w_{i-1}))
\end{aligned}
\tag{A.6}
$$

In equation A.6 we are treating the linguistic and the contextual information as conditionally independent. We know this to be false since at the very least the author of a tweet would have an impact on the words used in the tweet. It is safe to assume that the same is most likely true of the spatial and temporal variables. For example, people in different states might have different vocabularies, or seasons might have different effects on the emotional state of people in different states (e.g., people in warmer states might prefer winter months and vice versa). However, without making this approximation, parameter estimation for the full model would be intractable, therefore we assume all linguistic and contextual information to be conditionally independent.

Equation A.6 is our extended Bayesian model for integrating contextual information with more standard, word-based sentiment classification. In the next section we will explain how the contextual prior probabilities were calculated.

A.6 Sentiment in Context

We considered five contextual categories: one spatial, three temporal and one authorial. Here is the list of the five categories:

- The states in the USA (50 total) (spatial).
- Hour of the day (HoD) (24 total) (temporal).
- Day of week (DoW) (7 total) (temporal).

- Month (12 total) (temporal).
- Authors (57710 total) (authorial).

We used our massive emoticon labelled dataset to calculate the average sentiment for all of these five categories. A tweet was given a score of -1 if it was labelled as negative and a score 1 if it was labelled as positive, so an average sentiment of 0 for a contextual category would mean that tweets in that category were evenly labelled as positive and negative.

A.6.1 Spatial

All of the 18 million tweets in our dataset originate from the USA and are geo-tagged. Naturally, the tweets are not evenly distributed across the 50 states given the large variation between the population of each state. Figure A-1 shows the percentage of tweets per state, sorted from smallest to largest. Not surprisingly, California has the most number of tweets ($2,590,179$), and Wyoming has the least number of tweets ($11,719$).

Even the state with the lowest percentage of tweets has more than ten thousand tweets, which is enough to calculate a statistically significant average sentiment for that state. The sentiment for all states averaged across the tweets from the three years is shown in figure A-2. Note that an average sentiment of 1.0 means that all tweets were labelled as positive, -1.0 means that all tweets were labelled as negative and 0.0 means that there was an even distribution of positive and negative tweets. The average sentiment of all the states lean more towards the positive side. This is expected given that 62% of the tweets in our dataset were labelled as positive.

It is interesting to note that even with the noisy dataset, our ranking of US states based on their Twitter sentiment correlates with the ranking of US states based on well-being index calculated by Oswald and Wu [70] in their work on measuring well-being and life satisfaction across America. Their data is from the behavioral risk factor survey score (BRFSS), which is a survey of life satisfaction across the United States from 1.3 million citizens. Figure A-3 shows this correlation ($r = 0.44$, $p < 0.005$).

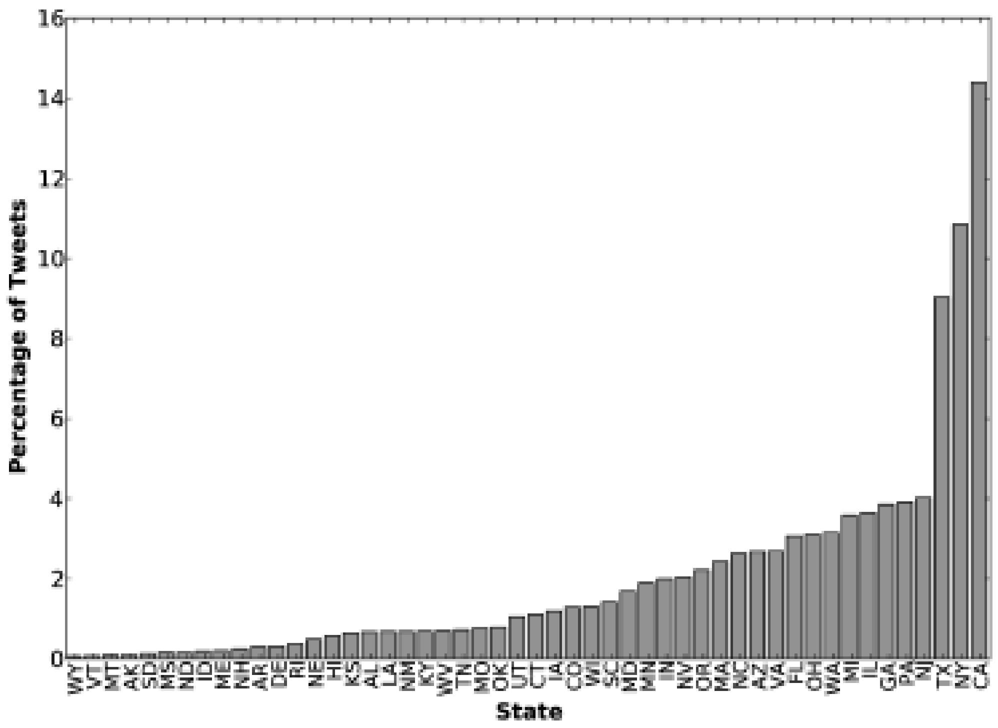


Figure A-1: Percentage of tweets per state in the USA, sorted from lowest to highest.

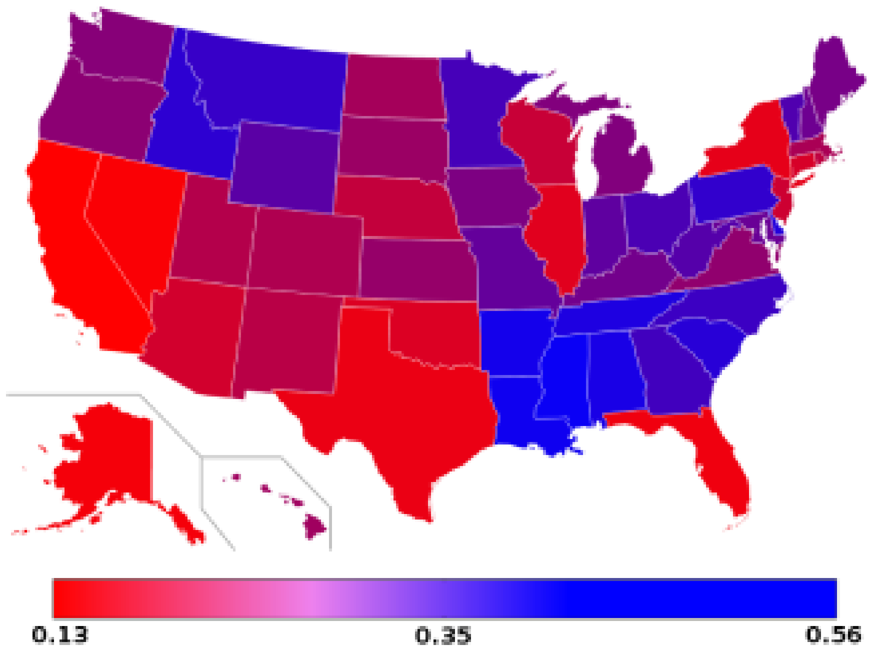


Figure A-2: Average sentiment of states in the USA, averaged across three years, from 2012 to 2014.

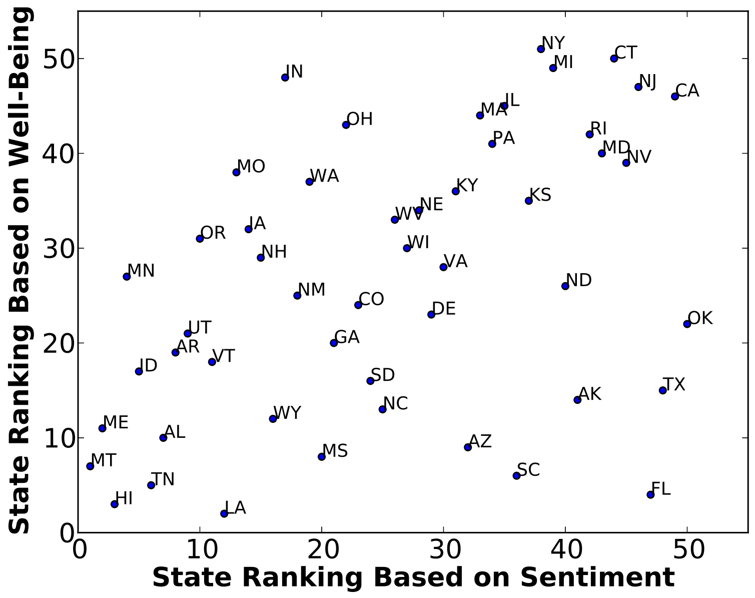


Figure A-3: Ranking of US states based on Twitter sentiment vs. ranking of states based on their well-being index. $r = 0.44$, $p < 0.005$.

A.6.2 Temporal

We looked at three temporal variables: time of day, day of week and month. All tweets are tagged with timestamp data, which we used to extract these three variables. Since all timestamps in the Twitter historical archives (and public API) are in the UTC time zone, we first converted the timestamp to the local time of the location where the tweet was sent from. We then calculated the sentiment for each day of week (figure A-4), hour (figure A-5) and month (figure A-6), averaged across all 18 million tweets over three years. The 18 millions tweets were divided evenly between each month, with 1.5 million tweets per month. The tweets were also more or less evenly divided between each day of week, with each day having somewhere between 14% and 15% of the tweets. Similarly, the tweets were almost evenly divided between each hour, with each having somewhere between 3% and 5% of the tweets.

Some of these results make intuitive sense. For example, the closer the day of week is to Friday and Saturday, the more positive the sentiment, with a drop on Sunday. As with spatial, the average sentiment of all the hours, days and months lean more towards the positive side.

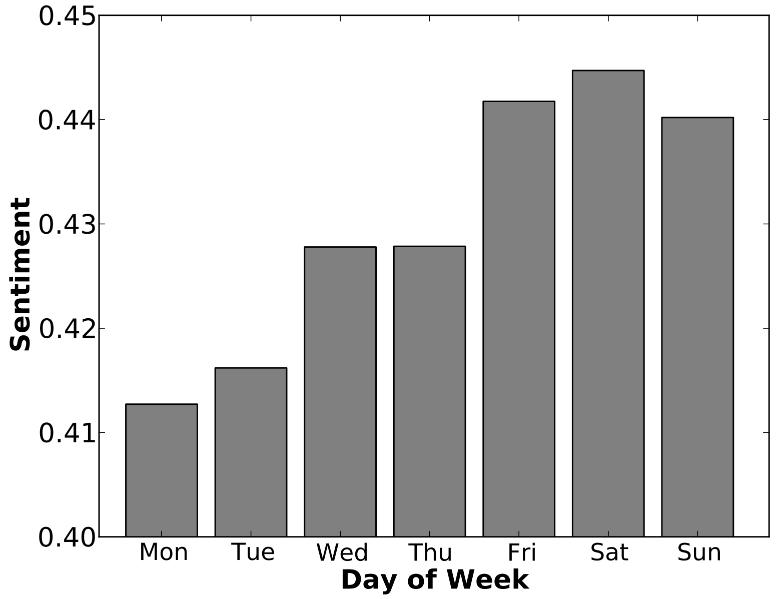


Figure A-4: Average sentiment of different days of the week in the USA, averaged across three years, from 2012 to 2014.

A.6.3 Authorial

The last contextual variable we looked at was authorial. People have different baseline attitudes, some are optimistic and positive, some are pessimistic and negative, and some are in between. This difference in personalities can manifest itself in the sentiment of tweets. We attempted to capture this difference by looking at the history of tweets made by users. The 18 million labelled tweets in our dataset come from $7,657,158$ authors.

In order to calculate a statistically significant average sentiment for each author, we need our sample size to not be too small. However, a large number of the users in our dataset only tweeted once or twice during the three years. Figure A-7 shows the number of users in bins of 50 tweets. (So the first bin corresponds to the number of users that have less than 50 tweets throughout the three year.) The number of users in the first few bins were so

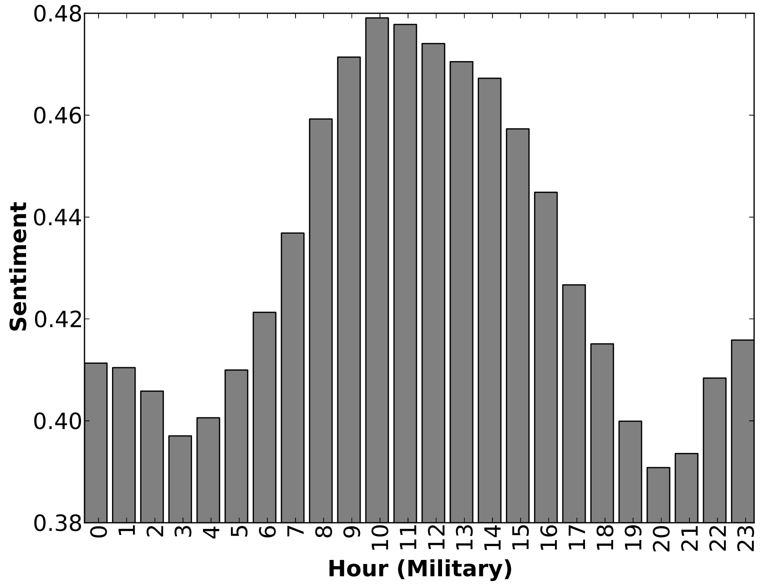


Figure A-5: Average sentiment of different hours of the day in the USA, averaged across three years, from 2012 to 2014.

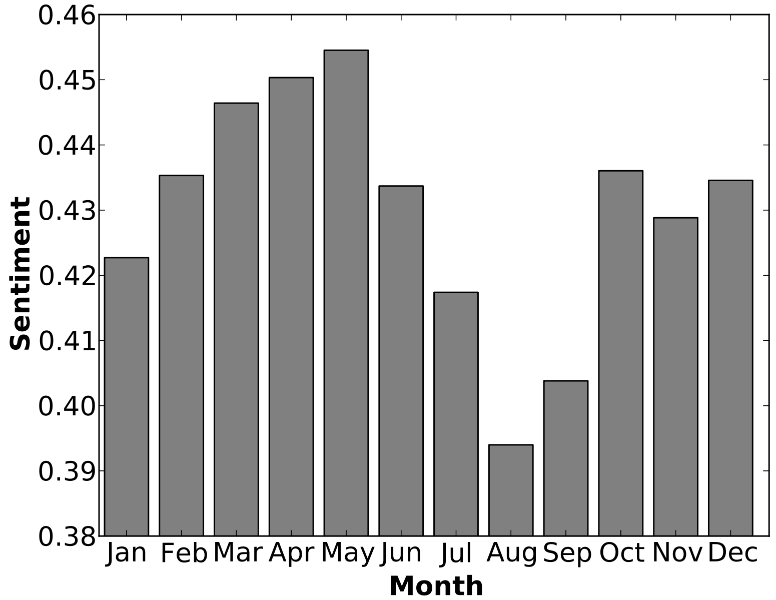


Figure A-6: Average sentiment of different months in the USA, averaged across three years, from 2012 to 2014.

large that the graph needed to be logarithmic in order to be legible. We decided to calculate the prior sentiment for users with at least 50 tweets. This corresponded to less than 1 of the users ($57,710$ out of $7,657,158$ total users). Note that these users are the most prolific authors in our dataset, as they account for 39% of all tweets in our dataset. The users with less than 50 posts had their prior set to 0.0, not favouring positive or negative sentiment (this way it does not have an impact on the Bayesian model, allowing other contextual variables to set the prior).

As it is not feasible to show the prior average sentiment of all $57,710$ users, we created 20 even sentiment bins, from $-1,0$ to 1.0. We then plotted the number of users whose average sentiment falls into these bins (figure A-8). Similar to other variables, the positive end of the graph is much heavier than the negative end.

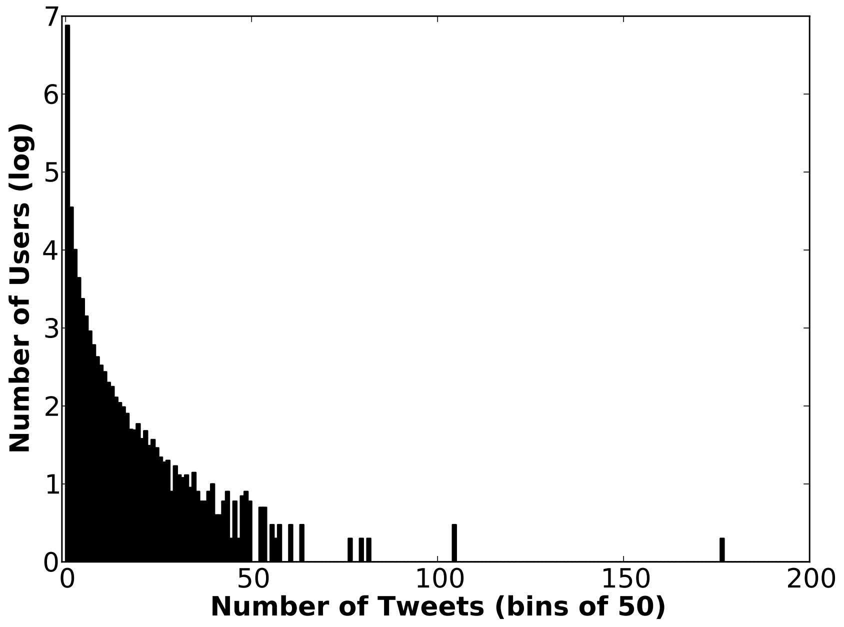


Figure A-7: Number of users (logarithmic) in bins of 50 tweets. The first bin corresponds to number of users that have less than 50 tweets throughout the three years and so on.

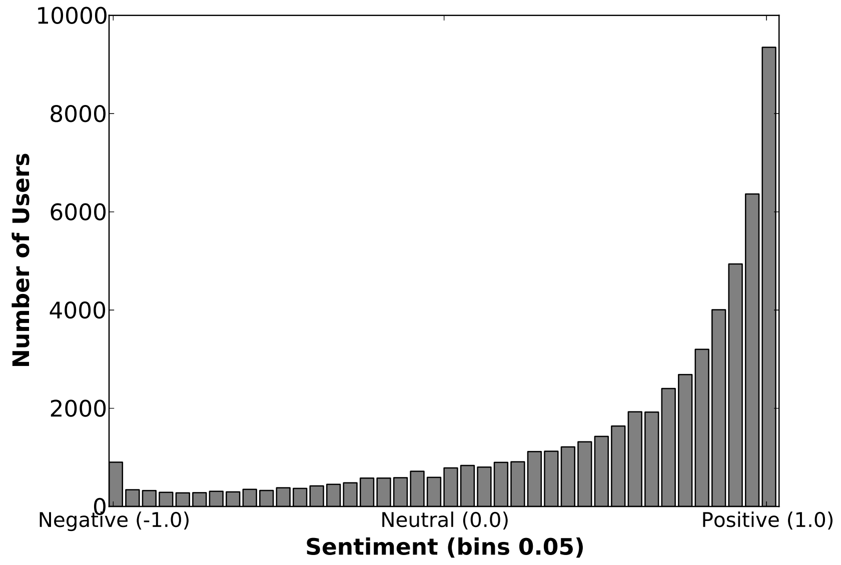


Figure A-8: Number of users (with at least 50 tweets) per sentiment bins of 0.05, averaged across three years, from 2012 to 2014.

A.7 Results

We used 5-fold cross validation to train and evaluate our baseline and contextual models, ensuring that the tweets in the training folds were not used in the calculation of any of the priors or in the training of the bigram models. Table A.3 shows the accuracy of our models. The contextual model outperformed the baseline model using any of the contextual variables by themselves, with state being the best performing and day of week the worst. The model that utilized all the contextual variables saw a 10% relative and 8% absolute improvement over the baseline bigram model.

Because of the great increase in the volume of data, distant supervised sentiment classifiers for Twitter tend to generally outperform more standard classifiers using human-labelled datasets. Therefore, it makes sense to compare the performance of our classifier to other distant supervised classifiers. Our contextual classifier outperformed the previous state-of-the-art distant supervised Twitter sentiment classifier by Go et al [34] by more than 3% (absolute).

Model	Accuracy
Baseline-Majority	0.620
Baseline-Bigram	0.785
Contextual-DoW	0.798
Contextual-Month	0.801
Contextual-Hour	0.821
Contextual-Author	0.829
Contextual-State	0.849
Contextual-All	$\mathbf{0.862}$

Table A.3: Classifier accuracy, sorted from worst to best.

Even though our contextual classifier was able to outperform the previous state-of-the-art, distant supervised sentiment classifier, it should be noted that it did so with the help of geo-tags. However, only about one to two percent of tweets in the wild are geo-tagged. Therefore, we trained and evaluated our contextual model using all the variables except for state. The accuracy of this model was 0.843, which is still slightly better than the performance of the previous state-of-the-art classifier. Fortunately, all tweets are tagged with timestamps and author information, so all the other four contextual variables used in our model can be used for classifying the sentiment of any tweet.

Note that the prior probabilities that we calculated need to be recalculated and updated every once in a while to account for changes in the world. For example, a state might become more affluent, causing its citizens to become on average happier. This change could potentially have an effect on the average sentiment expressed by the citizens of that state on Twitter, which would make our priors obsolete. Also, as mentioned earlier, our extended model assumes the contextual (and linguistic) variables are conditionally independent. This is an assumption that is most likely false but is a valid approximation. In the future, we would like to address the conditional dependence of the prior probabilities.

Bibliography

[1] P. Analytics. Twitter study–august 2009. *San Antonio, TX: Pear Analytics. Available at: www. pearanalytics. com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009. pdf*, 2009.

[2] J. Ang, Y. Liu, and E. Shriberg. Automatic dialog act segmentation and classification in multiparty meetings. In *ICASSP (1)*, pages 1061–1064, 2005.

[3] S. Aral and D. Walker. Identifying influential and susceptible members of social networks. *Science*, 337(6092):337–341, 2012.

[4] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 65–74. ACM, 2011.

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

[6] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. O'Reilly Media, Inc., 2009.

[7] J. Bollen, H. Mao, and A. Pepe. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. In *ICWSM*, 2011.

[8] P. Bordia and R. L. Rosnow. Rumor rest stops on the information highway transmission patterns in a computer-mediated rumor chain. *Human Communication Research*, 25(2):163–179, 1998.

[9] C. Burfoot and T. Baldwin. Automatic satire detection: Are you having a laugh? In *Proceedings of the ACL-IJCNLP 2009 conference short papers*, pages 161–164. Association for Computational Linguistics, 2009.

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

[11] T. P. R. Center. Internet overtakes newspapers as news outlet, December 2008. http://pewresearch.org/pubs/1066/internet-overtakes-newspapers-as-news-source[pewresearch.org; posted 23-December-2008].

[12] T. P. R. Center. Public evaluations of the news media: 1985-2009. press accuracy rating hits two decade low, September 2009. http://www.people-press.org/2009/09/13/press-accuracy-rating-hits-two-decade-low/[people-press.org; posted 13-September-2009].

[13] T. P. R. Center. Further decline in credibility ratings for most news organizations, August 2012. http://www.people-press.org/2012/08/16/further-decline-in-credibility-ratings-for-most-news-organizations/[people-press.org; posted 16-August-2012].

[14] D. Centola. The spread of behavior in an online social network experiment. *science*, 329(5996):1194–1197, 2010.

[15] E. Charniak. *Statistical language learning*. MIT press, 1996.

[16] D. Chen and C. D. Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, 2014.

[17] R. Cowan and N. Jonard. Network structure and the diffusion of knowledge. *Journal of economic Dynamics and Control*, 28(8):1557–1575, 2004.

[18] D. Crystal. Language and the internet. 2006.

[19] M. Csikszentmihalyi and J. Hunter. Happiness in everyday life: The uses of experience sampling. *Journal of Happiness Studies*, 4(2):185–199, 2003.

[20] S. Dann. Twitter content classification. *First Monday*, 15(12), 2010.

[21] D. Davidov, O. Tsur, and A. Rappoport. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 241–249. Association for Computational Linguistics, 2010.

[22] B. De Longueville, R. S. Smith, and G. Luraschi. Omg, from here, i can see the flames!: a use case of mining location based social networks to acquire spatio-temporal data on forest fires. In *Proceedings of the 2009 international workshop on location based social networks*, pages 73–80. ACM, 2009.

[23] R. Dhillon, S. Bhagat, H. Carvey, and E. Shriberg. Meeting recorder project: Dialog act labeling guide. *ICSI, Berkeley, CA, USA, Tech. Rep. TR-04-002*, 2004.

[24] N. A. Diakopoulos and D. A. Shamma. Characterizing debate performance via aggregated twitter sentiment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1195–1198. ACM, 2010.

[25] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

[26] P. Earle, M. Guy, R. Buckmaster, C. Ostrum, S. Horvath, and A. Vaughan. Omg earthquake! can twitter improve earthquake response? *Seismological Research Letters*, 81(2):246–251, 2010.

[27] B. Efron and B. Efron. *The jackknife, the bootstrap and other resampling plans*, volume 38. SIAM, 1982.

[28] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[29] E. K. Foster and R. L. Rosnow. Gossip and network relationships. *Relating difficulty: The process of constructing and managing difficult interaction*, pages 161–180, 2006.

[30] A. Friggeri, L. A. Adamic, D. Eckles, and J. Cheng. Rumor cascades. In *Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media*, 2014.

[31] M. Galley, K. McKeown, J. Hirschberg, and E. Shriberg. Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 669. Association for Computational Linguistics, 2004.

[32] Gallup-Healthways. State of american well-being. *Well-Being Index*, 2014.

[33] A. Ganesh, L. Massoulié, and D. Towsley. The effect of network topology on the spread of epidemics. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2, pages 1455–1466. IEEE, 2005.

[34] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12, 2009.

[35] S. Goel, D. J. Watts, and D. G. Goldstein. The structure of online diffusion networks. In *Proceedings of the 13th ACM conference on electronic commerce*, pages 623–638. ACM, 2012.

[36] F. E. Harrell. *Regression modeling strategies*. Springer Science & Business Media, 2001.

[37] A. L. Hughes and L. Palen. Twitter adoption and use in mass convergence and emergency events. *International Journal of Emergency Management*, 6(3):248–260, 2009.

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

[39] M. Jeong, C.-Y. Lin, and G. G. Lee. Semi-supervised speech act recognition in emails and forums. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1250–1259. Association for Computational Linguistics, 2009.

[40] L. Jiang, M. Yu, M. Zhou, X. Liu, and T. Zhao. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 151–160. Association for Computational Linguistics, 2011.

[41] F. Jin, W. Wang, L. Zhao, E. Dougherty, Y. Cao, C.-T. Lu, and N. Ramakrishnan. Misinformation propagation in the age of twitter. *Computer*, 47(12):90–94, 2014.

[42] D. Jurafsky, E. Shriberg, and D. Biasca. Switchboard swbd-damsl shallow-discourse-function annotation coders manual. *Institute of Cognitive Science Technical Report*, pages 97–102, 1997.

[43] D. Jurafsky and J. H. Martin. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall, 2nd edition, 2008.

[44] M. Karsai, G. Iñiguez, K. Kaski, and J. Kertész. Complex contagion process in spreading of online innovation. *Journal of The Royal Society Interface*, 11(101):20140694, 2014.

[45] S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(3):400–401, 1987.

[46] M. Kaufmann and J. Kalita. Syntactic normalization of twitter messages. In *International conference on natural language processing, Kharagpur, India*, 2010.

[47] K. Kireyev, L. Palen, and K. Anderson. Applications of topics models to analysis of disaster-related twitter data. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*, volume 1, 2009.

[48] L. Kong, N. Schneider, S. Swayamdipta, A. Bhatia, C. Dyer, and N. A. Smith. A dependency parser for tweets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, to appear*, 2014.

[49] E. Kouloumpis, T. Wilson, and J. Moore. Twitter sentiment analysis: The good the bad and the omg! *ICWSM*, 11:538–541, 2011.

[50] R. J. Kreuz and G. M. Caucci. Lexical influences on the perception of sarcasm. In *Proceedings of the Workshop on computational approaches to Figurative Language*, pages 1–4. Association for Computational Linguistics, 2007.

[51] L. Kundani. ”when the tail wags the dog: Dangers of crowdsourcing justice”, July 2013. http://newamericamedia.org/2013/07/when-the-tail-wags-the-dog-dangers-of-crowdsourcing-justice.php/[newamericamedia.org; posted 27-July-2013].

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

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

[54] S. Laird. "how social media is taking over the news industry", April 2012. http://mashable.com/2012/04/18/social-media-and-the-news/[mashable.com; posted 18-April-2012].

[55] V. Lampos, T. De Bie, and N. Cristianini. Flu detector-tracking epidemics on twitter. In *Machine Learning and Knowledge Discovery in Databases*, pages 599–602. Springer, 2010.

[56] D. Lee. "boston bombing: How internet detectives got it very wrong", April 2013. http://www.bbc.com/news/technology-22214511/[bbc.com; posted 19-April-2013].

[57] K. Lee, D. Palsetia, R. Narayanan, M. M. A. Patwary, A. Agrawal, and A. Choudhary. Twitter trending topic classification. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 251–258. IEEE, 2011.

[58] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[59] H. Liu and P. Singh. Conceptneta practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

[60] O. Maimon and L. Rokach. *Data mining and knowledge discovery handbook*, volume 2. Springer, 2005.

[61] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

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

[63] S. Matsumoto, H. Takamura, and M. Okumura. Sentiment classification using word sub-sequences and dependency sub-trees. In *Advances in Knowledge Discovery and Data Mining*, pages 301–311. Springer, 2005.

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

[65] G. Miller and C. Fellbaum. Wordnet: An electronic lexical database, 1998.

[66] M. Naaman, J. Boase, and C.-H. Lai. Is it really about me?: message content in social awareness streams. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 189–192. ACM, 2010.

[67] T. Nakagawa, K. Inui, and S. 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.

[68] M. E. Newman. Spread of epidemic disease on networks. *Physical review E*, 66(1):016128, 2002.

[69] N. O'Hare, M. Davy, A. Bermingham, P. Ferguson, P. Sheridan, C. Gurrin, and A. F. Smeaton. Topic-dependent sentiment analysis of financial blogs. In *Proceedings of*

the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion, pages 9–16. ACM, 2009.

[70] A. J. Oswald and S. Wu. Well-being across america. *Review of Economics and Statistics*, 93(4):1118–1134, 2011.

[71] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *HLT-NAACL*, pages 380–390, 2013.

[72] S. Owsley, S. Sood, and K. J. Hammond. Domain specific affective classification of documents. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 181–183, 2006.

[73] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*, 2010.

[74] R. Pastor-Satorras and A. Vespignani. Epidemic spreading in scale-free networks. *Physical review letters*, 86(14):3200, 2001.

[75] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[76] J. W. Pennebaker, M. R. Mehl, and K. G. Niederhoffer. Psychological aspects of natural language use: Our words, our selves. *Annual review of psychology*, 54(1):547–577, 2003.

[77] M. F. Porter. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 1980.

[78] K. Poulsen. Firsthand reports from california wildfires pour through twitter. *Retrieved February*, 15:2009, 2007.

[79] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[80] A. Rajaraman and J. D. Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

[81] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, A. Flammini, and F. Menczer. Detecting and tracking political abuse in social media. In *ICWSM*, 2011.

[82] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, S. Patil, A. Flammini, and F. Menczer. Detecting and tracking the spread of astroturf memes in microblog streams. *arXiv preprint arXiv:1011.3768*, 2010.

[83] J. Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL Student Research Workshop*, pages 43–48. Association for Computational Linguistics, 2005.

[84] R. L. Rosnow. Inside rumor: A personal journey. *American Psychologist*, 46(5):484, 1991.

[85] H. Saif, Y. He, and H. Alani. Alleviating data sparsity for twitter sentiment analysis. CEUR Workshop Proceedings (CEUR-WS. org), 2012.

[86] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[87] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49, 1978.

[88] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling. Twitterstand: news in tweets. In *Proceedings of the 17th acm sigspatial international*

conference on advances in geographic information systems, pages 42–51. ACM, 2009.

[89] J. R. Searle. *A taxonomy of illocutionary acts*. Linguistic Agency University of Trier, 1976.

[90] J. R. Searle. *Expression and meaning: Studies in the theory of speech acts*. Cambridge University Press, 1985.

[91] D. Shah and T. Zaman. Rumors in a network: Who's the culprit? *Information Theory, IEEE Transactions on*, 57(8):5163–5181, 2011.

[92] T. Shibutani. *Improvised news: A sociological study of rumor*. Ardent Media, 1966.

[93] E. Spertus. Smokey: Automatic recognition of hostile messages. In *AAAI/IAAI*, pages 1058–1065, 1997.

[94] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842. ACM, 2010.

[95] K. Starbird, L. Palen, A. L. Hughes, and S. Vieweg. Chatter on the red: what hazards threat reveals about the social life of microblogged information. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 241–250. ACM, 2010.

[96] W. Stassen. Your news in 140 characters: exploring the role of social media in journalism. *Global Media Journal-African Edition*, 4(1):116–131, 2010.

[97] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. Van Ess-Dykema, and M. Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373, 2000.

[98] P. Stone, D. C. Dunphy, M. S. Smith, and D. Ogilvie. The general inquirer: A computer approach to content analysis. *Journal of Regional Science*, 8(1):113–116, 1968.

[99] M. Valdes. ”innocents accused in online manhunt”, April 2013. http://www.3news.co.nz/Innocents-accused-in-online-manhunt/tabid/412/articleID/295143/Default.aspx/[3news.co.nz; posted 22-April-2013].

[100] S. Vieweg. Microblogged contributions to the emergency arena: Discovery, interpretation and implications. *Computer Supported Collaborative Work*, pages 515–516, 2010.

[101] S. Vieweg, A. L. Hughes, K. Starbird, and L. Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1079–1088. ACM, 2010.

[102] S. Vosoughi and D. Roy. Tweet acts: A speech act classifier for twitter. *Submited. Transactions of the Association for Computational Linguistics*, 2015.

[103] S. Vosoughi, H. Zhou, and D. Roy. Enhanced twitter sentiment classification using contextual information. *Submited. Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[104] X. Wang, F. Wei, X. Liu, M. Zhou, and M. Zhang. Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1031–1040. ACM, 2011.

[105] D. J. Watts and P. S. Dodds. Influentials, networks, and public opinion formation. *Journal of consumer research*, 34(4):441–458, 2007.

[106] A. Wierzbicka. *English speech act verbs: A semantic dictionary*. Academic Press Sydney, 1987.

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

[108] R. Zhang, D. Gao, and W. Li. What are tweeters doing: Recognizing speech acts in twitter. In *Analyzing Microtext*, 2011.

[109] R. Zhang, D. Gao, and W. 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.

[110] K. Zhao, J. Yen, G. Greer, B. Qiu, P. Mitra, and K. Portier. Finding influential users of online health communities: a new metric based on sentiment influence. *Journal of the American Medical Informatics Association*, pages amiajnl–2013, 2014.

[111] X. Zhao and J. Jiang. An empirical comparison of topics in twitter and traditional media. *Singapore Management University School of Information Systems Technical paper series. Retrieved November*, 10:2011, 2011.