Early Detection of Rumors in Twitter

CS 585 UMass Amherst Fall 2015

Anushree Ghosh
Rose Tharail John
Yamin Thuzar Tun

ABSTRACT

We are building on the paper “Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts” by Zhe Zhao, Paul Resnick, Qiaozhu Mei [1]. The goal is to perform Early Rumor detection using tweets. Our system presents a technique to identify trending rumors, which we define as topics that include disputed factual claims. Previous works on this problem, emphasize on using skepticism and questioning nature of the rumor-containing tweets. We imbibe this feature in our algorithm by using signature text phrases that are used to express skepticism about factual claims, as a signal for tweets that are discussing the rumor. But, beyond this, the key concept proposed by our system is the analysis of a **cluster** of tweets rather than individual tweets for detecting rumors.

INTRODUCTION

Before our rumor detection process, it is essential to define what rumor tweets are. Rumor in this problem is defined as a controversial and fact-checkable statement. Particularly in social media like twitter a rumor could be controversial statement that is widely discussed and retweeted. The controversy of rumor topic is important since our rumor detection model mainly relies on finding the public’s skeptical response to the rumor, instead of analyzing the content of the rumor topic itself inside each tweet. Whether a rumor statement is true or not should also be fact-checkable at the current moment by anyone with sufficient access to relevant evidence. This excludes the tweets that discuss incidents that will happen in the future since they are not fact-checkable. [1]

There are several famous incidents of rumors spreading in Twitter and causing confusion and sometimes catastrophic consequences. For example, after Boston marathon bombing, there were several rumors spreading on Twitter, stirring up anxiety on Twitter and causing the stock market to collapse for a short while. Such an aftermath is definitely preventable if, we detect rumours early and resolve them as necessary.

We discovered in our initial attempt with some baseline model assumption that it is extremely difficult to accurately classify whether or not every individual post is a rumor. But, we are able to identify trending rumors by recasting the problem as finding entire clusters of posts whose topic is a disputed factual claim.

The paper[1] has developed a technique based on searching for the enquiry phrases, clustering similar posts together, and then collecting related posts that do not contain these simple phrases. They then rank the clusters by their likelihood of really containing a disputed factual claim. The

detector, which searches for the very rare, but very informative phrases, combined with **clustering and classification of the clusters**, yields better results than the baseline model. The confidence of the predictions for top 10 clusters, which we report should motivate a curious human analyst to sift through the tweets and arrive at an informed decision on the topics.

RELATED WORK

Even though rumors have long been a subject in multiple disciplines, research on identifying rumors from online social media through computational methods has only begun in recent years. Previous work has shown that particular known rumors can be retrieved with a high accuracy by training a machine learning classifier for each rumor [8]. Here we seek to identify new rumors, not necessarily retrieve all the tweets related to them. Much previous research has tried to develop classifiers for a more challenging problem than ours, automatically determining whether a meme that is spreading is true or false ([2, 3, 4, 5]).

Many studies identified the popularity of a post (e.g., number of posts that retweeted or replied to the post) as a significant signal of a rumor. This information is used either directly as features of the "rumor" classifier [3, 4, 2, 5, 6, 7], or as filters to prescreen candidate topics (e.g., to only consider the most popular posts [9] or "trending topics"), or both [3, 4]. In fact, we used it as a feature in our baseline model.

Other work identified burstiness [7], temporal patterns [5, 9], or the network structure of the diffusion of a post/topic [10, 3, 11, 5] as important signals.

Our models focused significantly on NLP related measures like - appearance of signature question phrases, tweet popularity, word frequency distribution and statistical similarity measures to arrive at results.

DATA

We have 2 datasets which we acquired from the authors of the papers.

1) Prof. Dragomir R. Radev of [4] provided 5 separate tweet sets, each of which includes 5 sets of annotated tweets from 5 different stories.
2) Zhe Zhao of [1] provided us a list of tweet id's and annotated cluster id's related to the Boston Marathon bombing, which their paper Enquiring Minds is based on.

<u>For the baseline model, we used the first dataset (1)</u>

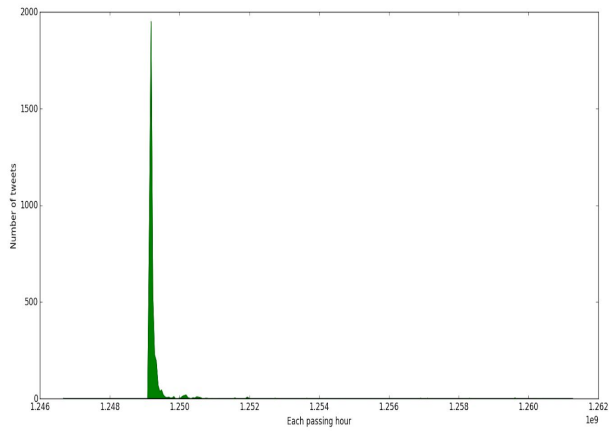
Out of 5 data sets, 3 datasets contain labelled tweet contents while 2 other datasets consists of only tweet ids. We fetched the tweets from the ids using the Twitter APIs. To have a balanced dataset, we also fetched a bunch of tweets we believe are non-rumors, including some from Prof. Brendan's timeline!

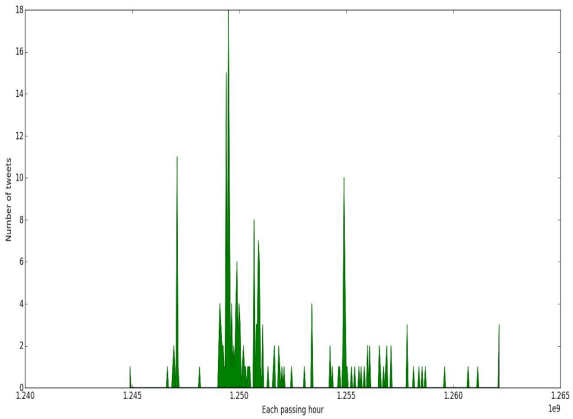
	Filename	Context	Num of tweets	Rumor Count Ratio
Test Set	airfrance.txt	Air France mid-air crash photos?	505	0.394
	michelle.txt	Michelle Obama hired too many staff?	299	0.722
	obama.txt	Is Barack Obama muslim?	105	0.685
	cell.txt	Cell phone numbers going public?	122	0.508
Training Set	palin.txt	Sarah Palin getting divorced?	4423	0.981
	boston.txt	Boston	~2000	0.99
		Non-rumorous tweets fetched from various topics	1200	N/A

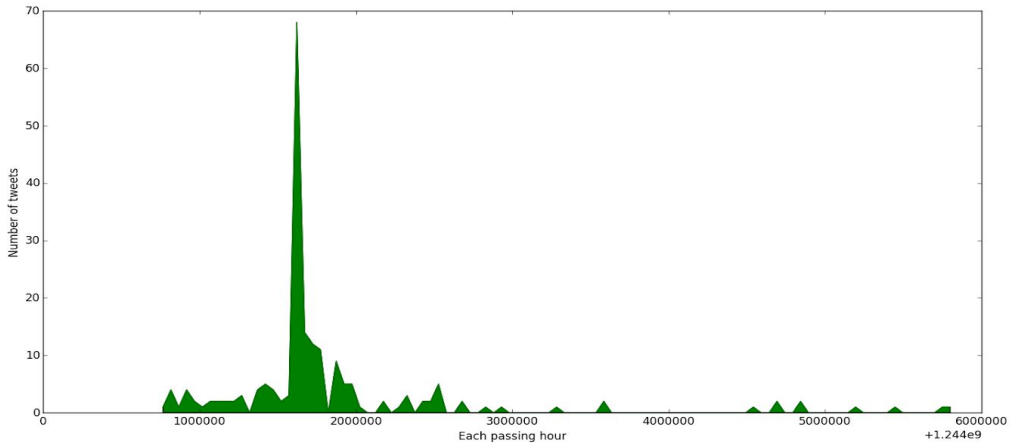
Table 1 - Overview of the data in baseline experiments

An important feature of rumor is that it usually has 1 or more peaks in count vs time. This means that at some point of time, the rumor would be spreading like fire!

Below are the plots of how our datasets look. We plot the number of rumors of the given topic every 16 hours. Rumor diffusion plot usually have a sharp peak as seen in 2 of the graphs. In the second plot, we observe several peaks, which correspond to rumor spread and consequent correction tweets.







For the actual model, we used the second dataset related to the Boston bombing(2).
It has the following statistical properties:

	Count	Rumor ratio
Number of tweets	23.2K	0.58
Number of signal clusters	209	0.58
Training clusters	140	0.56
Test clusters	69	0.62
Average cluster size	70	

Table 2 - Overview of the data in the final model experiment

Only about 500 of the tweets match the regular expressions used for signal filtering. In the signal tweet identification stage, however, it is far more important to have a high precision. Low recall of signal tweets may still be sufficient to get high recall of signal clusters.

METHOD

Baseline Model and Assumptions

Questioning nature of Rumor: Rumor tweets generally question the validity of the rumor. We call them enquiry tweets. There is some amount of skepticism in the possibility of the incident occurring and the users seek validation by asking questions. Therefore our baseline model tries to capture the questioning nature of such tweets. Often these question marks and question phrases are accompanied by exclamation marks. This model counts the number of question marks, number of question phrases and number of exclamation marks in each tweet and uses these counts as features. The question phrases were extracted with simple python regular expression matching.
Popularity of tweets: Rumors are often retweeted a lot and it is a sign of its spread. This model captures the popularity by counting number Retweet (RT …@) mentions within the tweet. Also, if the tweet contains any hashtags, we attribute it to be some popular topic and count the number of hashtags in the tweet. The urls as part of the tweet shows that the topic is already trending elsewhere and is being actively discussed.

Baseline Pipeline

The baseline pipeline is simply using a binary classifier to classify each tweet, we used: Random Forest, Logistic Regression and SVC. The following are 6 features for each tweet:

1. hashtag count
2. exclamation mark count

3. frequency of question phrases
4. frequency of question marks
5. retweet symbol count
6. url count

Final Model and Assumptions

Skeptical Enquiries in Response to Rumor: Our model expects widespread skeptical response to rumor from the public on twitter since the model is detecting rumor tweets based on the skeptical response from the public to the rumor instead of the rumor tweet content itself.

Feasibility of the Human-in-the-Loop: Our model also assumes that human analysts such as journalists can sift through our top-ranked rumor candidate cluster results to eventually obtain actual rumor topics with high precision. From these trending topics, important rumor topics can be clarified and responded to the public just-in-time before rumor diffusion. This assumption is reasonable because a high proportion of the reasonable number of rumor clusters is defined to be rumor. On average, approximately a third of the top 50 rumor clusters are categorized to be rumor clusters on Twitter. [1]

Terminology

1. ***Signal Tweets*** are defined to be tweets with skeptical enquiries that verifies or correct the rumor. One example of signal tweet would be a tweet containing the phrase - “is it true?”
2. ***Non-signal Tweets*** are the tweets that neither verifies nor corrects the rumor. An example of non-signal tweet would be “Breaking News: person X is dead.”
3. ***Full Rumor Clusters*** are clusters of tweets that include signal tweets and non-signal tweets.
4. ***Signal Tweet Clusters*** are clusters of tweets that only include signal tweets.

Algorithm

The key insights of the algorithm that we used is to perform Semi-supervised Learning for rumour classification by clustering and classifying all candidate rumor clusters.

The algorithm pipeline includes 5 main steps as shown in Fig 1:

1. Identify Signal Tweets
2. Cluster Signal Tweets
3. Summarize Signal Tweet Clusters
4. Gather Non-signal Tweets
5. Rank Candidate Rumor Clusters

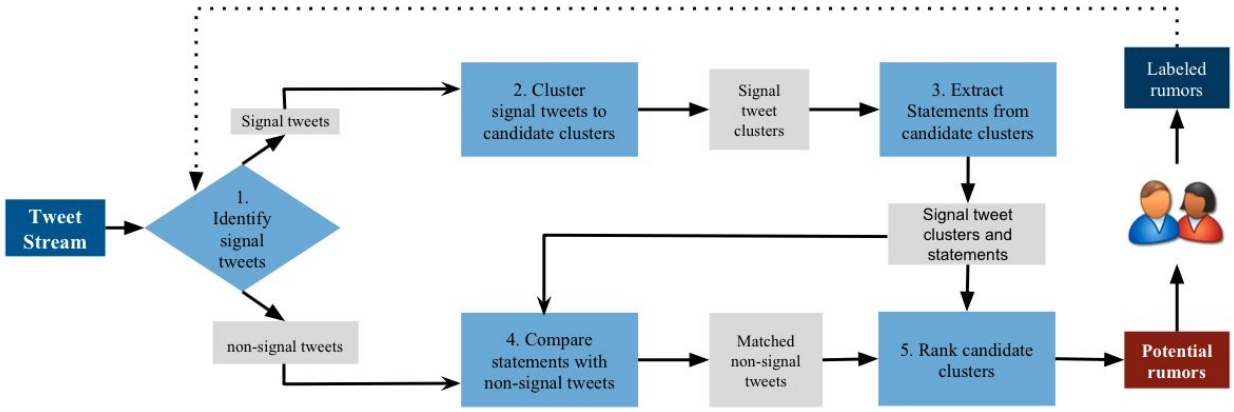


Fig 1: Rumor Detection Model Pipeline

1. Identifying Signal Tweets

Thousands of tweets related to a past incident that stirred several rumors, Boston Bombing, were fetched using Twitter API. This dataset has been selected for our training and testing process of our pipeline. Out of all these raw tweets, some tweets might not be rumor tweets that are not controversial and fact-checkable according to our rumor definition.

The algorithm performs the signal tweet detection process to obtain clues about which tweets are rumor-related tweets. The model detects signal tweets by pattern through regular expressions on the tweets fetched. The regular expressions used to capture skeptical enquiries are shown in Table 3. Some of the regular expressions for signal tweet detection were borrowed from Enquiring Minds[1]. Additional regular expressions for common signal tweets were added by us after a cursory glance on the general content in the tweets.

Pattern Regular Expressions	**Example Tweets**	**Type**
is\s+(that|this|it)\s+\w*\s*(real|rl|true)	is it true	Verification
(omg)|(o)(h)*\s*(my)\s*(god|gawd)	omg, o my god, oh my god, oh my gawd	Verification
(are|is)\s*(true)	the news is true	Verification
real(l|ll|lly|ly)*(\?|\!)+	really?!	Verification
wh[a]*t[?!][?1]*	what?!	Verification
(unconfirm)(ed)*	the unconfirmed report of X's death has been released	Verification
(tru)(e|th)|(false)|(fake)	is it true	Verification

(den)(ial|y|ied|ies)|(plausible)	it is not plausible	Verification
(debunk)(ed)*|(dismiss)(ed)*	the rumor was debunked	Correction
(rumor\s*)(\?)|(hoax)|(gossip)|(scandal)	the rumor about him has been spreading	Correction

Table 3 : Regular Expressions for Signal Tweet Detection Stage

2. Clustering Signal Tweets

The next step is clustering signal tweets. To explain why clustering is useful in our rumor classification problem, consider Boston Marathon Bombing incident. After Boston Bombing, several rumors have spread about a 7-year-old-girl victim of the Boston Bombing, or the White House getting bombed. Soon after a rumor has been diffused, the Twitter users retweet or write a new tweet about the original rumor content. This means that the new rumor tweets about the rumor have overlapping content or the tweets are entirely duplicates for retweeting. It is meaningful to cluster those duplicate tweets as well as similar tweets that talk about the same content. Each of the candidate rumor clusters can be further classified to be rumor or not by using classic machine learning techniques.

Clustering can be performed using different computationally expensive techniques such as K-means. Our training and testing data size is large. For real-time rumor detection, the data size will be even larger than current testing dataset, which increases computational overhead. Also, the purpose of clustering is not to find slight similarity among documents, but to detect if the clusters are almost duplicates or not. Therefore, connected component clustering is used for effective signal tweet clustering. [1] Similarity between tweets is measured using Jaccard coefficient with tri-grams of the tweets. The formula for Jaccard coefficient is given as below:

$$J(d_a, d_b) = \frac{|Ngram(d_a) \cap Ngram(d_b)|}{|Ngram(d_a) \cup Ngram(d_b)|}$$

Here is our conversion from tweets to an undirected graph. Consider each tweet as an isolated node in the graph. An edge is added between two nodes if the Jaccard similarity score of the two corresponding tweets is above 0.6. The connected components are found by performing breadth-first-search and depth-first-search. The connected components with fewer than 3 tweets are dropped from the graph.

3. Summarize Signal Clusters

The content of each signal cluster is summarized as follows. Summary statement for each signal cluster is built out of the most frequent 3-grams in order that appear in more than 80% of the tweets. In our classification process, we have used the summary statements of signal tweet cluster provided by UMichigan NLP group for Boston bombing data.

4. Identifying Non-signal Tweets

After selecting signal tweets and clustering them into groups, the remaining tweets in the dataset are non-signal tweets with no skeptical enquiries. Our candidate rumor clusters must include both signal

and non-signal tweets. To decide whether to add a non-signal tweet or not, the content of non-signal tweet is matched with the summary statement. The similarity score between non-signal tweets with summary statements for rumor clusters is computed. Only non-signal tweets with more than or equal to 0.6 similarity score are added to signal tweet clusters. Full candidate rumor cluster with both signal tweets and non-signal tweets are now ready to be classified.

5. Score Candidate Rumor Clusters

The final goal is to sift through the classified rumor clusters to find important rumor topics and take necessary actions to prevent damage related to rumor in the society. Therefore, it is necessary to rank top x number of rumor clusters with x being reasonably low number of rumor clusters enough for a human to look through each cluster for further investigation. The number of tweets in a cluster can be one of the features since it reflects that specific rumor's popularity. However, not all popular clusters are about rumor. For example, there are several popular pop-culture topics, such as X actress winning Oscar, that are not rumors since it is not a controversial topic. Therefore, for ranking, the model uses other statistical features of the candidate rumor clusters that reflects well upon whether the clusters are about rumor or not. To obtain likelihood for each rumor cluster, SVM, logistic regression and random forest models were used. The 13 features are listed as follows:

1. Percentage of signal tweets

   Ratio of signal tweets to all tweets in full candidate rumor clusters

   Higher number of signal tweets in a cluster shows the controversy of the topic. Therefore, the more signal tweets appear in a cluster, the more likely that the cluster is a rumor cluster.

2. Entropy ratio

   Ratio of entropy of the word frequency distribution in signal tweet clusters to that in full candidate rumor clusters.

   The entropy ratio shows the average information gain of each word in signal tweet cluster compared to that in any tweet in entire cluster. This feature is important since we consider signal tweet as rumor indicator in our clusters.

3. Tweet

   Average signal tweet length (in terms of word counts)

   Average length of any tweet in full tweet clusters

   Ratio of average signal tweet to average length of any tweet

   The length of tweet is a typical NLP feature. Signal tweets are more likely to be shorter than non-signal tweets. The average signal tweet is likely to be short compared to average length of any tweet. One intuition behind this is that shorter the signal tweets are in a typical cluster, the more likely that those tweets will be showing skepticism such as "what?!", "omg!" The shortness of the signal tweet indicates high possibility that the current cluster is about rumor.

4. Retweets

   Percentage of retweets in signal clusters

   Percentage of retweets in entire tweet clusters

   In a rumor cluster, the more popular a rumor is, the more retweets there will be in a cluster. High value of retweet will increase the likelihood of the cluster being rumor cluster.

5. URLs

   Average number of URLs in a typical signal tweet

   Average number of URLs in any tweet in the cluster

   For a controversial topic, people might refer to some URL links to support their statement in the tweet. Intuitively, the more URL there is in most tweets in a cluster, the more likely that a cluster is about rumor.

6. Hashtags

   Average number of hashtags per signal tweet

   Average number of hashtags per any tweet

   The more hashtags in a cluster shows the popularity of a cluster. This will increase the likelihood of being rumor.

7. @Mentions

   Average number of usernames mentioned in a typical signal tweet

   Average number of usernames mentioned in any tweet in the cluster

   The number of usernames mentioned could indicate the popularity of the tweet.

Analysis of the Pipeline

1. Early Detection

The algorithm does not use time-series features such as burstiness that is used previously by quite a number of research groups. Since burstiness feature becomes active only after the rumor has been widely spread, it is incapable of detecting rumors early. Instead of using such time-series feature, the algorithm uses the enquiry and reaction about rumor occurred in Twitter. The key idea behind the algorithm is that most people tend to react skeptically and take effort for further inquiry before the rumor has been believed by majority of people. Several of such reaction and enquiries are found on Twitter after the exposure of rumor to the public. The main reason of detecting rumors at the first

place is to prevent potential damage from confusion and anger in the public arisen from rumor. Therefore, the ability to detect rumor early is an important advantage of our algorithm.

One problem that could arise is that such signal tweets with reaction or enquiry to rumor are very informative but rare, which reduces recall for signal tweet detection. However, in detecting signal tweet, precision is more important than recall since a few signal tweets is sufficient for detecting severe rumors. [1] Our rumor cluster classification results shows reasonable precision and recall and proves that using signal tweets for clustering and classification shows sufficiency.

2. Scalability

One of the key advantages of the algorithm is that the algorithm detects the expected public's skeptical reactions from tweets to detect rumors on Twitter, instead of analyzing individual tweet content itself to find any correlation with specific rumor topics. Such approach of this algorithm makes it applicable to detect any rumor topics. Since this algorithm uses features that are not based on a specific rumor topic, it is scalable to detect "newly emerging, controversial topics" on numerous number of tweets.

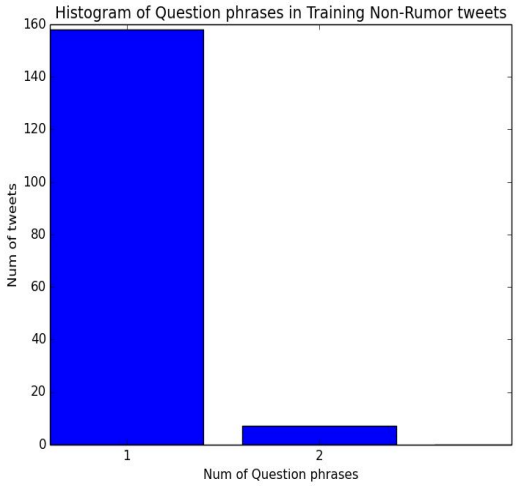
RESULTS

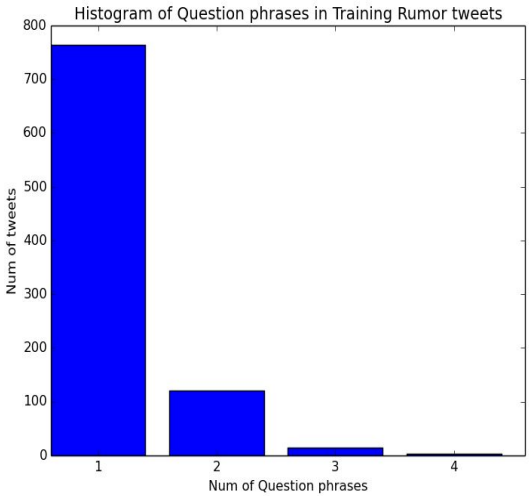
1. Baseline Model Experiments

The goal in our baseline model was to simply classify each individual tweet as a rumor or not. This was quite a misplaced goal, which we ended up realizing after the we analyzed the results in our baseline model. Our conclusion from baseline model experiment is that analyzing the content of tweet itself is hard for rumour classification.

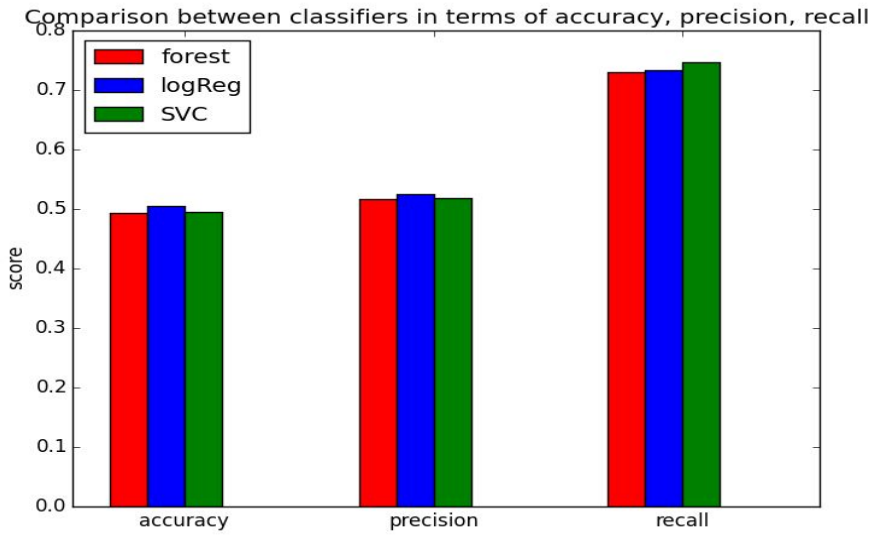
To restate how baseline model works, the baseline model views a tweet as a possible rumor-containing agent and looked for signals within the tweet which we believed to be rumors instead of looking at entire rumor clusters.

We performed some exploratory analysis on the dataset for our baseline model and found a significantly higher frequency for the above features in the tweets labelled as rumors than for non rumor tweets. We include here two sample comparison plots for convenience:

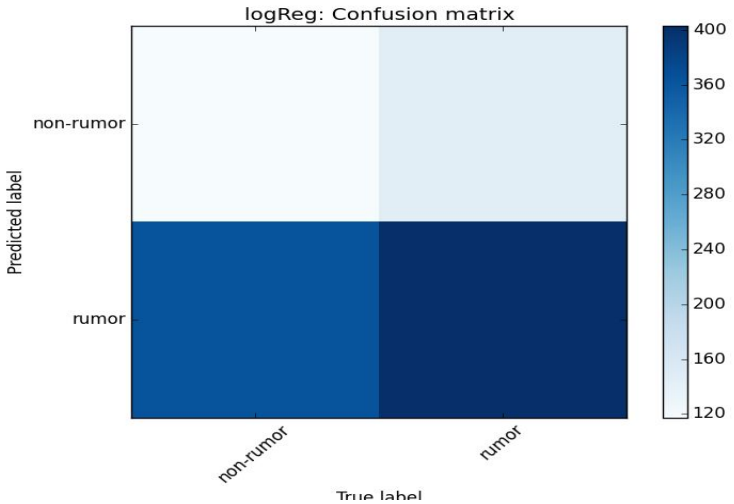




The baseline model is essentially a binary classifier and our natural choice was the well known ensemble method - Random Forest with Grid Search to classify the test tweets. Unfortunately we achieved an accuracy/precision of only 50-53%. In an attempt to verify if our choice of the classifier was the culprit of such low accuracy, we tried two other classifiers - Logistic Regression and Support Vector Classification. This only marginally improved the accuracy by a few decimal points. The results are summarized in the plot below.



Clearly such poor results pointed to a lacking in either the data or the model. The culprit seemed to be more likely the features we have chosen, and not so much as the data itself. We noticed that our model took a very conservative approach as can be seen in the confusion matrix below for the output of the Logistic Regression classifier(highest accuracy/precision). It was classifying most non-rumors as rumors as well. We concluded that our model was flawed and this prompted us to approach the problem in a different way rather than settling for the verdict that classifying a tweet as a rumor was a HARD problem.



2. Final Model Experiments

The key insight of our model is semi-supervised learning that includes clustering and classification. The experiments for final model was divided into 2 parts:

1. Clustering Experiment
2. Classification Experiment

1. Clustering Experiment

UMichigan NLP group clustered over 200 tweet clusters with thousands of tweets using parallel processing and computationally efficient similarity approximation technique named MinHash. Due to time and computation limitation, we have duplicated clustering results for only 3 clusters with 50 tweets in each cluster. This proves that we can potentially duplicate their clustering results given more time and computation power.

The algorithm uses connected component clustering as a way of clustering tweets into candidate rumor clusters before rumor classification stage. The idea of connected component clustering is to compute similarity score for every possible pair of tweets. Computing similarity score for one pair of tweets include computing trigrams for each tweet and find the intersection and union of trigrams between two tweets as mentioned in Method section. Since there will be huge computational cost for computing similarity for every possible pair of tweet in over thousand tweets in our dataset, our group selected 150 tweets out of thousands of tweets. We were able to duplicate the results of UMichigan for 150 tweets which are clustered into exactly 3 clusters. To prove that we were able to duplicate UMichigan group's clustering results, we did the following short experiment of clustering tweets.

The steps are shown in Fig 2 below. The first step is to select 3 tweet clusters from top-10 ranked clusters that were classified as rumor clusters in our classification stage. The next step is to collect tweets from Twitter API that are clustered into the selected 3 tweet clusters by UMichigan group. According to UMichigan's clustering results, each cluster contains about 100-400 tweets. Since having so many tweets makes visualization hard and takes longer computing, we have only chosen only 50 random tweets for each of 3 selected clusters. Since our purpose is to duplicate the result of their group clustering result, we mix up all 150 tweets and start clustering process from scratch.

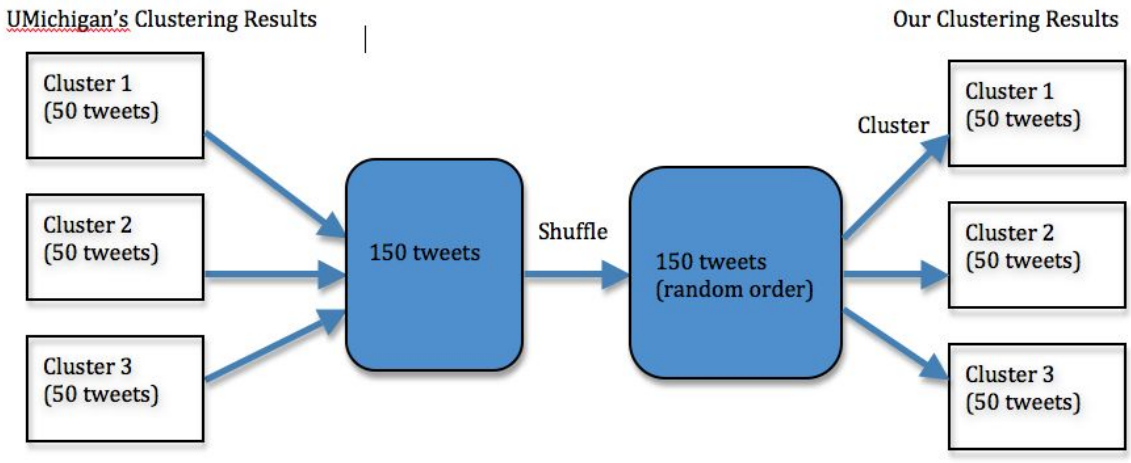


Fig 2: Process of Duplicating UMichigan's Clustering results for 3 clusters

We computed tri-grams of each tweet and Jaccard similarity between all possible pairs of tweets. At the threshold of 0.6, we added an edge between 2 nodes that are represented by each possible pairs of tweets. The final step is to visualize the clusters in NodeXL visualization tool. We expect to see 3 separate connected components in our clustering results. At the end of the clustering process, as shown in figure 3, we obtained 3 connected components in our clustering results which are on different rumor topics, as we expected. Therefore, we successfully duplicated the clustering results of UMichigan's group for 150 tweets, 3 clusters.

From this experiment, it is concluded that we can potentially scale clustering to 200 clusters or more faster in parallel on server. As an improvisation, we can use MinHash technique for faster and more efficient computation of similarity scores and cluster all tweets in the dataset.

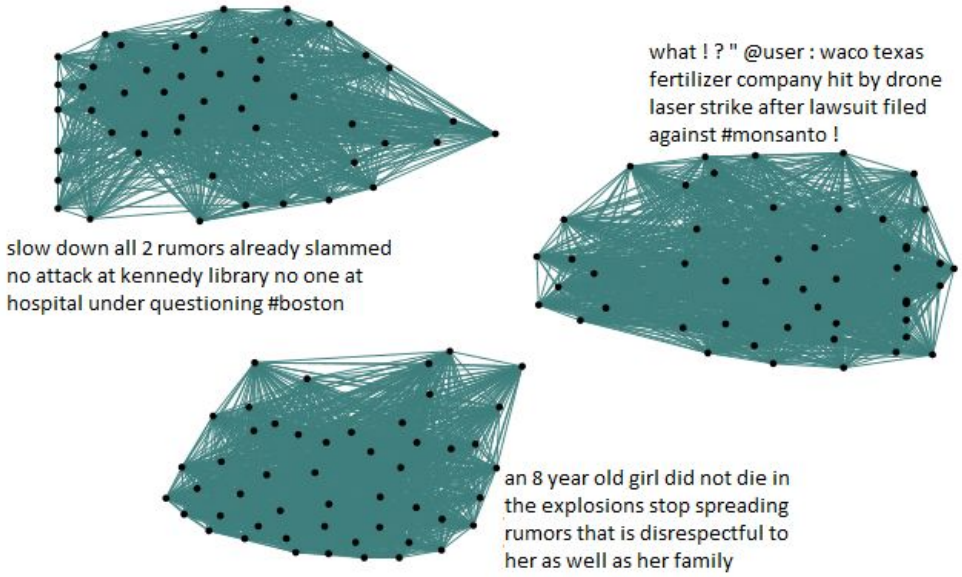


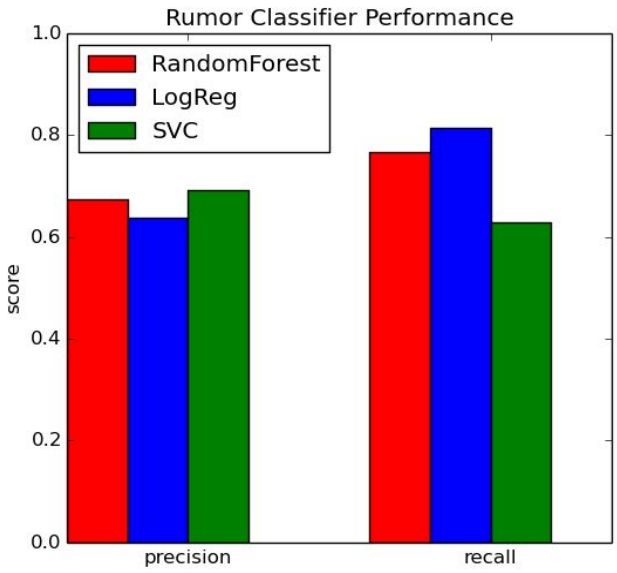
Fig 3 : Clustering results of 150 tweets from our training dataset: Three clusters out of top 6-ranked clusters with summary statements

2. Rumor Cluster Classification Experiment

Our data set is several clusters of rumor tweets provided by UMichigan NLP group. And for the purpose of classification, we are interested in finding if a cluster contains a rumor or not. After gathering the tweets, for each cluster, we partitioned each cluster into: Signal Tweets and Non-Signal Tweets. To identify if a tweet is a signal tweet or not, the system used the matching regular expressions in Table 3. In our clusters, we considered both enquiry and verification tweets as Signal tweets. Using this partition for each cluster, we computed the statistical features of the cluster as described in Method section. This resulted in a single vector of **13 features** for each of the ~200 clusters. Training and test set was created with 66-33% split of this data.

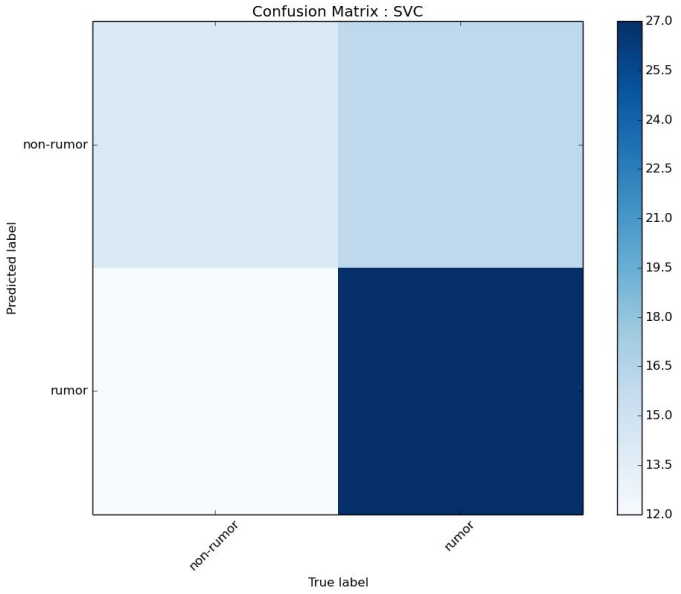
We trained three classifiers to predict if each cluster contained a rumor using the above feature matrix. The classifiers we used are same as in the Baseline model: Random Forest, Logistic Regression and SVC. During prediction process, each classifier ranks clusters based on the probability that the cluster contained a rumor. This was achieved with predict_proba() method in the classifier instance.

The following plot shows the precision and recall of rumor cluster classification for each of the 3 classifiers. SVC achieve the best precision in rumor cluster classification at about 69%. Random forest is the second best at 67%. Logistic regression performs classification at the third best at about 63%. Despite its relatively lowest precision out of all three classifiers, Logistic Regression shows highest recall at about 80% among all 3 classifiers. Random forest shows second best recall value at about 75%. SVC shows the third largest recall value at about 60%. From these results, trade-off between precision and recall can be seen. In summary, SVC performs the best in terms of precision while Logistic Regression achieves the best recall value. We used Grid Search to perform hyper parameter optimization and cross validation. The plot below summarizes the precision and recall of our classifiers.

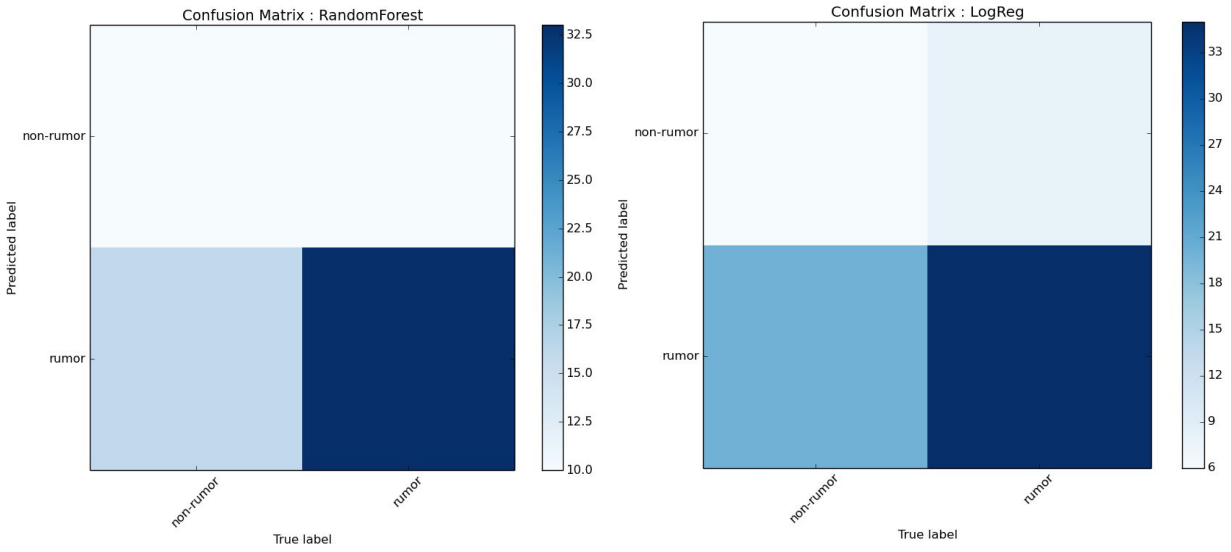


The confusion matrix for our classifier with the best precision- the SVC- is below. It is observed that there are high true positive with low false positive, which causes high precision in our results. The

model gives higher number of false negatives compared to the number of true positives. This results in lower recall compared to precision value.



The confusion matrix for SVC and logistic regressions with second and third best precision are shown below. The confusion matrix for logistic regression shows higher recall than precision since there were more false positives than false negatives. The confusion matrix for SVC shows lower recall than its precision since there are more false negatives than true positives. The confusion matrices for both logistic regression and SVC show high true positives.



3. Evaluation

Top 10 rumor clusters are ranked based on their likelihoods of being rumour clusters. Likelihood of each rumour cluster shows the confidence of the cluster containing a rumour.

Our results proved that this semi-supervised learning model of clustering the tweets and using aggregated statistical features over the entire cluster to rank rumor clusters has significantly improved the precision and recall of all the three models.

Below is the top 10-ranked rumor cluster results along with summary statements. Summary statements of top 10 clusters show mostly different rumor topics arisen from Boston Bombing incident. This shows that clustering process has grouped the tweets into different potential rumor topics. Top-10 Cluster summary statements also show that each statement is about mostly different rumorous topics stirred by Boston Bombing event. For example, from summary statement of rank-1 cluster, the rumor cluster talks about boston bombers carrying guns and gun control policy, which is a controversial topic. The third-ranked cluster talks about xenophobia among the public caused by Boston Bombing. The fourth-ranked cluster is on a popular false rumor that spread soon after the bombing: an 8 year old girl died in the explosion. The fifth-ranked cluster is definitely about Waco Texas Fertilizer Company being hit by drone After lawsuit filed against Monsanto and seems slightly out of context here wrt to the Boston Bombing, but still was a valid trending topic then. The 6th-ranked cluster is about JFK library bomb explosion which was false, actually caused by fire incident.

Rank	**Cluster summary**	**Probability of being rumor cluster**
1	wait what ? criminals ignore gun laws ? maybe more laws would help mt @user : report : boston suspects not licensed	0.871
2	@user do not believe the rumors which say that it is you who made the attack boston we have you directioners	0.867
3	brown black foreign accent rumors about boston suspect seem to be an expression of people's fear	0.862
4	an 8 year old girl did not die in the explosions stop spreading rumors that is disrespectful to her as well as her family	0.846
5	what ! ? " @user : waco texas fertilizer company hit by drone laser strike after lawsuit filed against #monsanto !	0.845
6	rt @user : slow down all 2 rumors already slammed no attack at kennedy library no one at hospital under questioning #boston	0.841
7	rt @user : update : reports of at least 12 dead dozens more injured in boston marathon explosions	0.837
8	what ? ! @user : official : boston bombing suspect has throat injury may not be able to talk	0.834

9	media center is on lock down at the finish of the boston marathon unconfirmed reports of two bombs at the #bostonmarathon : media center	0.832
10	wait what ? rt @user " police : boston bomb suspects lost hostage while buying snacks "	0.831

Table 4: SVC's result: top 10-ranked rumor clusters with probability of each cluster being a rumor cluster

The word cloud results for top 4th, 5th and 6th show the most popular rumor topics arisen after Boston Bombing. The frequent words in word cloud shows that the majority of tweets in the rumor clusters are talking about the same rumor topic.

rt @user : slow down all 2 rumors already slammed no attack at kennedy library no one at hospital under questioning #boston

(a) Word cloud for top 6th cluster

what ! ? " @user : waco texas fertilizer company hit by drone laser strike after lawsuit filed against #monsanto !	an 8 year old girl did not die in the explosions stop spreading rumors that is disrespectful to her as well as her family

(b) Word cloud for top 5th cluster (c) Word cloud for top 4th cluster

Fig 4 : Word Clouds for 3 different top-6 ranked clusters with their summary statements

4. Classifier Efficiency

SVC takes significantly the longest out of all models. This is due to the specific libsvm Scikit Learn implementation of SVC. I quote - " The fit time complexity is more than quadratic with the number of samples which makes it hard to scale." Logistic Regression model and Random Forest takes about the same training time - 5 minutes at maximum and they are reasonably short.

Model	**Training time**
SVC	~ 40 minutes
Random forest	~ 5 minutes
Logistic Regression	~ 2 minutes

Table 5 : Training time of Rumor Binary Classifiers

As we can clearly see, the SVC is taking the longest to train, but since all three models are not highly varying in their precision, any of the models could be used. But if we are pressed for time like we would be in a real time and critical situation, one should choose the Random Forest for the purpose of a real world model implementation of this problem

To summarize, in terms of training time, logistic regression and random forest perform the best while SVC performs much worse. SVC achieves the best precision while logistic regression gives the highest recall.

CONCLUSION

The problem we are trying to solve is "Detecting rumors **early** in Twitter". We can achieve this by fetching signal tweets that contain only enquiry phrases than correction phrases with framing clever regular expressions after some initial data analysis. Earliness can also be achieved if we run an online system, that streams tweets continuously, clustering tweets discussing a common topic - trending topics - and then analyzing each cluster based on the statistical features which are independent of the topic being discussed. Since the model also provides a ranking of the probabilities, the users of this model can act based on the confidence of the output.

Thus this semi supervised learning approach has helped us detect rumors even though we have not classified each tweet as a rumor or not. As a generalized first approach we could label the tweets within a cluster that is classified as a rumor as all rumors. Although this may not be the correct decision. In future we would like to analyze each tweet based on the occurrences of negation statements and other common language terms and classify each tweet as denying the rumor or propagating the rumor. For improved clustering techniques we could use Min Hash as discussed earlier.

Softwares and APIs used

- Scikit-learn, Python
- NodeXL visualization tool
- Twitter API, OAuth

References

[1] Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts Zhe Zhao, Paul Resnick, Qiaozhu Mei. WWW '15 Proceedings of the 24th International Conference on World Wide Web. University of Michigan, MI
[2] F. Yang, Y. Liu, X. Yu, and M. Yang. Automatic detection of rumor on sina weibo. In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, page 13. ACM, 2012
[3] C. Castillo, M. Mendoza, and B. Poblete. Information credibility on twitter. In Proceedings of the 20th international conference on World wide web, pages 675–684. ACM, 2011
[4] A. Gupta and P. Kumaraguru. Credibility ranking of tweets during high impact events. In Proceedings of the 1st Workshop on Privacy and Security in Online Social Media, page 2. ACM, 2012.
[5] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang. Prominent features of rumor propagation in online social media. In Data Mining (ICDM), 2013 IEEE 13th International Conference on, pages 1103–1108. IEEE, 2013
[6] S. Sun, H. Liu, J. He, and X. Du. Detecting event rumors on sina weibo automatically. In Web Technologies and Applications, pages 120–131. Springer, 2013.
[7] T. Takahashi and N. Igata. Rumor detection on twitter. In Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), 2012 Joint 6th International Conference on, pages 452–457. IEEE, 2012.
[8] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei. Rumor has it: Identifying misinformation in microblogs. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 1589–1599. Association for Computational Linguistics, 2011.
[9] A. Gupta, H. Lamba, and P. Kumaraguru. Analyzing fake content on twitter. In eCrime Researchers Summit (eCRS), 2013, pages 1–12. IEEE, 2013.
[10] [30] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, S. Patil, A. Flammini, and F. Menczer. Truthy: mapping the spread of astroturf in microblog streams. In Proceedings of the 20th international conference companion on World wide web, pages 249–252. ACM, 2011
[11] E. Seo, P. Mohapatra, and T. Abdelzaher. Identifying rumors and their sources in social networks. In SPIE Defense, Security, and Sensing. International Society for Optics and Photonics, 2012