Rumor Gauge: Predicting the Veracity of Rumors on Twitter

SOROUSH VOSOUGHI, MOSTAFA 'NEO' MOHSENVAND, and DEB ROY,
Massachusetts Institute of Technology

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. In this work, we developed models for automated verification of rumors (unverified information) that propagate through Twitter. To predict the veracity of rumors, we identified 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 trained and 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 about various real-world events reported on popular websites that document public rumors. The algorithm was able to correctly predict the veracity of 75% of the rumors faster than any other public source, including journalists and law enforcement officials. 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.

CCS Concepts: • **Information systems** → **Data mining**; **Information retrieval**; • **Computing methodologies** → *Information extraction*; Knowledge representation and reasoning;

Additional Key Words and Phrases: Twitter, rumor, fake news, veracity prediction, propagation

ACM Reference Format:
Soroush Vosoughi, Mostafa 'Neo' Mohsenvand, and Deb Roy. 2017. Rumor gauge: Predicting the veracity of rumors on twitter. ACM Trans. Knowl. Discov. Data 11, 4, Article 50 (July 2017), 36 pages.
DOI: http://dx.doi.org/10.1145/3070644

1. INTRODUCTION

In the last decade, the Internet has become a major player as a source of 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 United States and the second most important overall after television [Center 2008]. 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 [Java et al. 2007; Naaman et al. 2010]. Now more than ever, people turn to social media as their source of news [Laird 2012; Stassen 2010; Kwak et al. 2010]; 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

Authors' addresses: S. Vosoughi, M. Mohsenvand, and D. Roy, Laboratory for Social Machines, Massachusetts Institute of Technology, 75 Amherst St, E14-526B. Cambridge, Massachusetts 02139; emails: {soroush, mmv}@mit.edu, dkroy@media.mit.edu.

DOI: http://dx.doi.org/10.1145/3070644

85% of all *trending topics*[1] on Twitter are news [Kwak et al. 2010]. 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 [Vieweg 2010]. However, the same factors that make social media a great resource 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 crowdsourced 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 times 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 [Center 2009, 2012]. 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 area, 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 [Kundani 2013; Lee 2013; Valdes 2013]. Another example of this phenomenon is the 2010 earthquake in Chile, where rumors propagated in social media created chaos and confusion amongst the news consumers [Mendoza et al. 2010].

Motivated by these problems, we identified salient characteristics of rumors on Twitter 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 (HMMs). In this paper, we present *Rumor Gauge*, a system for automatically predicting the veracity of rumors that spread on Twitter during real-world events.

1.1. What is a Rumor?

In general, we define a rumor to be an unverified assertion that starts from one or more sources and spreads over time from node to node in a network. On Twitter specifically, a rumor is a collection of tweets, all asserting the same unverified assertion (however, the tweets could be, and almost certainly, are worded differently from each other), propagating through 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

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

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 two or more people.*
(3) *There are only two perpetrators.*
(4) *It was at least a team of five 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.)

2. APPROACH

We approached the problem of veracity prediction on Twitter as a real-time verification task. Specifically, we attempted to predict the veracity of rumors about real-world emergencies faster than any other public source. None of the previous works on veracity prediction take this approach.

The input to our system, *Rumor Gauge*, is a rumor. Recall that 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. The rumors used to train and validate *Rumor Gauge* were manually identified and annotated. The footprint of these rumors on Twitter (i.e., the tweets containing the unverified assertions) were extracted and clustered from the Twitter historical API using a semi-autonomous tools developed specifically for this purpose [Vosoughi and Roy 2015, 2016a, 2016b].

The function 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 [Bordia and Rosnow 1998; Foster and Rosnow 2006], and the study of gossips and rumors from the fields of psychology and sociology [Shibutani 1966; Rosnow 1991], 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 in later sections of this paper.

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 time-series are extracted from 209 manually annotated rumors and used to train two HMMs, one for true and one for false rumors. It is necessary to train two separate HMMs for false and true rumors since it is our hypothesis that the temporal dynamics of our features differ between false and true rumors. Each HMM captures the temporal dynamics of one type of rumor, thus allowing us to predict the veracity of new rumors by comparing their fit with respect to the false and true HMMs.

Note that the features are extracted only from data before trusted verification. By treating each feature as a time-series, we capture the "ebb and flow" of different features of rumors as they spread. This "ebb and flow," which is ignored in other works dealing with veracity prediction on social media, can reveal a lot about the nature of a rumor, specifically, it tends to be predictive of the veracity of a rumor. When a new rumor is passed to *Rumor Gauge*, the same time-series features are extracted at regular intervals as the rumor spreads. At every timestep, the temporal features are passed

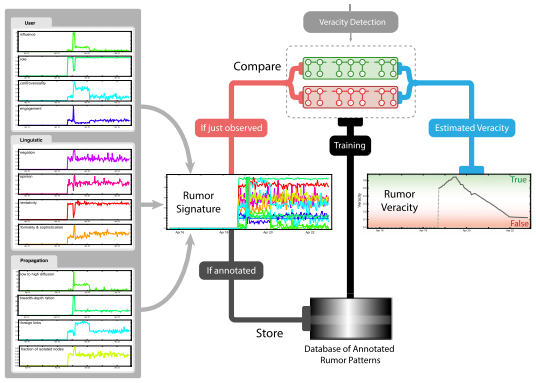

Fig. 1. An overview of the approach used for predicting the veracity of rumors. As a rumor spreads, at every timestep several time-series features are extracted from the rumor and passed to HMMs trained on false and true rumors. Each HMM measures the fitness of the data to its model and returns a probability score. These probability scores are then compared to predict the veracity of the rumor for each timestep. Over time the system generates a veracity curve.

Table I. Distribution of Manually Annotated Rumors used for Training and Evaluating the Rumor Verification System

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

to the false and true HMMs. 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 1 shows a graphical overview of our method.

3. DATA COLLECTION AND DATASETS

Our model was trained and evaluated on 209 rumors collected from several real-world events: the 2013 Boston Marathon bombings, the 2014 Ferguson unrest, and the 2014 Ebola epidemic, plus many other rumors about other events reported on Snopes.com and FactCheck.org (websites documenting rumors). These rumors were manually selected and annotated. Table I 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:

—*2013 Boston Marathon bombings*: The 2013 Boston Marathon bombings were a series of attacks and incidents that began on April 15, 2013, when two pressure cooker bombs exploded during the Boston Marathon at 2:49 pm EDT, killing three people 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 neighboring towns.[2]

[2]http://en.wikipedia.org/wiki/Boston_Marathon_bombings.

Table II. The Average Duration of Rumors (In Hours) for Each Class of Rumors, for Each Event

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

—*2014 Ferguson unrest*: The 2014 Ferguson unrest was a series of protests that began the day after Michael Brown was fatally shot by Darren Wilson, a policeman, on August 9, 2014, in Ferguson, Missouri.[3]
—*2014 Ebola epidemic*: The 2014 Ebola epidemic was the first Ebola outbreak to reach 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.[4]
—*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.[5]
—*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.[6]

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 extract tweets talking about that rumor. For this project, we were granted access to the full Twitter fire-hose, which enabled us to capture all the tweets that talked about the rumors we were interested in, enabling us to have full picture of the rumors as they spread, going back to their genesis. Additionally, the *trusted verification* time of each rumor was also manually annotated. We used sites, such as *Wikipedia*, *Snopes.com*, and *FactCheck.org,* that aggregate and cite trustworthy external sources (major governmental or news organizations) 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. Therefore, the duration of a rumor refers to the time elapsed from its very first tweet to the trusted verification time of that rumor.

The distribution of the duration of the 209 rumors can be seen in Figure 2(a). Table II shows the average duration of rumors in hours for each class of rumors, for each event. As seen with the count, false rumors on average are longer than true rumors. One possible explanation 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).

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

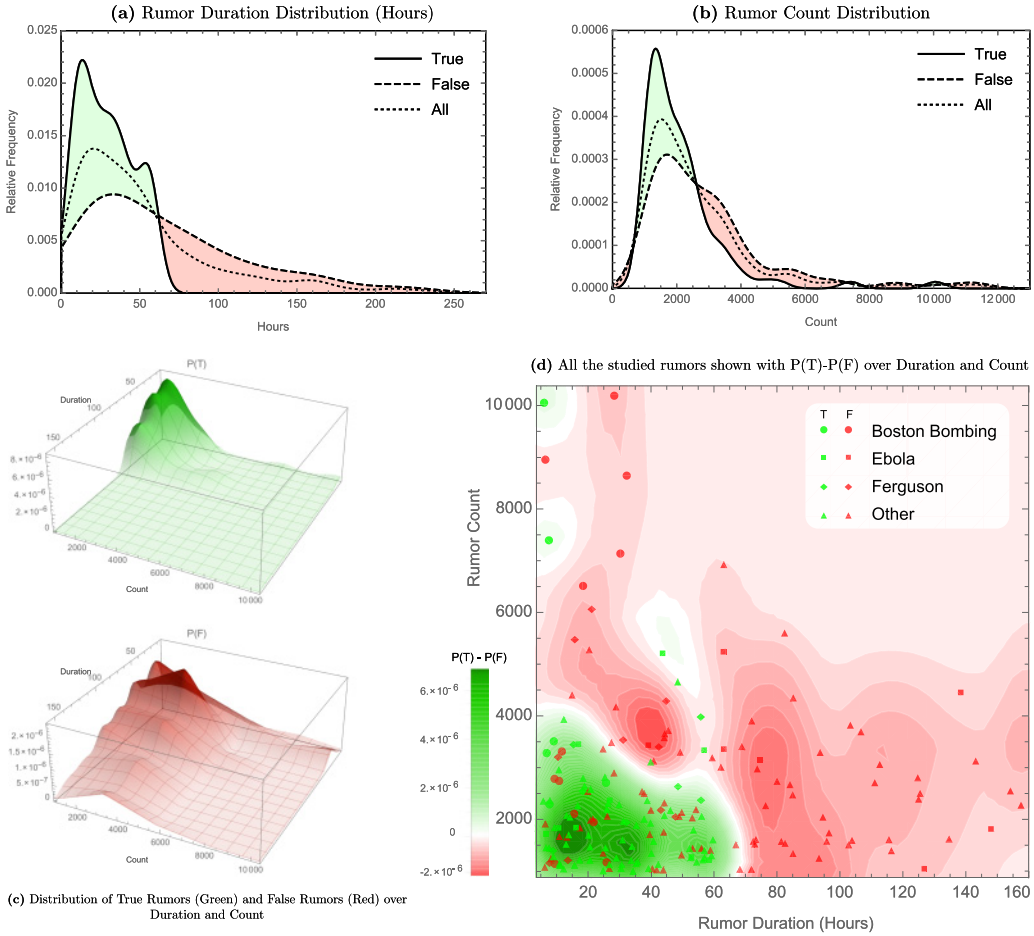

Fig. 2. Empirical kernel distribution (histogram) of 209 rumors obtained using a Gaussian kernel ($\sigma^2_{\text{Duration}} = 7, \sigma^2_{\text{Count}} = 500$). (a) False rumors tend to have a longer duration. (b) False rumors have a slightly larger count. (c) Joint KDE over count and duration. (d) Difference between joint KDEs in Figure c.

Table III. The Rounded Average Number of Tweets for Each Class of Rumors, for Each Event

Source/Event	All rumors	False rumors	True rumors
2013 Boston Marathon Bombings	9,334	11,002	4,887
2014 Ebola Pandemic	2,835	3,136	2,467
2014 Ferguson Unrest	3,011	3,274	2,635
Snopes.com & Factcheck.org	2,170	2,421	1,912
All	3,056	3,782	2,203

The count (number of tweets) and the duration (time) distribution of the 209 rumors can be seen in Figure 2(b). It was made certain that all of the rumors have at least 1,000 tweets. Any rumor that was identified with less than 1,000 tweets was discarded. Table III 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.

Figure 2(c) shows the tweet count and duration of false and true rumors in one plot. To better understand the difference between the tweet count and duration of false and true rumors, we plot their difference in Figure 2(d). It should be noted that though we only have 209 distinct rumors, these rumors contain on average 3,056 tweets and a minimum of 1,000 tweets (see Table III), enabling us to capture statistically significant differences between false and true rumors (similar to datasets used by other works that are closely related to ours, e.g., Castillo et al. [2011] and Kwon et al. [2013]).

Note that as shown in Figure 2(d), false and true rumors have generally different counts and durations across all rumors. False rumors tend to be longer and contain more tweets.

4. FEATURES

A rumor can be described as a temporal communication network, where nodes correspond to users, the edges correspond to communication between nodes, and the temporal aspect captures the propagation of messages through the network. The intuition is that there are measurable differences between the temporal communication networks corresponding to false and true rumors. In order to capture these differences, we identified 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 behavior 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 the related fields of psychology and sociology [Shibutani 1966; Rosnow 1991], meme-tracking [Leskovec et al. 2009; Ratkiewicz et al. 2011a], diffusion and virality in social networks [Matsubara et al. 2012; Friggeri et al. 2014; Goel et al. 2012], and information credibility estimation [Castillo et al. 2011; Kwon et al. 2013], we composed a list of interesting features for each of the three categories. Overall, we studied 15 linguistic, 12 user-based, and 10 propagation features, for a total of 37 features. The contribution of each of these features to the prediction of veracity was studied and not all were found to be significant. To measure the contribution of each feature, we ranked them by the probability of the *Wald chisquare* test [Harrell 2001]. 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. Measuring the statistical significance of the features is especially important since our dataset contains only 209 rumors. Table IV shows the set of all significant features determined from the chi-square test at 0.05-level($p < 0.05$).

After removing all features that did not significantly contribute to the outcome of our model, we were left with 4 linguistic, 6 user-based and 7 propagation features (see Table IV), for a total of 17 features. The contributions of each feature will be explored later in the evaluation section. Below, each of the 17 features from the three categories are explained in detail.

4.1. Linguistic

The linguistic features capture the characteristics of the text of the tweets in a rumor. A total of four linguistic features were found to significantly contribute to the outcome of our model. In the descending order of contribution these features are as follows: 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.

4.1.1. Ratio of Tweets Containing Negation. This feature measures the ratio of tweets containing negation over the total number of tweets in a rumor. Figure 3 shows two

Table IV. The Set of All Statistically Significant Features Determined Using Chi-Square Test ($p < 0.05$)

Fraction of low-to-high diffusion
Average depth-to-breadth ratio
Fraction of nodes in LCC
Ratio of tweets containing negation
User controversiality
Ratio of new users
Ratio of original tweets
User credibility
User originality
Fraction of tweets with outside links
Average formality
User influence
Ratio of Tweets containing tentatives
Ratio of tweets containing opinion
User engagement
User role
Fraction of isolated nodes

Fig. 3. Two example tweets from the same rumor, both containing assertions. Tweet (b), however, contains a negation, while tweet (a) does not.

example tweets containing references to the same rumor. The tweet shown in Figure 3(b), however, contains a negation while the tweet shown in Figure 3(a) does not.

Negations are detected using the Stanford NLP parser [Chen and Manning 2014] to generate typed dependencies of tweets. For example, the sentence, *"Sunil Tripathi is NOT the Boston Marathon suspect."* from the tweet in Figure 3(b) 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 it can be inferred that the sentence contains a negation (shown in bold red) and that it is applied to the noun *suspect*. Note that the noisy and unconventional nature of Twitter reduce the performance of standard NLP parsers, like the Stanford parser. This is an area that improved upon in the feature (for instance, by normalizing the language in tweets as was suggested by Kaufmann and Kalita [2010]).

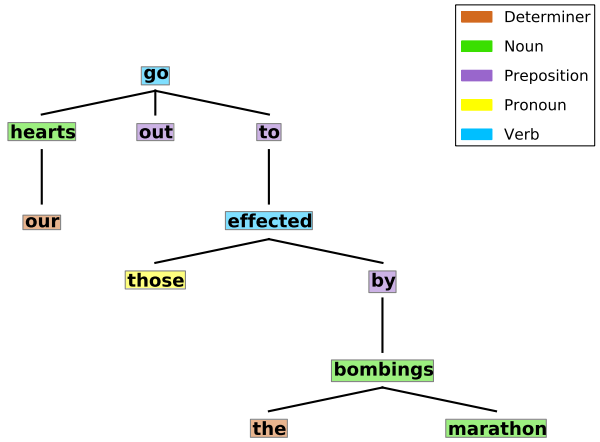

Fig. 4. The dependency tree and the part of speech tags of a sample tweet.

In addition to detecting negations using syntactic analysis, a more nuanced approach to finding negations would be to use relational semantic databases like WordNet [Miller and Fellbaum 1998] and ConceptNet [Liu and Singh 2004] to identify antonyms of terms in tweets. For example, the syntactic approach for identifying negations cannot 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.

4.1.2. 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:

—*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.

Each tweet is checked against collections of vulgar words, abbreviations, and emoticons. These collections were assembled using online dictionaries[7] (and in the case of abbreviations, also Crystal's book on language used on the internet [Crystal 2006]). There were a total of 349 vulgar words, 362 emoticons, and 944 abbreviations.

The average word complexity of a tweet is estimated by the average number of characters in each word in a tweet. 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 4.5.

The sentence complexity of a tweet is estimated by the depth of its dependency parse tree. We used Kong et al.'s [Kong et al. 2014] Twitter dependency parser for English to generate dependency trees. A sample dependency tree can be seen in Figure 4. In this example, the tweet,*"our hearts go out to those effected by the marathon bombings,"* has a depth of 5.

[7]http://www.noswearing.com/dictionary, http://pc.net/emoticons/, http://www.netlingo.com/category/acronyms.php.

4.1.3. Ratio of Tweets Containing Opinion & Insight. We collected a list of *opinion* and *insight* words from the Linguistic Inquiry and Word Count (LIWC).[8] The LIWC dictionary provides psychologically meaningful categories for the words in its collection [Pennebaker et al. 2003]. One of these categories is *opinion & insight* words. This includes words like, *know*, *consider*, *think*, and so on. Each tweet is checked against the *opinion & insight* words from LIWC.

4.1.4. Ratio of Inferring & Tentative Tweets. Another category in the LIWC is the *inferring & tentative* words. This includes words like, *perhaps*, *guess*, *maybe*, and so on. Each tweet is checked against the *inferring & tentative* words from LIWC.

4.2. User Identities

The user features capture the characteristics of the users involved in spreading a rumor. A total of six user features were found to significantly contribute to the outcome of our model. In the descending order of contribution these features are as follows: controversiality, originality, credibility, influence, role, and engagement. Below, we describe each of these features in detail.

4.2.1. Controversiality. The controversiality of a user is measured by analyzing the replies to the user's tweets. The replies to the last 1,000 tweets of a user are collected. These replies are then run through a state-of-the-art Twitter sentiment classifier [Vosoughi et al. 2015], which classifies them as either positive, negative, or neutral.

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 (1). In that equation, p is the total number of positive replies and n is the total number of negative replies. This formula was motivated by the way the *Reddit* (an online social media and discussion website) identifies controversial threads[9] based on up-votes and down-votes (Reddit's code is open sourced and freely available[10]).

$$Controversiality = (p + n)^{min(\frac{p}{n}, \frac{n}{p})}. \quad (1)$$

Figure 5 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 do not 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. To score highly on controversiality a tweet needs both. In other words, 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.

Another way to think of this is that the controversiality equation is comparing the number of paired positive and negative replies to the number of unpaired replies. A tweet with 500 positive and 50 negative replies has 50 pairs, but 450 unpaired positive replies, so its controversiality score would be low (i.e., the system would be rather confident that the tweet is not controversial). On the other hand, a post with 200 positive and 250 negative replies has 200 pairs and only 50 unpaired replies, so it would score very highly on the controversiality scale.

This method is almost the same as scoring tweets with similar number of positive and negative replies as controversial, but it also has the advantage of considering

[8]http://www.liwc.net/descriptiontable1.php.
[9]https://www.reddit.com/controversial.
[10]https://github.com/reddit/reddit.

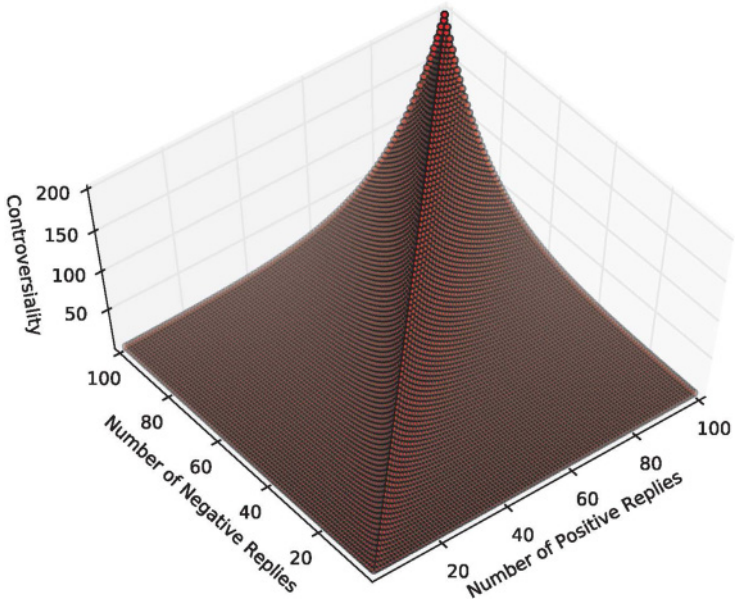

Fig. 5. Controversiality values for different number of positive and negative replies. This is an example to illustrate the distribution of controversiality scores as a function of positive and negative replies. Note that this is an example illustration so the number of positive and negative replies do not exceed 100, which is not necessarily the case for the users in our dataset.

the number of replies a tweet has received. Therefore, most controversial tweets, as measured by this method, tend to be the ones that have received a lot of replies, with almost a 50:50 split in positive and negative replies. Simply looking for a 50:50 split, without considering the total number of replies, would have meant that the most controversial tweets would be the relatively insignificant ones, with a few positive replies and equal number of negative replies (which is something we wanted to avoid).

4.2.2. Originality. Originality is a measure of how original a user's communications are on Twitter. Originality is calculated by the ratio of the number of original tweets a user has produced, to the number of times the user just retweeted 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 behavior by the user (i.e., just repeating what others say).

4.2.3. Credibility. The credibility of a user is a binary feature measured by whether the user's account has been officially verified by Twitter or not.

4.2.4. 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.

4.2.5. Role. Role measures the ratio of followers to followees of a user. A user with a high follower to followee ratio is a broadcaster. Conversely, a user with a low follower to followee ratio is a receiver.

4.2.6. Engagement. Engagement measures how active a user has been on Twitter ever since joining. Equation (2) shows how this is calculated.

$$\text{Engagement} = \frac{\#\text{Tweets} + \#\text{Retweets} + \#\text{Replies} + \#\text{Favourites}}{\text{Account Age}}. \quad (2)$$

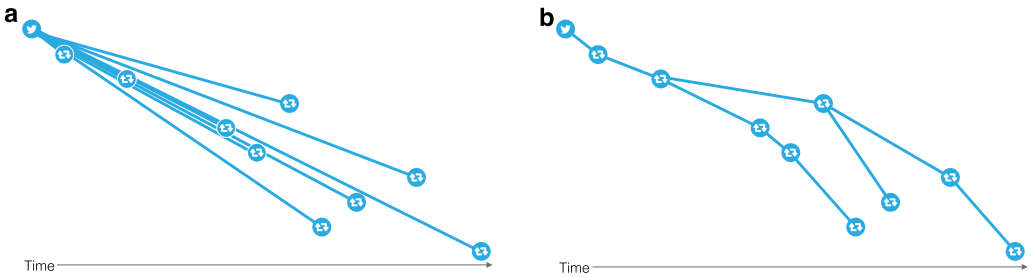

Fig. 6. (a) The retweet tree of a tweet as provided by the Twitter API. Each node represents a user and the x-axis is time. The Twitter symbol on the top left represents an original tweet and the arrows represent retweets. (b) The time-inferred retweet tree.

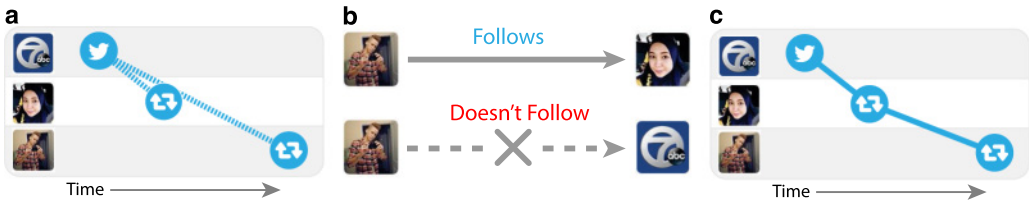

Fig. 7. Using Twitter's follower graph to infer the correct retweet path of a tweet. Panel (a) shows the retweet path provided by the Twitter API. Panel (b) shows that the bottom user is a follower of the middle user but not that of the top user (the user who tweeted the original tweet). Panel (c) shows 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.

4.3. Propagation Dynamics

The propagation features capture the temporal diffusion dynamics of a rumor. A total of seven propagation features were found to significantly contribute to the outcome of our model. In the descending order of contribution these features are as follows: 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 describe how the diffusion graph was created.

4.3.1. Time-Inferred Diffusion. A rather straight-forward way to capture the diffusion of tweets is through analyzing the retweet path of those tweet. Since each tweet and retweet is labeled 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 6(a) 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 6(a), 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 Twitter's follower graph. Figure 7 shows how this is achieved. The left panel in the figure shows the retweet path provided by the Twitter API. The middle panel shows that the bottom user is a follower of the middle user but not of the top user (the user who tweeted the original tweet). Finally, the right 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 user most likely retweeted the middle user and not the top user. If the bottom user was

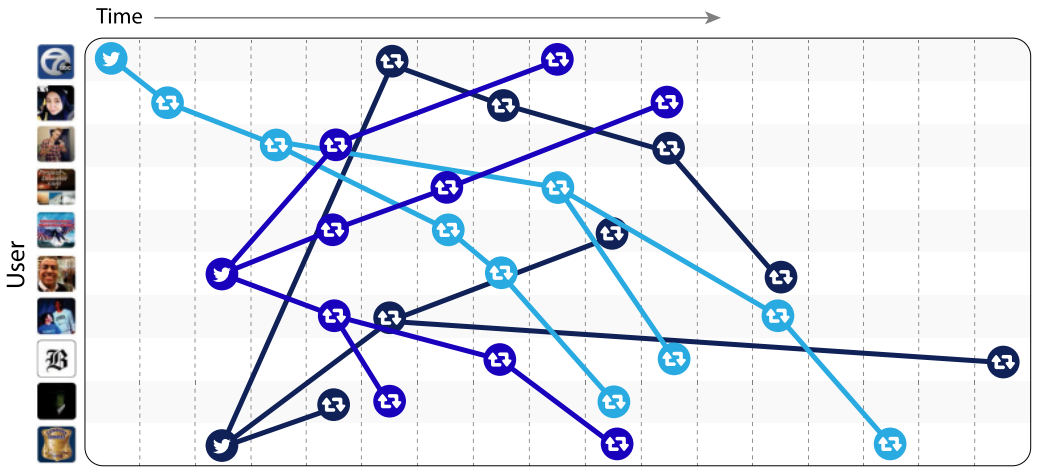

Fig. 8. A rumor is composed of many retweet trees. This figure is a simplified illustration of what the diffusion of a rumor might look like. Note that the diffusion is composed of several time-inferred diffusion trees.

a follower of the top user, then the original diffusion pattern shown in the left panel would stand (i.e., it would have been inferred that both the middle and bottom users were retweeting the top user). This method of reconstructing the true retweet graph is called time-inferred diffusion and is based on work by Goel et al. [2012].

Using this method, we can convert our example retweet tree shown in Figure 6(a) to a more accurate representation of the true retweet tree, shown in Figure 6(b). Note that a rumor is composed of many retweet trees. Figure 8 is a simplified illustration of what the diffusion of a rumor might look like. As it is shown, the diffusion is composed of several time-inferred diffusion trees. Using these time-inferred diffusion trees, we can trace a rumor's propagation through Twitter and extract informative features about the nature of the rumor's diffusion. Next, we will explain these features in detail.

4.3.2. Fraction of Low-to-High Diffusion. Each edge in the propagation graph of a rumor (such as the one shown in Figure 6(b)) 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 (3)). Influence of a user corresponds to the user's number of followers (as defined earlier). To illustrate this further, Figure 9 shows an enhanced version of the diffusion tree shown in Figure 6(b), where the size of the nodes corresponds 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).

$$\%\text{Low-High Diffusion} = \frac{\#\text{Low-high diffusions}}{\#\text{All diffusion events}}. \tag{3}$$

Figure 10 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).

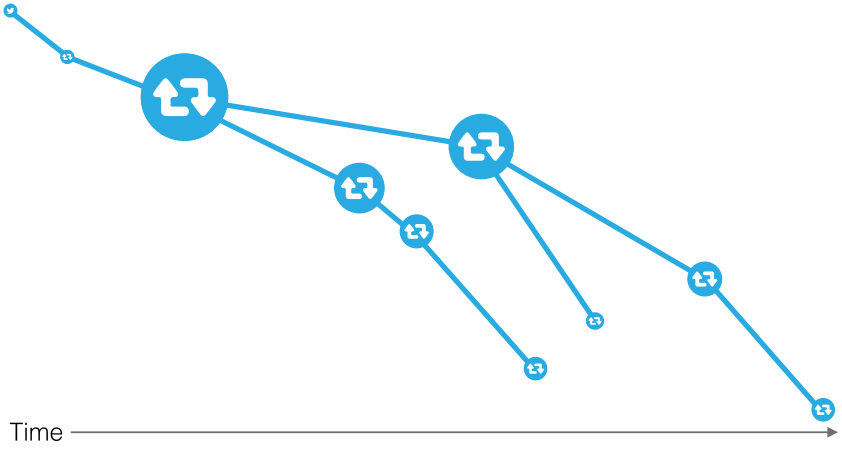

Fig. 9. This figure illustrates an enhanced version of the diffusion tree shown in Figure 6(b), where the size of the nodes corresponds to the influence of the users. Here, we can more clearly see the concept of low-to-high diffusion. For example, the diffusion between the second and third nodes (from left). As with Figure 6(b), the x-axis represents time.

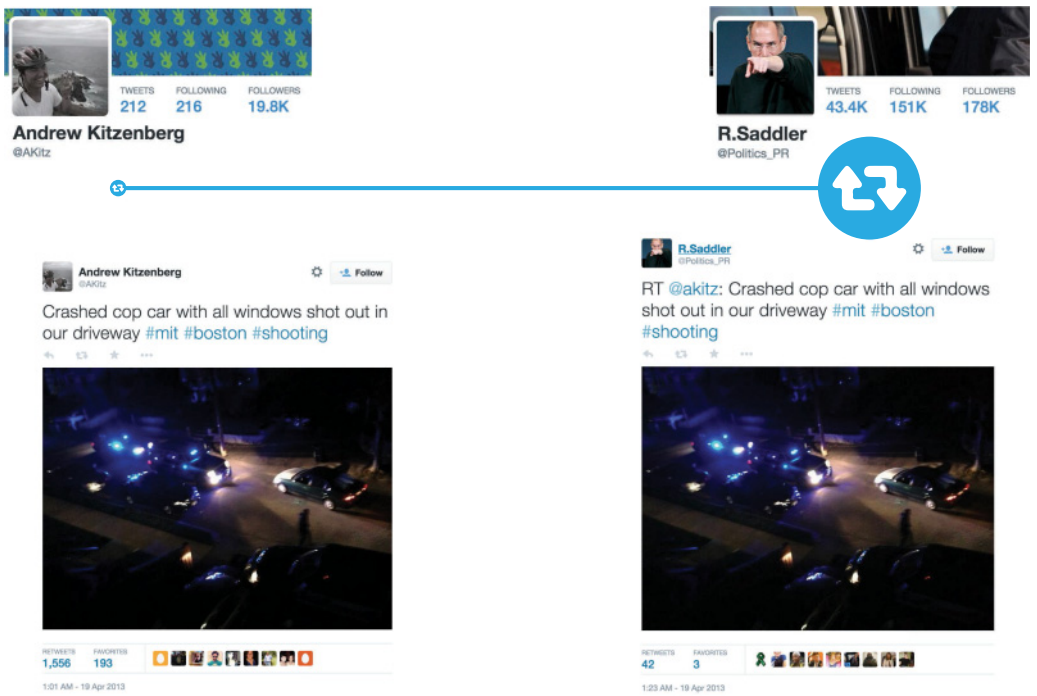

Fig. 10. This figure shows a real example of a low-to-high diffusion from the Boston Marathon bombings. As you can see, the user on the left, with 19.8K followers, was retweeted by the person on the right, with 178K followers (roughly one order of magnitude more influential than the other user).

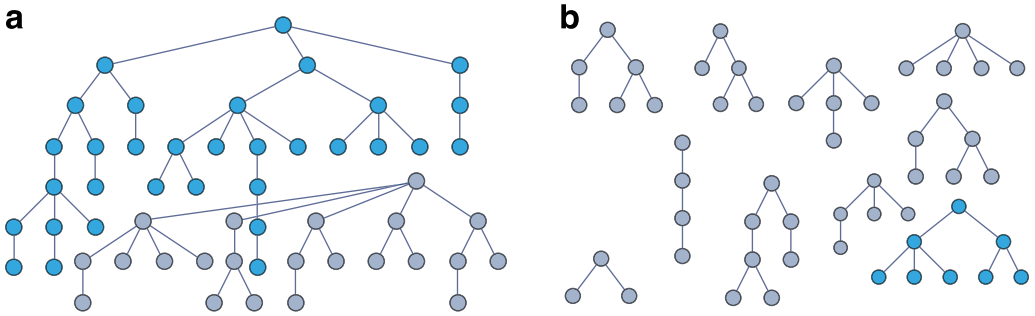

Fig. 11. Two example diffusion trees with same number of nodes, (a) with a large fraction of nodes in LCC, and (b) with a relatively lower fraction of nodes in LCC. The blue nodes represent the nodes in the LCC.

This feature is very informative as it captures the *eye-witness* phenomenon that is prevalent during real-world events (i.e., ordinary people being in the right place at the right time). It also highlights the role of Twitter as a source for breaking-news through eye-witnesses on the ground. As explained later in this paper, this feature is most predictive of the veracity of rumors compared to any other feature by itself.

4.3.3. 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 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)). This feature captures the longest conversation chain of a rumor on Twitter. To illustrate this feature, Figure 11 shows two example diffusion trees with the same number of nodes (54). The tree in Figure 11(a) has a large fraction of nodes in LCC ($\frac{32}{54} = 0.6$), and the tree in Figure 11(b) has a relatively lower fraction of nodes in LCC ($\frac{9}{54} = 0.16$).

$$\%\text{Nodes in LCC} = \frac{\#\text{Nodes in LCC}}{\#\text{All Nodes}}. \tag{4}$$

4.3.4. 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 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 8), we average the depth to breadth ratio of all the diffusion trees in a rumor. Equation (5) shows exactly how this feature is calculated for a rumor (here N refers to the number of diffusion trees in a rumor).

$$\text{Average \%depth-to-breadth} = \frac{\sum^{N} \frac{\#\text{Nodes in largest chain}}{\#\text{All Nodes}}}{N}. \tag{5}$$

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

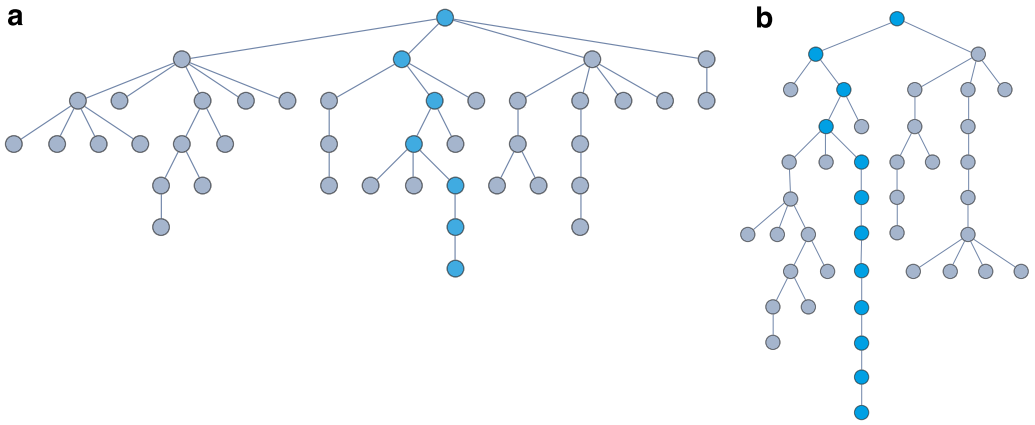

Fig. 12. Two example diffusion trees with the same number of nodes, (a) with shallow chains, and (b) with deep chains. The blue nodes represent the nodes in the deepest chain.

Fig. 13. A sample tweet about the Boston Marathon bombings with a link to an outside source (CNN).

4.3.5. 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 timestep (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 (6) shows how this feature is calculated.

$$\%\text{New Users}(t_i) = \frac{\sum \left\{\text{users}(t_i) \mid \text{users}(t_i) \notin \text{users}(t_0 \ldots t_{i-1})\right\}}{\sum \left\{\text{users}(t_i)\right\}}. \tag{6}$$

4.3.6. Ratio of Original Tweets. This is a simple measure of how captivating, engaging, and original the conversation about a rumor is. This is measured by the ratio of new tweets and replies (i.e. not retweets) in the diffusion graph of a rumor. Equation (7) shows the exact formula used to calculate this feature.

$$\%\text{Original Tweets} = \frac{\#\text{Tweets} + \#\text{Replies}}{\#\text{Tweets} + \#\text{Replies} + \#\text{Retweets}}. \tag{7}$$

4.3.7. 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 13 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

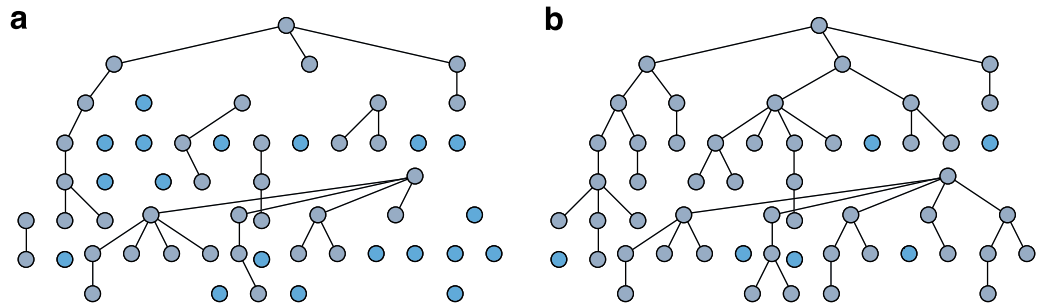

Fig. 14. Two example diffusion trees with same number of nodes, (a) with a large fraction of isolated nodes, and (b) with a relatively lower fraction of isolated nodes. The blue nodes represent the isolated nodes.

corroboration factor. Equation (8) shows how this feature is calculated.

$$\%\text{Tweet-with-URL} = \frac{\#\text{Tweets containing a URL}}{\#\text{All Tweets}}. \tag{8}$$

4.3.8. 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% of tweets never get a response (retweet or reply).[11] 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 analyzed for all the tweets contained in a rumor, it can reveal something about the nature of the rumor.

A tweet that is not retweeted or replied to shows up as an isolated node (a node 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 (9) shows exactly how this feature is calculated. To illustrate this feature, Figure 14 shows two example diffusion trees with the same number of nodes (54). The tree in Figure 14(a) has a large fraction of isolated nodes ($\frac{19}{54} = 0.35$), and the tree in Figure 14(b) has a relatively lower fraction of isolated nodes ($\frac{6}{54} = 0.11$).

$$\%\text{Isolated Nodes} = \frac{\sum \{nodes \mid degree(nodes) = 0\}}{\sum \{nodes\}}, \tag{9}$$

5. MODELS

Recall that all the features described in the last section are temporal, meaning that each feature is calculated at every timestep, 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 exceptions, it is the case that the magnitude (i.e., the values without the temporal dynamics) of our features are not very 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 as little as only a thousand responses (see Figure 2 for the distribution of the size the 209 rumors in our dataset). The temporal dynamics of our features are mostly invariant to the size of the rumors, allowing our model to generalize to events and rumors of various sizes.

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

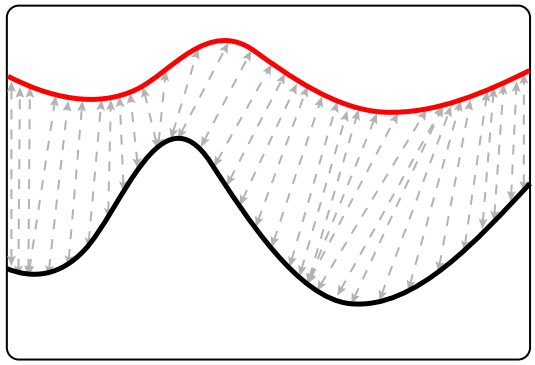

Fig. 15. An example sketch of dynamic time warping applied to two curves. The non-linear alignment aspect of DTW can clearly be seen.

Given that our features are temporal, we selected models best suited for dealing with temporal features. Moreover, the models had to be invariant to time. This is crucial since in addition to containing different volumes of tweets, rumors have varying durations, ranging from a few hours to days (see Figure 2 for the distribution of the duration 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 Markov Models (HMM). Below, we explain these models in detail.

5.1. Dynamic Time Warping

Originally developed for speech recognition, DTW is a time-series alignment algorithm. DTW can find an optimal non-linear alignment between two time-series [Sakoe and Chiba 1978]. 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 15 shows a sketch of DTW applied to two curves. The non-linear alignment aspect of DTW can be clearly seen in the figure.

The input to the DTW model is two time-series. The models return the minimum distance between the series (amongst other things). We set the cost measure used by the DTW to be the standard *L1 norm* (i.e., *Manhattan Distance*). The L1 distance between two points is the sum of the absolute differences of their Cartesian coordinates (see Equation (10)).

$$\ell_1(p,q) = ||p-q||_1 = \sum_{i=1}^{n} |p_i - q_i|. \tag{10}$$

We used DTW to measure the similarity between rumors. Since each rumor is composed of 17 features, we needed to average over the similarity between all 17 time-series in the rumors. Equation (11) 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.

$$S(R_c, R_i) = \frac{\sum_{f=1}^{17}(1 - DTW(R_c^f, R_i^f))}{17}. \tag{11}$$

For the purposes of training a classifier, DTW can be used as the distance measure for a *Nearest Neighbors (NN)* classifier [Ding et al. 2008]. Specifically, we used $S(R_c, R_i)$ shown in Equation (11) as the distance measure for a NN classifier. The similarity between an input rumor and a class of rumors is shown in Equation (12). 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 neighbors, K in a K-NN classifier. We set N to be 10.

$$S(R_i, C) = \sum_{j=1}^{N} \frac{S(R_j^c, R_i)}{N}. \tag{12}$$

Using the $S(R_i, C)$ function, the veracity of an input rumor, R_i can be calculated as shown in Equation (13).

$$Veracity = \begin{cases} True & \text{if } S(R_i, True) > S(R_i, False) \\ False & \text{if } S(R_i, True) < S(R_i, False) \\ Indecisive & Otherwise \end{cases}. \tag{13}$$

Finally, since the output of the similarity function, $S(R_i, C)$, is normalized, we can approximate the confidence of the prediction using Equation (14). 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 HMM explained in the next section provide a much more rigorous estimation of confidence.

$$Confidence = \mid S_{true} - S_{false}. \tag{14}$$

5.2. Hidden Markov Model

DTW is limited by the fact that it assumes that all of 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 paper). HMM address both of these shortcomings in DTW.

HMM are generative and probabilistic. In an HMM, a sequence of observable variables, X, is generated by a sequence of internal hidden states, Z, which cannot be directly observed. In an HMM, it is assumed that the transitions between the hidden states have the form of a Markov chain [Rabiner 1989]. 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 [Rabiner 1989]:

(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.

In our case, we started with multivariate observations for false and true rumors. We used HMMs to model the temporal dynamics of these multivariate observations. The hidden states capture the different events that drive the dynamics of the time-series (e.g., sudden influx of trustworthy sources, which would correspond to an increase in

influential and verified users). We experimented with different number of states and found that 20 states were sufficient for our purposes.

We trained two HMMs, one on observed data from false rumors and one on observed data from true rumors. We first addressed item number 3 from the list: given the observed data, estimate the model parameters. This was done using the standard, iterative *Expectation-Maximization (EM)* algorithm, also known as the *Baum–Welch* algorithm [Rabiner 1989]. 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.

In order to predict the veracity of a new rumor, we solve item number 2: given the model parameters and observed data, calculate the likelihood of the data. We calculated the likelihood of the new observed data for the false and true HMMs. This is achieved by using the standard *Forward-Backward* algorithm [Rabiner 1989]. We then compared 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 (15)).

$$Veracity = \begin{cases} True & \text{if } P(X_{new} \mid \Pi_t, A_t, \Theta_t) > P(X_{new} \mid \Pi_f, A_f, \Theta_f) \\ False & \text{if } P(X_{new} \mid \Pi_t, A_t, \Theta_t) < P(X_{new} \mid \Pi_f, A_f, \Theta_f) \\ Indecisive & Otherwise \end{cases}. \quad (15)$$

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 (16).

$$Confidence = \mid \log(P(X_{new} \mid \Pi_t, A_t, \Theta_t) - \log(P(X_{new} \mid \Pi_f, A_f, \Theta_f) \mid . \quad (16)$$

6. EVALUATION

The evaluation paradigm used for *Rumor Gauge* is shown in Figure 16. The figure shows a sample rumor diffusion in Twitter. The purple vertical lines correspond to the times at which the rumor was verified by trusted sources. Recall that trusted verification is defined to be verification by trusted channels (trustworthy major governmental or news organizations). All 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. The timestamps of the trusted verifications (e.g., dates on news articles or press releases) were used to place the verifications in the time-frame of the rumors' diffusion.

Given this framing, there were 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 shown in Figure 16).
(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 individually.
(4) Measure the accuracy of our model as a function of latency (i.e., time elapsed since the beginning of a rumor).

Given the relative sparsity of our dataset (209 annotated rumors), we used the leave-one-out evaluation method (also known as jackknife evaluation [Efron 1982]) to validate our model. At each evaluation step, a data point is held-out for testing, while

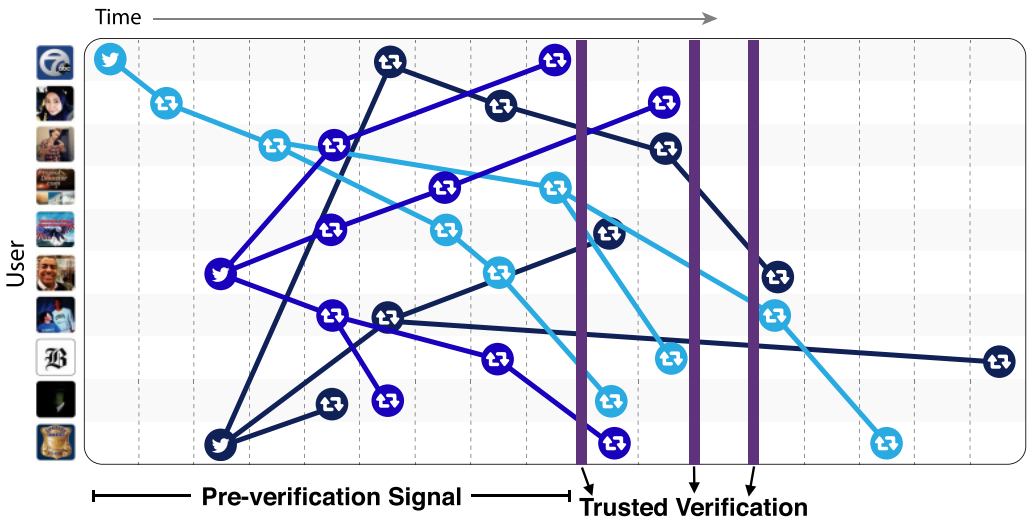

Fig. 16. The evaluation paradigm for rumor verification. This figure illustrates a sample rumor diffusion in Twitter. The purple vertical lines correspond to the times at which the rumor was verified by trusted sources. We want to evaluate how accurately our models can predict the veracity of a rumor before the first trusted verification (i.e., using the pre-verification signal).

Table V. Accuracy of *Rumor Gauge* Models and a Few Baselines

Model	All rumors	False rumors	True rumors
Majority	.54	**1.**	.0
Retweet	.56	.61	.50
N-Gram	.59	.61	.58
CAST	.64	.68	.60
SVM	.67	.69	.65
LR	.68	.69	.67
KWON	.69	.70	.69
DTW	.71	.73	.69
HMM	**.75**	**.77**	**.73**

the other data points (208 in this case) are used to train the model. 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).

6.1. Model Performance

The first evaluation task measured the overall performance of our models in comparison with certain baselines. The task with which these models were evaluated, was the correct prediction of the veracity of rumors just before the trusted verification time. Recall that out of the 209 rumors, 113(54%) were false and 96(46%) were true. Table V shows the performance of the DTW and HMM models compared to four baselines. These baselines are as follows: a majority classifier, a retweet classifier, an N-gram classifier, a classifier trained on features used by Castillo et al. [2011], called *CAST*, and a classifier trained on the features used by Kwon et al. [2013], called *KWON*. 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 1,000 most common unigrams, bigrams, and trigrams in false and true rumors. The

Table VI. Accuracy of *Rumor Gauge* using Each of the Three Feature Categories Independently

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 VII. The Performance of the Model Trained on Each of the 17 Features Individually. The First Letter Corresponds to the Type of the Feature: "P" Corresponds to Propagation Features, "U" to user Features, and "L" to Linguistic Features

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

retweet and N-Gram classifiers represent simple linguistic and propagation features and the *CAST* and *KWON* classifiers are the veracity prediction models most related to our work. We also trained two standard non-temporal models – an SVM and a logistic regression (LR) – using our features to better understand the contributions of the temporal models. These models are referred to as SVM and LR, respectively.

As it can be seen in Table V, both the DTW and HMM models greatly outperform the baseline models, with the HMM model performing the best with an overall accuracy of 0.75. Since HMM is the best performing model, from this point on, unless otherwise noted, the model being discussed is the HMM model. Note that both non-temporal models (SVM and LR) performed considerably worse than the temporal models, showcasing the advantage of using temporal models for this task. We analyze this phenomenon in more details later in this paper (Section 6.3).

Table VI 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 outperformed the linguistic and user models by a sizeable margin. Table VII shows the performance of the model trained on each of the 17 features individually. It is clear from this table that the propagation features do most of the heavy lifting, with one particular feature, the *fraction of low-to-high diffusion* contributing a great deal to the model. However, as we show in the next section, all three categories of features contribute to the performance of the model at different times.

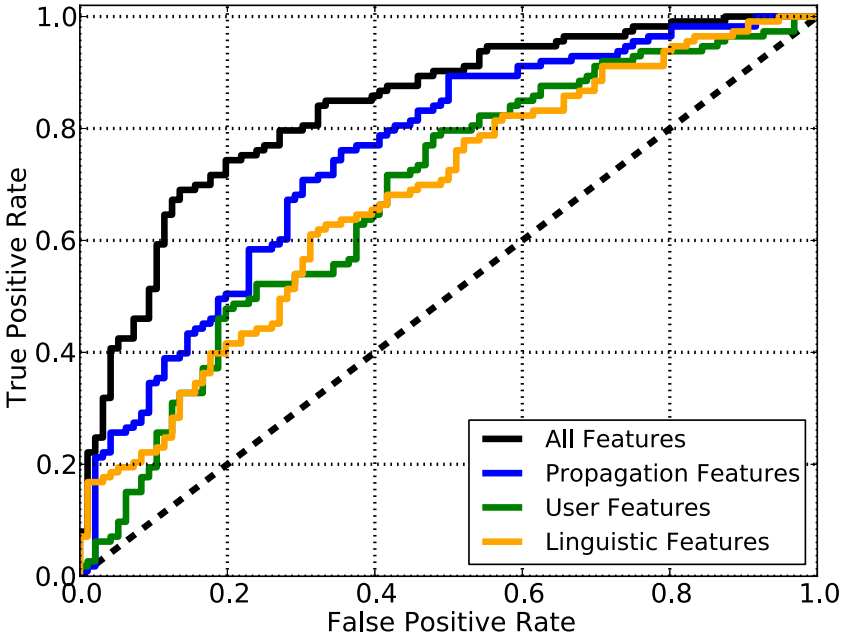

Fig. 17. ROC curves for the HMM model trained on different sets of features.

In order to get a better understanding of the performance of the model, we can look at their receiver operating characteristic (ROC) curves. Figure 17 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 points 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 or she 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.

6.2. Accuracy vs. Latency

Next, 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 was to assess how well our system would function for time-sensitive tasks. Additionally, through this evaluation, we understand 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

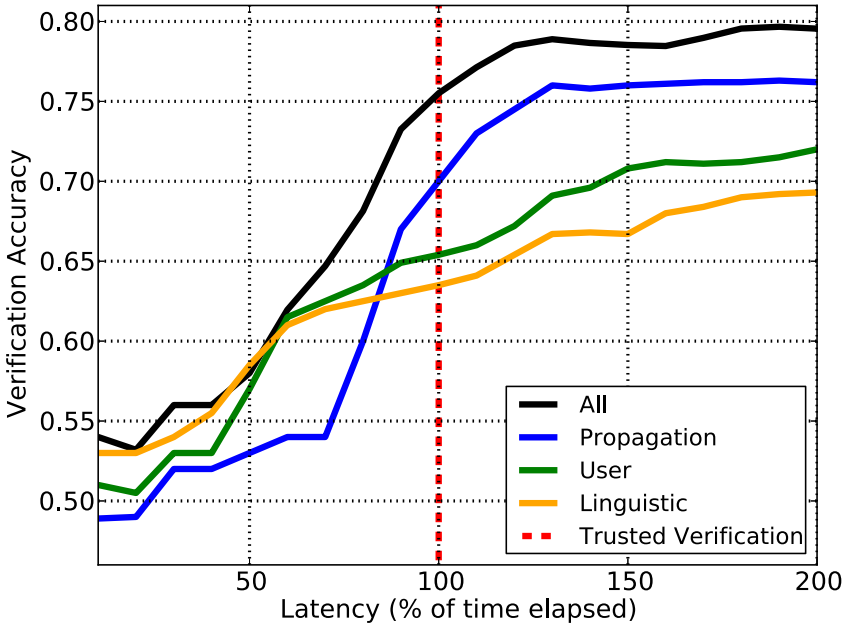

Fig. 18. Accuracy of the model as a function of latency. All 209 rumors have been aligned by using the percentage of duration instead of hours. The dashed red line represents trusted verification of the rumors.

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 trusted verification.

Figure 18 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. 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 (54%) before 50% latency. Second, the contributions of the different categories of features vary greatly over time. Different categories of features kick-in and plateau at different times. For example, the propagation features did not contribute much until around 65% latency. The early 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 only on the propagation features to be tightly correlated. This is not surprising since earlier we saw that the propagation features do most of the heavy lifting in our model (refer to Table VI).

6.3. Temporal vs. Non-Temporal Models

We showed in the evaluation section that the temporal models outperform non-temporal models; in this section, we analyze this phenomenon further. Table VIII breaks down the performance of the top performing temporal and non-temporal models (HMM and LR, respectively) by each of the three categories of features (the values for the HMMs are the same as the ones reported in Table VI. It can be seen that the model trained on propagation features took the biggest hit when switching from a

Table VIII. Accuracy of the HMM and LR Models Using Each of the Three Feature Categories Independently

Feature category	All rumors (HMM/LR)	False rumors	True rumors
Linguistic	.64/.63	.70/.71	.58/.54
User	.65/.62	.64/.62	.66/.62
Propagation	.70/.65	.72/.67	.66/.63
All	.75/.68	.77/.69	.73/.67

temporal model to a non-temporal one, dropping by 5%. The linguistic model was the least affected. This makes intuitive sense since the it is the propagation features that are most dependent on time.

For instance, while the overall fraction of low-to-high diffusion events in a cascade might be informative as a proxy for the "trustworthiness" of the information being spread (since influential people have a lot to lose if they spread untrustworthy information from someone less influential than them), it is the change in this ratio over time that best captures the response of people to the information being propagated. For example, if this ratio is high in the beginning but suddenly drops, it might signal the arrival of new information that has put the validity of the assertion being in doubt. On the other hand, while the overall fraction of low-to-high diffusion events might be lowered because of the new information, the change might be extremely small, especially if the initial response was very big.

The user features were the second most hit when switching from temporal to non-temporal models. This was also not unexpected since for several of the user features (e.g., number of influential users involved in a cascade), it is the movement of the people in and out of a cascade that seems to be informative of the trustworthiness of the information being spread. For instance, there might be a massive rush of credible (i.e., verified) users joining the cascade of a true assertion later in the game, when there is more confidence that the assertion is indeed true. Similarly, there might be a rush of credible people out of cascades of assertions that at first seem true but as time passes seem more likely to be false.

Next, we compared the accuracy of the temporal and non-temporal models as a function of latency. We discussed accuracy as a function of latency for our temporal model in detail in the last section, here we regenerated the accuracy vs. latency curves for the non-temporal models as well. Figure 19 shows these curves for the temporal and non-temporal models using all the features, the propagation features, the user features, and the linguistic features. Note that before the propagation features kick in for the temporal model, at around 70% latency, the performance of the non-temporal model tracks that of the temporal model very closely; in fact, during that period the non-temporal model performs slightly better than the temporal model. Moreover, the performance of the non-temporal model converges much faster than the temporal model, which again can be attributed mostly to the delayed effect of the propagation features in the temporal model. The non-temporal linguistic and user models track their corresponding temporal models fairly closely, with the linguistic models being the most similar.

6.4. Near Real-Time Veracity Prediction

One of the contributions of this work is the ability to do predict the veracity of rumors in time-sensitive situations. Our system is able to achieve that since the HMM model capture the temporal dynamics of different features of rumors, meaning that it is able to issue a verdict on the veracity of a rumor at any stage of the rumor as it is spreading (though we showed in the last section that the accuracy of the system increases the higher the latency). Furthermore, our system can operate in near-real time. As tweets about a particular rumor come in, they are immediately processed

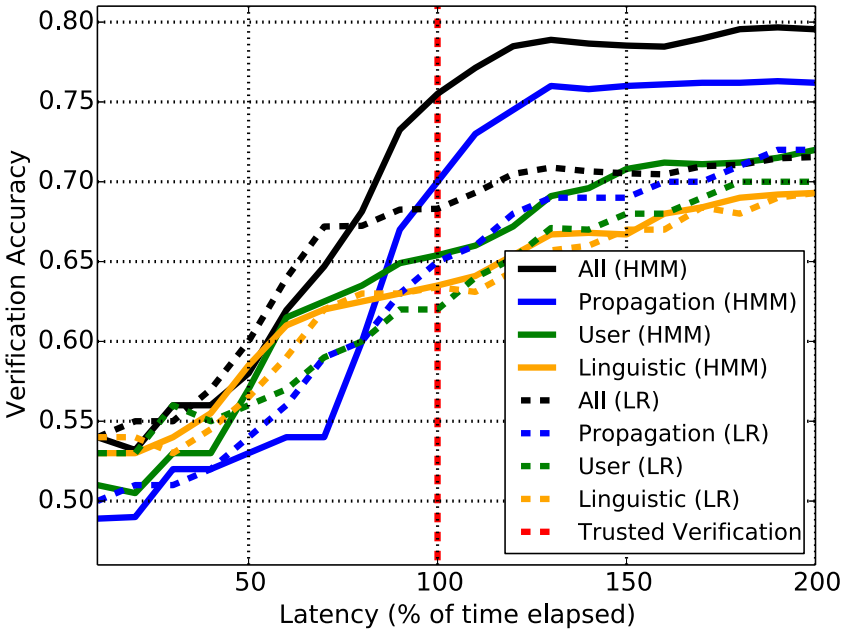

Fig. 19. Accuracy of the temporal and non-temporal models as a function of latency.

(i.e., have their features extracted). At set timesteps (e.g., every 5 minutes), the system issues a new verdict on the veracity of the rumor based on the tweets collected and processed so far. Of the three types of features, the bottleneck for processing speed are the propagation features. Specifically, because of the Twitter API restrictions, the time-inferred diffusion required for several of the propagation features is not quite real-time.

In order to test the operational speed of our system, we measured the average (median) feature extraction time for each of the three feature categories. The features extraction was done at 30 minutes time intervals using the 209 rumors in our dataset. Figure 20 shows the results. As can be seen, the majority of the 30-minute batches can be processed very quickly. Though our system is not quite real-time, it is near real-time, with the greatest latency coming from extracting the propagation features (note that since the three type of features can be extracted in parallel, the overall feature extraction time is the same as the slowest category, which is propagation). The overall median latency is around 12 minutes for analyzing the 30-minute batches. This is not really a problem since the system can simultaneously extract the features from a batch of data whilst capturing the next batch of data. Given the diversity of our dataset (our rumors from many different events), we expect the feature extraction time shown here to be representative of the speed of our system in the wild.

Since the propagation features are essential to the performance of our model and since they are most time-intensive features to calculate, below we explain exactly how these features are computed for a given rumor. This process is independent for each rumor, meaning that all rumors can be processed in parallel. As we explain below, the computation power required for computing the features is minimal, the real bottleneck, is the Twitter API limit.

As described earlier in the paper, a rumor is composed of multiple original tweets and their retweet cascades (see Figure 8). At every pre-set timestep (e.g., every 5 minutes), all tweets and retweets belonging to a rumor are captured using Twitter's streaming

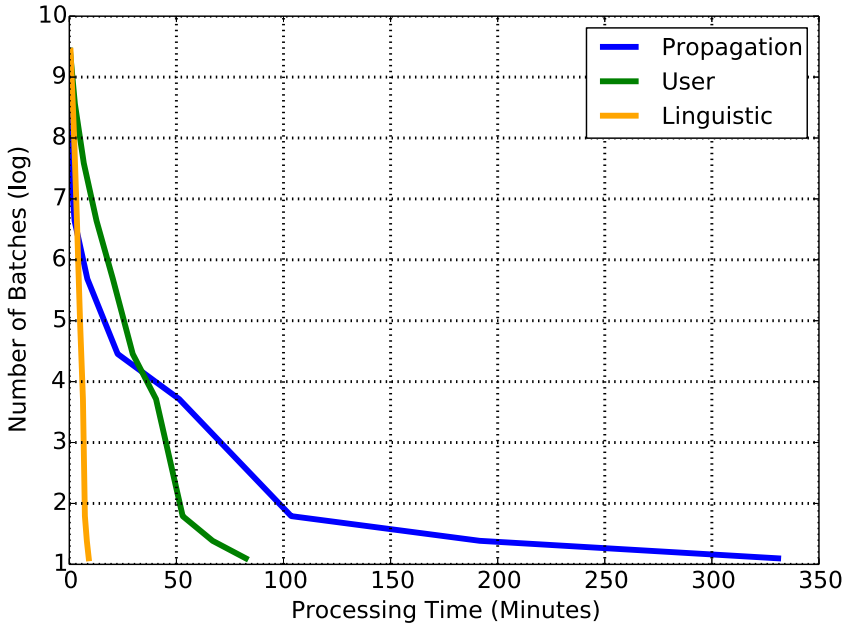

Fig. 20. The processing time of 30-minute long rumor batches for each of the three feature categories.

API. Python's NetworkX[12] is used to create a graph of the retweet cascades for the given rumor. Each cascade is represented as a tree, with the root being the original tweet. Each node represents a user and the edges represent retweet events. The nodes are attributed, meaning that they store captured and calculated information (such as, username, time of retweet, number of followers, number of followees, whether the user is verified, etc.). As we show later in this section, by saving these attributes in the nodes, the propagation features can all be computed in maximum O(N) time.

As new tweets and retweets are captured, they are added to the appropriate tree. If it is a new tweet, then it is used to create a new tree. If it is a new retweet, it is added to the diffusion tree of the original tweet that it belongs to. The linguistic and the user features of the new node are captured and calculated on the spot and added as attributes to the node (note that the linguistic features are computed for the original tweets only, since retweets have the same linguistic content as the original tweet). Next, if the new data is a retweet, its diffusion path is inferred. Finally, the propagation features are calculated. Below you can see a simplified pseudo code of the process:

Algorithm: create-or-update diffusion trees
Input: A new tweet or retweet
if new-data is tweet then
    create new tree
    add new-data as root node
    extract linguistic features and add to node as attributes
    extract user features and add to node as attributes
else if new-data is retweet then
    add new-data as node to appropriate tree

[12]https://networkx.github.io.

extract user feature and add to node as attributes
infer-diffusion of new-data
extract propagation features

The first step in computing the propagation features is to infer the diffusion paths of the retweets as they are captured. We used the *exists_friendship* function provided by Tweepy[13] (a Twitter Python API) to quickly infer the diffusion pattern using the method explained in Section 4.3.1. Below you can see the pseudo code for how this is done. Note that the running time of this algorithm is O(N) as the retweet nodes are visited maximum once.

Algorithm: Infer-diffusion
Input: A new retweet of an original tweet
for each retweet in retweets-captured
  if exists_friendship between retweet.user and new-retweet.user then
    add-edge between retweet and new-retweet
    break
append new-retweet to retweets-captured

Once the diffusion path is inferred, an edge is added between the two nodes involved in the diffusion. The diffusion trees can be updated as new information comes in and do not need to be recreated. After the trees are updated, the propagation features are calculated for that timestep. With the information saved as attributes in the nodes, all propagation features can be calculated in either O(1) (constant time) or O(N) time. Here, we go over each of the seven propagation features and explain how they can be calculated in O(1) or O(N).

—*Fraction of low-to-high diffusion*: As a new retweet node is added to a cascade tree after its diffusion path has been inferred, we compare the number of followers of the new user to the number of followers of the parent user (this information is saved as an attribute of the node) and use that to determine whether the diffusion was low-to-high or high-to-low. We keep track of the total low-to-high and high-to-low diffusions, this is used to calculate the fraction of low-to-high diffusions. The running time for this calculation (shown in Equation (3)) is O(1) as it involves a simple division.
—*Ratio of new users*: We keep a set of all unique users that have tweeted or retweeted about a rumor. As a new node is added, we use this set to determine whether the user is new (and update the set accordingly). This information is used to calculate the ratio of new users, using Equation (6). The running time for this calculation is O(N) (with N being the number of unique users in the set).
—*Ratio of new tweets*: This is calculated by comparing the number of root nodes (tweets) to all nodes (tweet and retweets). As new tweets and retweets are added to the trees, this information is updated. Calculating this ratio can thus be done in O(1), using Equation (7).
—*Fraction of tweets with outside links*: We keep track of how many original tweets contain and do not contain a URL. As a new tweet node is added, these variables are updated. With these variables, we can calculate the fraction of tweets with outside links in O(1), using Equation (8).
—*Fraction of isolated nodes*: This corresponds to the number of trees with just one node. As with previous features, we keep track of this number as new nodes are added. If a node is added to a tree with previously only one node, we reduce the number by one and if a new tree is created with just one node, we increase the number by one.

[13]www.tweepy.prg.

We also keep track of the total number of nodes. With these variables, the fraction of isolated nodes can be calculated in O(1), using Equation (9).

—*Average depth-to-breadth ratio*: As a new node is added to a cascade tree, it stores as an attribute the number of ancestors that it has. It does that by extracting the number of ancestors of its parent node (which the parent node has stored as an attribute) and adding one to that number and storing it. Since we are dealing with trees, each node can only have one parent, so the number of ancestors corresponds to the depth of the node. As the nodes are added, we also keep track of the node with the largest number of ancestors for each cascade. This number corresponds to the depth of the cascade. We also keep track of the number of nodes in a cascade as nodes are added. This way, the depth-to-breath ratio of a cascade can be calculated in O(1), using Equation (5).

—*Fraction of nodes in LCC*: The LCC of a rumor is the cascade with the largest number of nodes (or retweets). Since we keep track of the number of nodes in each cascade, the LCC can easily be identified and the fraction of nodes in LCC can be calculated using Equation (4) in O(1) time.

In brief, by updating the trees and the node attributes and several global variables in real-time as the tweets and retweets are captured, we can calculate all propagation features in maximum O(N) time (this includes the time required for inferring the diffusion paths of the retweets). Since the features for each rumor can be computed independently of other rumors, a new feature extraction process can be launched for every unique rumor that is being tracked. The only bottle-necks are memory for storing the cascades (which is minimal, so one should be able to analyze thousands of rumors in parallel) and the Twitter API limit. In practice, the most time-intensive part of calculating the propagation features is inferring the diffusion paths of retweets. As mentioned, the speed of this part is bottle-necked by the Twitter API limits (the running time of the algorithm by itself is O(N)). That is the reason why there is a drop in the processing speed of propagation features for very large cascades, as shown in Figure 20.

Two observations can be made about this, first, as Figure 20 shows, the propagation features for most rumor batches can be processed in less than an hour, with only a very small number of batches taking more than 2 hours to process, so for the vast majority of the rumors, the Twitter API limit is not an issue. Case in point, it is shown in Table III and Table II that the average number of tweets and retweets per rumor is around 3,056, with the average duration being 51.3 hours. Even taking the most extreme case in our data, the Boston Marathon bombings, the average number of tweets and retweets per rumor was 9,334, with the average duration being 17.3 hours. This is well within the Twitter API limits, which means that on average the limits should not be an issue. Second, this is not an algorithmic bottle-neck, it is a data access limit imposed by Twitter's public API. This can be resolved by either having multiple API keys or by paying for "elevated access" to the Twitter firehose[14]; this would only be necessary if planning to scale the system to analyse tens of thousands of rumors at the same time.

7. ANATOMY OF RUMORS ON 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 characteristics in the spread of false and true rumors. In addition to enabling the creation a state-of-the-art rumor verification system, these features also shed some light on the anatomy of rumors on Twitter.

[14]https://dev.twitter.com/streaming/firehose.

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). Our hypothesis is that false rumors with a high level of linguistic sophistication correspond to malicious rumors (to make the rumor look more legitimate and believable), while the ones with a low level of sophistication are mostly careless but not malicious rumors (i.e., information that has not been poorly researched). Third, not surprisingly perhaps, false rumors are more likely to be spread by users who are influential with a high controversiality score, while true rumors are more likely spread by users who are influential and credible (i.e., verified), with a low controversiality score. Finally, from the very genesis of false rumors, there tends to be people refuting these rumors (as captured by our negation detection), much more so than for true rumors. Though unfortunately, their collective voice is usually muffled by the much louder voice of the people spreading the rumor.

8. RELATED WORK

Works related to this paper include work on the role of Twitter in real-world emergencies, work from the field of network science about the diffusion and propagation of information in social networks, work on detecting suspicious behavior in online social networks, and finally, the relatively new work on veracity prediction on Twitter and other domains.

8.1. 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 news for many people [Java et al. 2007; Analytics 2009; Naaman et al. 2010]. A study by Kwak et al. [2010] showed 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 [Sakaki et al. 2010; Sankaranarayanan et al. 2009], and to track epidemics [Lampos et al. 2010]. 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 [Vieweg 2010; Vieweg et al. 2010; Starbird et al. 2010], earthquakes [Kireyev et al. 2009; Earle et al. 2010], forest fires [De Longueville et al. 2009], and hurricanes [Hughes and Palen 2009]. One particular study about wildfires in California [Poulsen 2007] outlined 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,*[15] was created by researchers for measuring the Twitter impact of real-world events.

8.2. Modeling Cascades in Networks

There has been extensive work done on modeling the spread of information in networks. The research in this area has mainly focused on modeling various diffusion and cascade

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

structures [Cowan and Jonard 2004; Goel et al. 2012], the spread of "epidemics" [Pastor-Satorras and Vespignani 2001; Newman 2002; Ganesh et al. 2005; Lampos et al. 2010; Karsai et al. 2014], knowledge [Cowan and Jonard 2004], behavior [Centola 2010], and propaganda [Ratkiewicz et al. 2011b]. Work has also been done on identifying influential players in spreading information through a network [Watts and Dodds 2007; Bakshy et al. 2011; Zhao et al. 2014; Aral and Walker 2012] and identifying sources of information [Shah and Zaman 2011].

In a work more directly related to our research direction, Mendoza et al. have looked at the difference between propagation behavior of false rumors and true news on Twitter [Mendoza et al. 2010]. Additionally, Friggeri et al. [2014] and Jin et al. [2014] have analyzed the cascade and propagation structures of rumors on social networks. Jin et al. analyzed the spread of rumors surrounding the Ebola pandemic and found that rumors can spread just like true news. Friggeri et al. provided an anatomy of rumor cascades in Facebook.

8.3. Detecting Suspicious Behavior in Online Social Networks and Services

A vast body of research exists on detecting whether a particular user behavior, such as liking, sharing, reviewing/rating, or following as well as generating targeted content is done by fake or ill-intended users. Such behaviors are generally referred to as "suspicious behaviors" [Jiang et al. 2016b] or "fake engagement" [Li et al. 2016] and the techniques used for detecting them have many aspects in common with our work. The common theme in this field of research is finding dense clusters of User-[Action,Time]→ Entity edges in the adjacency tensor of the engagement network of the users and entities (e.g., pages, tweets, users, products, etc.) (see Beutel et al. [2013] and Jiang et al. [2014, 2016a]). Generally speaking, every user interaction of this type can be specified by features of the user (including their social network), the type, content and time of the action and the type, content, and network properties of the entity. Another method, used by Li et al. [2016], tracks fake engagement activities on YouTube by analyzing the engagement behavior pattern between users and videos on YouTube. Their method looks for patterns of behavior similar to a set of known spammer seeds. Their method can be deployed using MapReduce, making it extremely fast at catching "fake users." The general insight acquired from these models is that a remarkable increase in accuracy of detection can be achieved by the combination of all features, usually summarized as content, network, and behavior. The behavior of users in our work is encapsulated and taken into account under our definition of dynamic network features where the suspiciousness of behavior has an implicit effect on our prediction of the veracity of rumor. Moreover, as Li et al. demonstrate, for real-time applications (or near real-time systems such as ours), the scalability of the approach is of paramount importance. As shown in Section 6.4, our system can run in near real-time.

8.4. Veracity Prediction on Social Media

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. A relevant early work on veracity prediction on Twitter data is due to Qazvinian et al. [2011] that ignores the identity of rumors and their temporal dependency and provides a tweet-level classifier for veracity. Most relevant are the works of Castillo et al. [2011] and Kwon et al. [2013]. Castillo et al. studied the credibility of rumors for real-world emergencies and were able to predict the veracity of rumors retrospectively (i.e., after the events), while Kwon et al. studied the propagation of urban legends (such as bigfoot) on Twitter. Both Castillo et al. and Kwon et al. proposed 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. Given the proximity of works by Castillo et al. and Kwon et al to our own, we used the models proposed by them as baselines in our evaluation. Note that all of these works focus on predicting the veracity of rumors "after the fact," whereas our system is near real-time and can be used to predict the veracity of rumors as they are spreading. Also, note that the work presented in this paper is based on and shares similarities with the first author's PhD thesis [Vosoughi 2015].

There has also been research done on verification of information on domains other than Twitter. Yang et al. [2012] and Liang et al. [2016] have both done work similar to Castillos work on Sina Weibo, Chinas leading micro-blogging service. Liang et al. have also proposed a new method for annotating collected data from Weibo automatically, allowing for creation of large datasets. The Washington Posts TruthTeller3, 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 facts.

On the whole, our work considers more features (such as the previous history and behavior of the users that are involved in spreading a rumor, and novel linguistic and propagation features) in comparison to the previous works, utilizes the temporal dynamics of these features in order to boost veracity prediction, provides an event level, near real-time, objective and collective prediction of veracity that surpasses the accuracy of other works (when evaluated on a balanced dataset). Our work also provides insights on the anatomy of rumor cascades on Twitter.

9. CONCLUSIONS AND FUTURE WORK

This paper described *Rumor Gauge*, a system for automatic verification of rumors about real-world events on Twitter. We identified salient characteristics of rumors by examining three aspects of diffusion: linguistic, the users involved, and the temporal propagation dynamics. A time-series of these features extracted for rumors was demonstrated to be predictive of the veracity of that rumor using HMM. *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 automatically predict the veracity of rumors can have real-world applications for news consumers, financial markets, journalists, and emergency services.

The work presented in this paper is different than other works in the area of social media veracity prediction in several aspects. First, we approached this problem as a real-time verification task, specifically we were interested in the prediction of the veracity of rumors about real-world emergencies, faster than any other public source. None of the previous works on veracity prediction take this approach. Moreover, we discovered several novel features that are predictive of the veracity of information on Twitter. Finally, by treating the features as time-series and using HMMs, our models take into account the "ebb and flow" of the features of rumors as they spread. This "ebb and flow," which is ignored in other works dealing with veracity prediction, is highly predictive of the veracity of rumors.

Our work can be extended in several ways. First, we would like to extend our system to cover other media platforms. As the title of the paper implies, the work presented here is focused on rumors on Twitter. However, similar techniques and algorithms could potentially be applied to other online and publicly available social media platforms (e.g., Reddit, Facebook, etc.). Though some of the features described in this paper are Twitter-specific, many of the features are platform-agnostic and can readily be extracted and processed from different platforms.

Second, our current work currently is focused on English-language tweets and rumors, it would be relatively easy to extend our work to handle rumors in other languages as well. Two of our most predictive categories of features (propagation and user characteristics) are language-agnostic. The third category of features (the linguistic features), however, needs to be reimplemented for every new language. To extract our linguistic features in a new language one needs the following for the target language: parser part-of-speech tagger; and a lexicon of vulgar, opinion, and tentative words.

Third, the current system does not differentiate between malicious and accidental misinformation. For obvious reasons, being able to distinguish between the two types of rumors would be of great use. Our preliminary studies have shown their to be distinct features that separate the two types of rumors. For instance, the sophistication of the language used in malicious misinformation is much higher than the language used in accidental misinformation.

Fourth, in addition to predicting the veracity of rumors, we would like 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 especially relevant for emergency services who might want to respond to false rumors that might have a large negative impact.

Fifth, the rumors used in this study were obtained by a semi-autonomous tool that utilizes manual annotation as well as hierarchical clustering [Vosoughi and Roy 2015]. As shown by Zhao et al. [2015], early detection of rumors can be enhanced and automated by detecting clusters of trending topics that occasionally express disputed claims and skepticism through signature phrases such as "Is this true?," "Really?," "What?," and the like. Moreover, progress in automatic early detection of rumors can help with integration of other real-time platforms such as Reddit, Facebook, and Yik Yak.

Finally, a natural extension of our work is to develop a strategy that can be used to dampen the effects of false rumors. This would be of great use to the emergency services dealing with real-world emergencies as they are the ones that usually have to deal with the consequences and the fallout of rumors on social media. For example, in the case of Boston Marathon bombings, there were several unfortunate instances of innocent people being implicated in witch-hunts [Kundani 2013; Lee 2013; Valdes 2013], one of which became the Boston Marathon bombings' 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 be used to 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.

We plan to release our dataset to other researchers working on this area. If interested, please contact the corresponding author.

ACKNOWLEDGMENTS

The authors would like to thank Helen Zhou for her help with the preparation of this document. Also, thanks go to Dr Allen Gorin and Prof. Sinan Aral for their invaluable advice and guidance on this project.

REFERENCES

Pear Analytics. 2009. Twitter Study–August 2009. Available: https://pearanalytics.com/wp-content/uploads/2009/08/Twitter-Study-August-2009.pdf. Accessed 2015 March 13.

Sinan Aral and Dylan Walker. 2012. Identifying influential and susceptible members of social networks. *Science* 337, 6092 (2012), 337–341.

Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. 2011. Everyone's an influencer: Quantifying influence on Twitter. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*. ACM, 65–74.

Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. Copycatch: Stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 119–130.

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

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on Twitter. In *Proceedings of the 20th International Conference on World Wide Web*. ACM, 675–684.

Damon Centola. 2010. The spread of behavior in an online social network experiment. *Science* 329, 5996 (2010), 1194–1197.

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

Robin Cowan and Nicolas Jonard. 2004. Network structure and the diffusion of knowledge. *Journal of economic Dynamics and Control* 28, 8 (2004), 1557–1575.

David Crystal. 2006. Language and the Internet (2nd). Cambridge: Cambridge University Press.

Bertrand De Longueville, Robin S. Smith, and Gianluca Luraschi. 2009. 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*. ACM, 73–80.

Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. 2008. Querying and mining of time series data Experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment* 1, 2 (2008), 1542–1552.

Paul Earle, Michelle Guy, Richard Buckmaster, Chris Ostrum, Scott Horvath, and Amy Vaughan. 2010. OMG earthquake! Can Twitter improve earthquake response? *Seismological Research Letters* 81, 2 (2010), 246–251.

Bradley Efron. 1982. The Jackknife, the Bootstrap, and Other Resampling Plans. (SIAM Monograph #38) Philadelphia: Society for Industrial and Applied Mathematics.

Eric K. Foster and Ralph L. Rosnow. 2006. Gossip and network relationships. *Relating Difficulty: The Process of Constructing and Managing Difficult Interaction* (2006), 161–180.

Adrien Friggeri, Lada A. Adamic, Dean Eckles, and Justin Cheng. 2014. Rumor cascades. In *Proceedings of the 8th International AAAI Conference on Weblogs and Social Media*.

Ayalvadi Ganesh, Laurent Massoulié, and Don Towsley. 2005. The effect of network topology on the spread of epidemics. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2005*, Vol. 2. IEEE, 1455–1466.

Sharad Goel, Duncan J. Watts, and Daniel G. Goldstein. 2012. The structure of online diffusion networks. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. ACM, 623–638.

Frank E. Harrell. 2001. *Regression Modeling Strategies*. Springer Science & Business Media.

Amanda Lee Hughes and Leysia Palen. 2009. Twitter adoption and use in mass convergence and emergency events. *International Journal of Emergency Management* 6, 3 (2009), 248–260.

Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. 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*. ACM, 56–65.

Meng Jiang, Alex Beutel, Peng Cui, Bryan Hooi, Shiqiang Yang, and Christos Faloutsos. 2016a. Spotting suspicious behaviors in multimodal data: A general metric and algorithms. *IEEE Transactions on Knowledge and Data Engineering* 28, 8 (2016), 2187–2200.

Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Catchsync: Catching synchronized behavior in large directed graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 941–950.

Meng Jiang, Peng Cui, and Christos Faloutsos. 2016b. Suspicious behavior detection: Current trends and future directions. *IEEE Intelligent Systems* 31, 1 (2016), 31–39.

Fang Jin, Wei Wang, Liang Zhao, Edward Dougherty, Yang Cao, Chang-Tien Lu, and Naren Ramakrishnan. 2014. Misinformation propagation in the age of Twitter. *Computer* 47, 12 (2014), 90–94.

Márton Karsai, Gerardo Iñiguez, Kimmo Kaski, and János Kertész. 2014. Complex contagion process in spreading of online innovation. *Journal of The Royal Society Interface* 11, 101 (2014), 20140694.

Max Kaufmann and Jugal Kalita. 2010. Syntactic normalization of Twitter messages. In *Proceedings of the International Conference on Natural Language Processing*. Kharagpur, India.

Kirill Kireyev, Leysia Palen, and K. Anderson. 2009. Applications of topics models to analysis of disaster-related Twitter data. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*. Amherst, MA.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. 2014. A dependency parser for tweets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. ACL, 1001–1012.

Lalit Kundani. 2013. When the Tail Wags the Dog: Dangers of Crowdsourcing Justice. Retrieved from http://newamericamedia.org/2013/07/when-the-tail-wags-the-dog-dangers-of-crowdsourcing-justice.php/.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is Twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 591–600.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *Proceedings of the 13th International Conference on Data Mining (ICDM)*. IEEE, 1103–1108.

Sam Laird. 2012. "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].

Vasileios Lampos, Tijl De Bie, and Nello Cristianini. 2010. Flu detector-tracking epidemics on Twitter. In *Machine Learning and Knowledge Discovery in Databases*. Springer, 599–602.

Dave Lee. 2013. Boston bombing: How internet detectives got it very wrong. Retrieved from http://www.bbc.com/news/technology-22214511/.

Jure Leskovec, Lars Backstrom, and Jon Kleinberg. 2009. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 497–506.

Yixuan Li, Oscar Martinez, Xing Chen, Yi Li, and John E. Hopcraft. 2016. In a world that counts: Clustering and detecting fake social engagement at scale. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 111–120.

Gang Liang, Jin Yang, and Chun Xu. 2016. Automatic rumors identification on Sina Weibo. In *Proceedings of the12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD'16)*. IEEE, 1523–1531.

Hugo Liu and Push Singh. 2004. ConceptNeta practical commonsense reasoning tool-kit. *BT Technology Journal* 22, 4 (2004), 211–226.

Yasuko Matsubara, Yasushi Sakurai, B. Aditya Prakash, Lei Li, and Christos Faloutsos. 2012. Rise and fall patterns of information diffusion: Model and implications. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 6–14.

Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. 2010. Twitter under crisis: Can we trust what we RT? In *Proceedings of the 1st Workshop on Social Media Analytics*. ACM, 71–79.

George Miller and Christiane Fellbaum. 1998. Wordnet: An electronic lexical database. (1998).

Mor Naaman, Jeffrey Boase, and Chih-Hui Lai. 2010. Is it really about me? Message content in social awareness streams. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*. ACM, 189–192.

Mark E. J. Newman. 2002. Spread of epidemic disease on networks. *Physical review E* 66, 1 (2002), 016128.

Romualdo Pastor-Satorras and Alessandro Vespignani. 2001. Epidemic spreading in scale-free networks. *Physical Review Letters* 86, 14 (2001), 3200.

James W. Pennebaker, Matthias R. Mehl, and Kate G. Niederhoffer. 2003. Psychological aspects of natural language use: Our words, our selves. *Annual Review of Psychology* 54, 1 (2003), 547–577.

The Pew Research Center. 2008. Internet Overtakes Newspapers As News Outlet. (December 2008). http://pewresearch.org/pubs/1066/internet-overtakes-newspapers-as-news-source[pewresearch.org; pos ted 23-December-2008].

The Pew Research Center. 2009. Public Evaluations of the News Media: 1985-2009. Press Accuracy Rating Hits Two Decade Low. Retrieved from http://www.people-press.org/2009/09/13/press-accuracy-rating-hits-two-decade-low/.

The Pew Research Center. 2012. Further Decline in Credibility Ratings for Most News Organizations. Retrieved from http://www.people-press.org/2012/08/16/further-decline-in-credibility-ratings-for-most-news-organizations/.

Kevin Poulsen. 2007. Firsthand reports from California wildfires pour through Twitter. Available: www.wired.com/threatlevel/2007/10/firsthand. Accessed 2009 Feburary 15.

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

Lawrence Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77, 2 (1989), 257–286.

Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer. 2011a. Detecting and tracking political abuse in social media. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media (ICWSM'11)*. AAAI, 297–304.

Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Goncalves, Snehal Patil, Alessandro Flammini, and Filippo Menczer. 2011b. Detecting and tracking the spread of astroturf memes in microblog streams. In *Proceedings of the 20th International Conference Companion on World Wide Web*. ACM, 249–252.

Ralph L. Rosnow. 1991. Inside rumor: A personal journey. *American Psychologist* 46, 5 (1991), 484.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 851–860.

Hiroaki Sakoe and Seibi Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing* 26, 1 (1978), 43–49.

Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. 2009. Twitterstand: News in tweets. In *Proceedings of the 17th ACM Sigspatial International Conference on Advances in Geographic Information Systems*. ACM, 42–51.

Devavrat Shah and Tauhid Zaman. 2011. Rumors in a network: Who's the culprit? *IEEE Transactions on Information Theory* 57, 8 (2011), 5163–5181.

Tamotsu Shibutani. 1966. *Improvised News: A Sociological Study of Rumor*. Ardent Media.

Kate Starbird, Leysia Palen, Amanda L. Hughes, and Sarah Vieweg. 2010. 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*. ACM, 241–250.

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

Manuel Valdes. 2013. Innocents accused in online manhunt. Retieved from http://www.3news.co.nz/Innocents-accused-in-online-manhunt/tabid/412/articleID/295143/Default.aspx/.

Sarah Vieweg. 2010. Microblogged contributions to the emergency arena: Discovery, interpretation and implications. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*. ACM, 241–250.

Sarah Vieweg, Amanda L. Hughes, Kate Starbird, and Leysia Palen. 2010. 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*. ACM, 1079–1088.

Soroush Vosoughi. 2015. *Automatic detection and verification of rumors on Twitter*. Ph.D. Dissertation. Massachusetts Institute of Technology.

Soroush Vosoughi and Deb Roy. 2015. A human-machine collaborative system for identifying rumors on Twitter. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW'15)*. IEEE, 47–50.

Soroush Vosoughi and Deb Roy. 2016a. A semi-automatic method for efficient detection of stories on social media. In *Proceedings of the10th International AAAI Conference on Web and Social Media*. AAAI, 707–710.

Soroush Vosoughi and Deb Roy. 2016b. Tweet acts: A speech act classifier for Twitter. In *Proceedings of the10th International AAAI Conference on Web and Social Media*. AAAI, 711–714.

Soroush Vosoughi, Helen Zhou, and Deb Roy. 2015. Enhanced Twitter sentiment classification using contextual information. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Association for Computational Linguistics, 16–24. http://aclweb.org/anthology/W15-2904.

Duncan J. Watts and Peter Sheridan Dodds. 2007. Influentials, networks, and public opinion formation. *Journal of consumer research* 34, 4 (2007), 441–458.

Kang Zhao, John Yen, Greta Greer, Baojun Qiu, Prasenjit Mitra, and Kenneth Portier. 2014. Finding influential users of online health communities: A new metric based on sentiment influence. *Journal of the American Medical Informatics Association (JAMIA)* 21, e2 (2014), e212–e218.

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1395–1405.

Received November 2015; revised October 2016; accepted March 2017