TwitterStand: News in Tweets*

Jagan Sankaranarayanan† jagan@cs.umd.edu
Hanan Samet† hjs@cs.umd.edu
Benjamin E. Teitler† bteitler@cs.umd.edu
Michael D. Lieberman† codepoet@cs.umd.edu
Jon Sperling‡ Jon.Sperling@hud.gov

ABSTRACT

Twitter is an electronic medium that allows a large user populace to communicate with each other simultaneously. Inherent to Twitter is an asymmetrical relationship between friends and followers that provides an interesting social network-like structure among the users of Twitter. Twitter messages, called tweets, are restricted to 140 characters and thus are usually very focused. We investigate the use of Twitter to build a news processing system, called *TwitterStand*, from Twitter tweets. The idea is to capture tweets that correspond to late breaking news. The result is analogous to a distributed news wire service. The difference is that the identities of the contributors/reporters are not known in advance and there may be many of them. Furthermore, tweets are not sent according to a schedule: they occur as news is happening, and tend to be noisy while usually arriving at a high throughput rate. Some of the issues addressed include removing the noise, determining tweet clusters of interest bearing in mind that the methods must be online, and determining the relevant locations associated with the tweets.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Storage and Retrieval

General Terms

Algorithms, Design, Performance

*This work was supported in part by the National Science Foundation under Grants EIA-08-12377, CCF-08-30618, and IIS-07-13501, as well as NVIDIA Corporation, Microsoft Research, Google, the E.T.S. Walton Visitor Award of the Science Foundation of Ireland, and the National Center for Geocomputation at the National University of Ireland at Maynooth.

†Department of Computer Science, Center for Automation Research, Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742, USA.

‡HUD Office of Policy Development & Research (PD&R), 451 7th St. SW, Room 8146, Washington, DC 20410, USA.

ACM GIS '09, November 4–6, 2009. Seattle, WA, USA
© 2009 ACM ISBN 978-1-60558-649-6/09/11 ...$10.00.

Keywords

Twitter, News, Geotagging, Online clustering

1. INTRODUCTION

Twitter[1] is a social networking website that recently has been gaining much attention and following. Twitter is composed of users who send messages (termed *tweets*) to each other, where each tweet contains a maximum of 140 characters. At this time, it is estimated that there are 6 to 7 million users who use Twitter a total of 134 million times a month [4], and this number is increasing at a rapid rate. For example, for the year of 2008, Twitter grew in terms of the number of tweets sent at a rate of 1382% [12] which is a testament to the immense popularity and wide adoption of this service. The popularity of Twitter stems from its availability on a number of different electronic devices (e.g., web, cell phones, etc.), as well as the prevalence of a subculture in Twitter that encourages users to acquire a large friend pool, as well as send tweets on a wide variety of subjects, typically several times a day. The restriction on the lengths of Twitter messages invariably means that the tweets do not necessarily contain well formed ideas, being rather brief, yet complete enough so that users can make sense of the ideas that they convey. Note that tweets also have a mechanism by which the user can link to other objects on the web such as articles, images, videos, etc. (termed *artifacts*) which is typically used to link tweets to related material on the Internet.

The goal of this paper is to demonstrate how to use Twitter to automatically obtain breaking news from the tweets posted by Twitter users, and to provide a map interface for reading this news, since the geographic location of the user as well as the geographic terms comprising the tweets play an important role in *clustering* tweets and establishing clusters' geographic foci. In contrast to news aggregators such as Google News, Bing News, and Yahoo! News, we introduce a system called *TwitterStand* that works exclusively with only the tweets posted by the users of Twitter. The key novelty behind TwitterStand is one of mobilizing the millions of users in Twitter to be our eyes and ears in the world, bearing in mind that geographically proximate users often tweet about the same breaking news. In other words, we rely on Twitter users to be either providers of original news content (e.g., the 2008 Southern California earthquake [13] and the 2009 Iranian election [3]), or expressers of opinions on current news topics (i.e., mini blogs), both of which enable TwitterStand to automatically identify current news topics and cluster the corresponding tweets into appropriate news stories. We also associate an importance score with each news topic which can

[1] http://twitter.com

naturally be deduced from the Twitter environment, as any news that motivates a large portion users to express their opinions quickly and loudly should be an important news event. A good measure for determining importance of a news topic is simply to keep track of the number of tweets on the news topic that were found by TwitterStand, and how quickly it found them. In some sense, TwitterStand works because Twitter users and the tweets themselves are wired to the real world, which make Twitter an excellent candidate for tracking news.

Another advantage of Twitter is that it is a social networking website, which means that users need not be viewed in isolation, but instead can be viewed as part of a large network of other users, user groups, and user cliques. Moreover, users have some meta-data information, such as description, source location, friends, etc., which means that the social network structure in Twitter can aid us in finding users that are most likely to tweet about news belonging to a particular geographic location or region. The source location, or the geographic location to which the user belongs, can aid in geotagging of news which means that TwitterStand can perform geotagging on news topics and associate them with the spatial focus of these news stories. It is important to observe that TwitterStand is distinguished from the NewsStand system [18], which is an earlier effort by the authors, in the sense that NewsStand only works with news articles which is a much easier domain than tweets. The availability of user meta-data also means that we can break down opinions on news topics based on a user's sex, location, interests, and other characteristics, which provides interesting perspectives into how these differences manifest themselves into differences of views on issues.

TwitterStand differs from conventional news aggregators. First of all, news aggregators work with a set of content providers (e.g., newspapers, television stations, news blogs, etc.) where any article that they find is related to news. These aggregators, in some sense, leverage on the work done by conventional newspaper organizations. In our case, we simply use the tweets posted by users of Twitter, which means that we have to first determine if the tweet is news or not, as most of the tweets are not related to news and, moreover, they enter into TwitterStand at a very high throughput. So, working with Twitter becomes quite a challenge as TwitterStand needs to be fast, resilient to noise, and has to literally pull needles (i.e., news) out of mountains of haystacks (i.e., tweets). The brevity of the tweets also means that the context of the conveyed information is not present in the tweet itself — the assumption here is that the intended reader is already familiar with the context. What this means is that the information in tweets is time critical, or in other words, there is a small window of time when the users can relate to a tweet, after which it becomes difficult to parse for meaning.

There are a number of reasons why Twitter, even though a very challenging medium, is useful for tracking news. Not surprisingly, Twitter does not mandate any rules or guidelines as to what constitutes an acceptable or desirable tweet. The consequence is that users post tweets on every conceivable subject, which makes it a wonderful medium for any kind of *concept* extraction. TwitterStand can be viewed as a system to extract concepts where the concept we are trying to identify from tweets is *news*. Twitter attracts a diverse community of users with varied interests [9], although it is not clear what motivates users to record or distribute their thoughts in this medium. Of course, many explanations can be posited including the small length of the tweets, and its availability on so many different electronic mediums. Thus, if we want to find expressions of a particular concept on Twitter, we are most likely to find it, except that it is not clear how we would go about doing it. However, given the diversity of the topics, finding a single concept such as news is, as we said before, akin to finding needles in stacks of tweets. In our domain, any tweet that does not belong to the news domain is termed *noise*. Thus, one of the main contributions of TwitterStand is the ability to identify and cluster news tweets in a very noisy medium, which is a hard problem. Note that if Twitter was a more restrictive medium, then it would not be interesting for the kinds of issues that we address in this paper.

Twitter is a technology that breaks down communication barriers. It is a medium of instantaneous feedback which means that any action in the real world usually receives a near instant reaction or feedback in terms of tweets expressing opinions or reactions to the action. This is another reason that makes Twitter attractive for capturing breaking news, as there is very little lag between the time that an event happens or is first reported in the news media and the time at which it is the subject of a posting on Twitter. The data is "pushed" by the content providers (i.e., people who send tweets) and is delivered nearly instantaneously to the content consumers (i.e., people who receive tweets, and TwitterStand). This is in contrast with conventional news aggregators that must constantly poll the content providers for updates with web spiders, which means that there could be a significant lag between the time the news is published and is first picked up by the news aggregators. Thus we see that from the point of view of speed (i.e., the ability to have a scoop), Twitter gives TwitterStand an edge over conventional news aggregators. This can be seen by examining Figure 1 which shows the relative increase in samples of overall tweet activity (in tweets per hour) about the recent tragic death of Michael Jackson, where tweets on his illness and death were reported more than an hour before conventional news media.

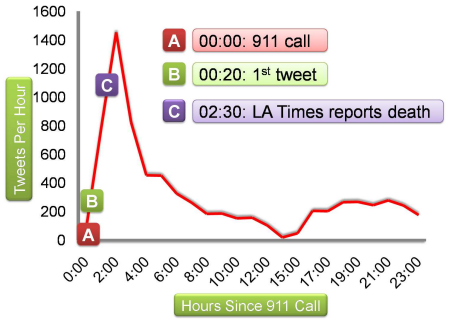

Figure 1: Traffic in tweets per hour relating to Michael Jackson's death. The first tweet in the story was reported 20 minutes after the 911 phone call, which was almost an hour before conventional news media first reported on his condition.

The conventional news media is often criticized for not exposing consumers to alternate or less popular view points, which is also a problem for news aggregators as they simply aggregate news from conventional news media. TwitterStand, on the other hand, aggregates tweets which are basically original content, albeit a sentence or two long, composed by thousands of different users. This naturally results in a wide diversity of opinions on a news topic which makes TwitterStand a good medium for exposing readers to the various facets of a news topic that are otherwise not addressed by

conventional news media. An immediate problem with using such diverse sources is that the news might not be so credible which sometimes results in misinformation being put out as news [2]. Although this issue is the not the focus of our work, it can be resolved by imposing a rating system of users, similar to the ones used in auction systems (e.g., Ebay, Amazon) that rewards users by giving them high credibility scores for publishing quality and reliable news.

The rest of the paper is organized as follows. Section 2 describes the working of tweets and Twitter, while Section 3 lays out the basic strategies in working with Twitter. Section 4 presents the architecture of TwitterStand. Finally, concluding remarks are drawn in Section 5.

2. TWITTER

In this Section we describe some of the concepts of Twitter, the understanding of which is important to designing TwitterStand. In particular, we will describe the Twitter system in terms of users, friends, followers, tweets, hashtags, and artifacts. We will also describe some of the services provided by Twitter's public Application Programming Interface (API).

2.1 User, Friend and Follower

A user is a *person* or a *system* that posts a tweet or a message on Twitter. In Twitter, a user is identified by a unique user name. When a user chooses to obtain all tweets written by another user, a *friend-follower* relationship is established. That is, if user a agrees to receive all the tweets from another user b, then b is said to be a *friend* of a. Conversely, when a added b as a friend, from the point of view of b, a became a *follower* of b. All tweets posted by b are now received by a, while the converse need not be true as a may not be a friend of b. This asymmetrical relationship between friends and followers provides an interesting social network-like structure among the users of Twitter, and is probably the primary reason for the popularity of Twitter. Also, Twitter imposes a limit of 2000 friends for a user, but there is no limit of how many followers a user can have. It is not uncommon for some users, such as celebrities, to have millions of followers. These limits mean that given two users in a friend-follower relationship, it can be hard to ascertain the attribute that is common to them, if any. A variety of meta-data is associated with each Twitter user, such as the list of friends and followers, source location (i.e., the geographic location of the user), and other user attributes such as real name, a brief biography, photograph, total number of tweets posted, last date when the user posted a tweet, and favorite links.

2.2 Tweet

A tweet is a short message from a user to its set of followers. In general, tweets are related to blog posts, in the sense that a tweet is to a blog as an SMS is to an email. A key property of tweets is that they are restricted to 140 characters and hence are much more concise than blog posts, which is probably the reason why users of Twitter send them more often, and usually on diverse topics. The short allowance on tweet size means that the users must be concise, and sometimes have to resort to phrase abbreviations. Interestingly enough, there is a rich and well understood set of abbreviations which is surprisingly consistent across user groups, and even across other electronic mediums such as SMS and chatrooms. Twitter has its own set of abbreviations, such as "RT", "Retweet", which means that a past tweet is being sent once again by a user, and "DM", for a direct message from one user to another. Apart from these abbreviations, tweets may also contain spelling errors or be grammatically incorrect, which all contribute toward making Twitter a challenging medium to work with.

As Twitter is a broadcast medium, tweets are usually directed towards a general audience, but there are special provisions by which a user can direct a tweet at another user, or send it as a response to an earlier tweet. This is usually done by appending a "@" symbol to the intended user name (e.g., "@newsstandumd") and including this keyword towards the beginning of the tweet. Direct messages, which are part of the input to our system, are useful as they provide a way of establishing connections between two users, or between two tweets.

Another peculiarity of tweets is that sometimes they contain words prepended by "#" (hash symbols), which are referred to as *hashtags*. By including a hashtag in a tweet, the user originating the tweet is suggesting that the word denoted by the hashtag makes for a good candidate as a search key for the tweet. For example, tweets about the 2009 Iranian elections typically contain the hashtag "#IranianElection", "#Iran", or some number of other variations, as shown in Figure 6. It is a fascinating yet not so well understood phenomenon that even though there is no entity regulating the hashtags assigned to tweets, there seems to be only a small number of hashtags for a particular news topic. In fact, a simple clustering just based on the similarity of the hashtags yields surprisingly good cluster quality, and this simple idea forms the basis of services like *Twitter Search* [21], *Twist* [20], *Monitter* [15], *Hashtags* [7], *TweetMeme* [19], and others. However, we observe that the resulting clusters are not comparable to those from an online clustering based on tweet content.

It is important to note that 140 characters does not allow for tweets that are verbose, or those that contain a well-developed idea. This is probably the reason that tweets oftentimes contain URL link which either directs the reader to a news article, a blog post, or in many cases interesting artifacts such as related websites, videos, photographs, etc. Since URL strings can be much longer than 140 characters, tweets usually make use of forwarding services that shorten these links to under 10–20 characters in length. Some of these services are "`http://tinyurl.com`", "`http://bit.ly`", "`http://r.im`", and "`http://z.pe`".

2.3 Twitter Services

Twitter provides the necessary infrastructure to ensure that tweets written by a user are broadcasted to all the user's followers, and that tweets from the user's friends are delivered to the user in a timely fashion. This stream of tweets is available to the user through several different sources, including the main Twitter website as well as several other websites built around Twitter. These tweets are also available through the Twitter API service in Extensible Markup Language (XML) and JavaScript Object Notation (JSON) formats. Twitter also publishes differently-sized samples of the entire set of tweets generated by all Twitter users through three public feeds, called the *public timeline*, *spritzer*, and *GardenHose*. These feeds are of different and increasing sizes, although the public timeline and spritzer feeds are relatively small, with a difference of several orders of magnitude. On the other hand, the GardenHose feed provides a limited set of approved Twitter users with access to a much larger stream of tweets, although there is no way of determining what percentage of tweets constitute it, except that it is still a very small percentage of all the tweets generated by all the users of Twitter. In addition to the above feeds, a limited set of approved Twitter users also have access to a service called *BirdDog*, that allows

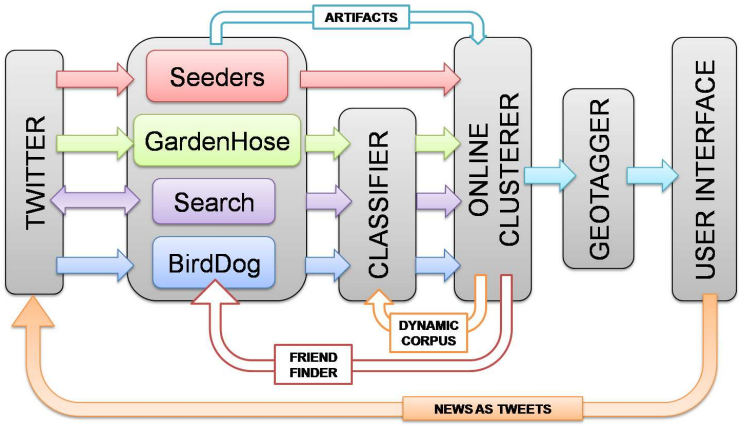

Figure 2: System architecture of TwitterStand.

users to obtain all the tweets written by up to 200,000 users in Twitter. In other words, the BirdDog mechanism allows certain users to exceed the normal limit of following just 2,000 other users. It is important to note that each such user has a different BirdDog feed. Therefore, the key difference between the GardenHose and the BirdDog feeds is that in the case of the BirdDog feed, we have to specify a list of users to obtain all of their tweets, while the GardenHose feed is just a sparse sampling of all the feeds posted by all the users in Twitter with no control over the identity of the users that generate the tweets in the feed. Note also that all of these feeds vary periodically in their traffic according to the time of day, such as weekdays versus weekends, day versus night, etc.

Twitter also provides a search service to continuously monitor or search tweets by keywords, hashtags or user name, but this service is limited to 40 search keywords. Also, Twitter has API functions to obtain user-specific information that can be used to build a network of friends.

3. KEY STRATEGIES

In this Section we present a few key strategies that we developed in order to build a news system from Twitter, which is a very noisy medium with data coming in at a high throughput rate.

1. Tweets arrive at a furious rate, and an attractive aspect of Twitter is that these tweets capture the pulse of the moment. The worst thing that we can do with such a data stream is to not process it right away but to do so in batches. However, processing tweets right away is more easily said than done. We are familiar with algorithms that have access to the entire input set before the start of execution, but designing algorithms that can work without seeing the entire dataset is not easy. In particular, it calls for algorithms that are *online* in nature, which means that they can work on input datasets where the input is encountered one element at a time.

2. Tweets are inherently noisy, as most of them are of very little interest to a broad audience. The challenge in working with such a noisy medium is that we must first improve the quality of the input by trying to extract useful information from noise. Even those tweets that are related to news have less informational content than news articles or blog posts, due to the limitations on the size of the tweets as there is only so much that can be conveyed in 140 characters. Also, spelling errors, abbreviations of words, and grammatically incorrect language make solving this problem a challenge.

3. Twitter is a constantly evolving medium where users leave, new users join, active users become inactive, and users become friends and followers. Consequently, algorithms intended for such an evolving medium should keep up with its dynamics. This means that systems built on Twitter should make new knowledge discovery an intrinsic part of their inner workings, or their effectiveness will diminish as the medium changes significantly since the time that the system was designed.

4. One of the main reasons for Twitter's success is that it has an open policy when it comes to which users can be friends of other users. In particular, there is no limit on who can be your friend or who can have you as a friend as well as how many of them (in this case they are your followers); however, there is a limit on how many friends you can have (i.e., follow). As a result, it is commonplace for users to be friends with other users who are unknown to them in real life. This has the consequence that two users that are friends need not always have similar interests. In our application, we are interested in tweets related to news events. Therefore, for a news system like TwitterStand to be successful, it is necessary to identify a core group of people who mainly tweet about news. A good strategy for finding users with certain interests (e.g., news) is to manually identify users that are related to news and then look for the most common set of followers among them. In our experience, this works much better than other approaches that try to mine structures in the social networks of users in Twitter. Furthermore, studies (e.g., [8]) have shown that a small core group of people accounting for less than 10% of the users in Twitter are responsible for more than 90% of the tweets, indicating that manual identification of a core group of news-related users is feasible.

5. In most cases, Twitter will lag behind conventional news outlets, excepting those cases where the users themselves provide the news such as the 2008 Southern California earthquake [13], or the 2009 election results in Iran [3]. Detecting breaking news is one of TwitterStand's goals, but this is a hard problem. Another aspect of TwitterStand that distinguishes it from conventional news aggregators is that by aggregating tweets on a news topic, we also obtain rich user-generated news content such as videos and photographs, or links to con-

tent on the Internet that is not typically covered by conventional news media. The news posted as tweets usually have a different flavor from conventional news sources in the sense that, at least presently, it is more biased toward entertainment, politics and technology than towards, say, science and philosophy.

4. ARCHITECTURE OF TWITTERSTAND

Our goal is to build a news processing system from Twitter tweets. The idea is to capture tweets that correspond to late breaking news. The result is analogous to a distributed news wire service. The difference is that the identities of the contributors/reporters are not known in advance and there may be many of them. The tweets are not sent according to a schedule. We have no reporters being assigned to cover stories. The tweets occur as news is happening and are noisy while usually arriving at a high throughput rate. In the following we present the system architecture of TwitterStand, which is shown in Figure 5.

4.1 Inputs

Figure 3: Source location of users in the Seeders feed.

Seeders.

Seeders come from 2,000 handpicked users that are known to publish news. They consist of newspapers and television stations that publish news in tweets, as well as users drawn from newspaper reporters, bloggers who write on news topics, and other users who are primarily interested in news. We show the source location of the users in the Seeders feed in Figure 3. Tweets from Seeders usually contain links to articles on news websites (i.e., artifacts). Note that even though this input is similar to a conventional news website, there are some differences between this and systems that spider news websites for updates. In the case of Twitter, Seeders' objective is to get the news as quickly as possible to their intended audience, meaning they will push news tweets as soon as possible. As a result, there is almost no lag between the time that the news is produced and the time that it is obtained by us. On the other hand, systems like Google News and NewsStand must poll news websites for updates. So, there is an advantage in Seeders finding news as opposed to the other way around.

GardenHose.

We have access to the *GardenHose*, which is a sampling of all the tweets published by all the users of Twitter. This is a very noisy input feed and contains tweets from diverse topics, most of which are not related to news.

BirdDog.

This service allows us to obtain feeds from up to 200,000 users. One of the goals of TwitterStand is to constantly keep identifying those users that usually tweet about news. This set is initially empty, but as we identify users that are good candidates, we add them to this set. The algorithms for identifying such users are described in Section 4.7.

Artifacts.

A tweet may sometimes link to an external resource referred to as an *artifact*, which is usually intended to be additional content given the brevity of the tweet. Usually, these linked artifacts are articles, images, videos, or other media on the Internet. We use an image and video extraction algorithm from NewsStand to index and extract these artifacts. When the artifact is an article, we can either use the text from these articles, or simply discard them. In TwitterStand, we only retain those artifacts from the Seeders feed, and discard those from other feeds.

Track.

TwitterStand's track service allows users to specify up to 40 keywords, and provides a stream of tweets containing one or more matching input keywords. This affords users a powerful way to have TwitterStand automatically generate a pool of appropriate search keys to scour Twitter for potential news tweets of interest. We discuss a method to automatically generate these search keys in Section 4.6.

4.2 Separating the Chaff

To distinguish news from noise, we apply filtering on incoming tweets from all the inputs, with the exception of those tweets from the Seeders. In particular, we classify incoming tweets as either *junk* or *news*, where junk tweets have a good chance of not being related to the news and hence, are discarded, while the news tweets have a good chance of being related to news. Note that our goal is not to completely rid of noise, which may not even be possible given the uncertain boundary between news and noise, but instead to find a way of discarding tweets that clearly cannot be news. So, the goal here is to throw away as many tweets as possible without losing many news tweets. For this purpose, we use a naive Bayes classifier [14] that is trained on a training corpus of tweets that have already been marked as either news or junk. Given a tweet t represented as a set of words $w_1, w_2, \ldots w_k$, the probability that t is junk is denoted by $P(J|w_1, w_2, \ldots w_k)$, which can be rewritten as follows using Bayes' theorem:

$$P(J|w_1, w_2, \ldots w_k) = p(J) \cdot \frac{p(w_1, w_2, \ldots w_k|J)}{p(w_1, w_2, \ldots w_k)}$$

Similarly, given a tweet t, the probability that it is a news tweet is given by $P(N|w_1, w_2, \ldots w_k)$, which can also be rewritten using Bayes' theorem.

$$P(N|w_1, w_2, \ldots w_k) = p(N) \cdot \frac{p(w_1, w_2, \ldots w_k|N)}{p(w_1, w_2, \ldots w_k)}$$

Using the assumption of independence among the words in t, we can reduce the above equalities as follows, where Z is a normalizing factor that ensures $P(J|w_1, w_2, \ldots w_k) + P(N|w_1, w_2, \ldots w_k) = 1$.

$$P(J|w_1, w_2, \ldots w_k) = \frac{p(J)}{Z} \cdot \prod_{i=1}^{k} \frac{p(w_i|J)}{p(w_i)}$$

$$P(N|w_1, w_2, \ldots w_k) = \frac{p(N)}{Z} \cdot \prod_{i=1}^{k} \frac{p(w_i|N)}{p(w_i)}$$

Dividing the above equalities by each other, and taking the logarithm of both sides, we get:

$$D = \log \frac{P(J|w_1, w_2...w_k)}{P(N|w_1, w_2...w_k)} = \log(\frac{P(J)}{P(N)}) + \sum_{i=1}^{k} \log \frac{p(w_i|J)}{p(w_i|N)}$$

If $D < 0$, then the tweet is classified as *news*, else the tweet is classified as *junk* and discarded. In the above equation, $P(J)$ is the fraction of the total number of words in tweets marked as junk in the corpus to the total number of words in the corpus. Similarly, $P(N)$ is the fraction of the total number of words of tweets marked as news in the corpus to the total number of words in the corpus. Now, $P(w_i|J)$ is obtained by taking the ratio of the number of times word w_i appears in tweets marked as junk in the corpus to the total number of times it appears in the corpus. Similarly, $P(w_i|N)$ is obtained by taking the ratio of the number of times word w_i appears in tweets marked as news in the corpus to the total number of times it appears in the corpus. Note that we calculate the values of $P(J)$, $P(N)$, $P(w_i|J)$, and $P(w_i|N)$ apriori and store them in a disk-based table.

In order to ensure that we do not classify tweets related to news as junk, we use a corpus that has both a *static* and a *dynamic* component. The static corpus is made up of a large collection of news tweets labeled as *news* as well as another large collection of tweets that are *junk*. In addition to the static corpus, we also use a smaller dynamic corpus of tweets, labeled as *news*, which is periodically obtained from the clustering module. This corpus contains recent news tweets belonging to the news topics at the moment. The idea here is that the static corpus aids us in identifying news tweets on topics that we have not encountered previously, while the dynamic corpus aids us in identifying tweets about current events. In addition to keywords, the dynamic corpus also contains names of people in the news and current hashtags, which both aid in classifying tweets as news. Those tweets classified as news now proceed to the next stage of execution.

4.3 Online Clustering

The main goal here is to automatically group news tweets into sets of tweets, termed *clusters*, such that each cluster contains tweets pertaining to a specific topic. This problem, called *topic detection*, is similar to clustering in the document domain, and differs from tweet classification in that TwitterStand does not know the identity of topics beforehand. Furthermore, given that we are interested in detecting new topics, no training set can accurately predict future events. As news tweets enter this stage, we assign them to news clusters, which is a one-shot process, meaning that once a tweet is added to a cluster, it remains there forever. We will never revisit or recluster the tweet, which is desirable because tweets are coming into TwitterStand at a high throughput rate, and we need a fast and efficient tweet clustering system that maintains good quality clustering output. In other words, our clustering algorithm is an *online* algorithm, and the additional constraints imposed on this problem add new complexity. In particular, we use a clustering algorithm, called leader-follower clustering [5], which allows for clustering in both content and time. We modified it sufficiently so that it works in an online fashion as well as becoming more resilient to noise.

Our online clustering algorithm maintains a list of active clusters. Along with each cluster, we associate a *feature vector* (i.e., weighted list of keywords), collected from the contained tweets' terms and weighted using the TF-IDF [10, 16] measure. We also store the *time centroid* of each cluster, which is the mean publication time of all the tweets forming the cluster. A cluster is marked *inactive* if the time centroid is greater than three days, in which case no additional tweet can be added to the cluster. When an input news tweet t is obtained, we first represent t by its feature vector representation using TF-IDF. We then use a variant of the *cosine similarity measure* [17] for computing the distance between t and a candidate cluster c, which is defined as follows:

$$\delta(t, c) = \frac{\overrightarrow{TFV_t} \bullet \overrightarrow{TFV_c}}{||\overrightarrow{TFV_t}|| \, ||\overrightarrow{TFV_c}||}$$

where $\overrightarrow{TFV_t}$, $\overrightarrow{TFV_c}$ are the feature vectors of t and c, respectively. t is added to the closest cluster c as long as $\delta(t, c) \leq \epsilon$, where ϵ is a pre-specified constant. If no such cluster exists, then we start a new cluster with t as its only member.

To account for the temporal dimension in clustering, we apply a Gaussian attenuator on the cosine distance that favors input tweets being added to those clusters whose time centroids are close to the tweet's publication time. In particular, the Gaussian parameter takes into account the difference in days between the cluster's time centroid and the tweet's publication time. Our modified distance formula is

$$\dot{\delta}(t, c) = \delta(t, c) \cdot e^{\frac{-(T_t - T_c)^2}{2(\sigma)^2}}$$

where T_t is t's publication time and T_c is cluster c's time centroid.

To expedite the search for a cluster c that is nearest to t as well as within a distance of ϵ from it, we maintain an inverted index of the cluster centroids. That is, for each feature f, the index stores pointers to all clusters containing f. We use this index to reduce the number of distance computations required for clustering. When a new tweet t is encountered, we only compute the distances to those clusters that have at least one feature in common with t. As a further optimization, we maintain a list of *active* clusters whose centroids are less than a three days old. Only those clusters in the active list are considered as candidates to which a new article may be added. Together, these optimizations enable our algorithm to minimize the number of distance computations necessary for clustering a tweet.

Nevertheless, given the noisy input and the additional constraints on the clustering algorithm, it is not surprising that this algorithm alone does not work very well. Below, we briefly describe some additional improvements to this algorithm that make it better suited for the developmental challenges unique to Twitter, and ensure better clustering results.

Dealing with Noise.

It is important to understand that we are trying to cluster in a very noisy medium which is a big challenge in itself. Here, we take a useful tip from gardening, which is that if you are careful about the seeds that you plant, you will only grow the plants that you desire. The analogy in the document domain is that if we are careful about seeding good quality clusters, we will obtain relatively noise-free news clusters at the end of the clustering process. A simple way of doing this is to only allow tweets from reputable sources, such the *Seeders*, to form new clusters, while all the other feeds are permitted to add to existing clusters, but are not allowed to start new clusters. In other words, if a tweet from any feed other than the Seeder feed is not within ϵ of an existing cluster, we will simply discard it. The drawback of this compromise is that if a particular news topic (e.g., the 2008 Southern California earthquake) is reported by users of Twitter before conven-

tional news sources, then this clustering policy will drop the tweets on this breaking news event until a cluster is seeded by a tweet from the Seeders feed. In order to overcome this drawback, we relax the above rule by allowing any tweet to form a cluster, but the resulting cluster is marked as inactive if after k tweets have been added to the cluster, none of the k tweets are from the Seeders feed. This is analogous to a gardener waiting for a sapling to grow before deciding if it is a weed or not. From our experience, the idea of using Seeders to determine if a cluster is news or not, is a very powerful idea as it demonstrates how good clustering results can be achieved even in very noisy environments, which is a contribution in itself.

Fragmentation.

A major problem with the online clustering algorithm is that it leads to *fragmentation* meaning that there can be several duplicate clusters on a single topic, which is undesirable. This occurs frequently with online clustering algorithms when an input tweet t belonging to the same news topic as a cluster c is at a distance of ϵ from c, which means that t starts a competing duplicate cluster c'. Subsequent topical tweets now can be added to either c or c', which means that in the long run, c can potentially lose half of its tweets to c'. In reality, this situation can deteriorate to such an extent that the incoming tweets are being distributed amongst tens or even hundreds of duplicate clusters on the same topic, which means that there will essentially be no clustering. We overcome this problem by periodically checking amongst all the active clusters if they are potential duplicate clusters on the same topic. In particular, if we find that c and c' are duplicate clusters, we mark the older one in terms of the time centroid, say c, as the *master* cluster, and mark c' as the *slave* cluster. Now, any input tweet that is within a distance of ϵ of a slave cluster is simply added to the master cluster. This arrangement works well because it practically removes the problem of fragmentation at the expense of a few tweets being lost to the slave clusters.

Weight Upper Bounds.

One issue that arises with a dynamic corpus is that new features that enter into the corpus for the very first time will have a large TF-IDF values, even if they are relatively unimportant, or more common in tweets, or simply misspelled words. Consider a tweet t which arrives into TwitterStand, and contains a feature f which is not present in our corpus due to which f will initially have a high TF-IDF value. For the sake of argument, let us assume for the rest of this discussion that f is not an important feature. As f's TF-IDF value is large, this may cause incoming tweets containing the feature f to cluster with it, resulting in clusters that only have f in common, resulting in spurious clustering. In order to avoid this problem, we stipulate that a single feature alone (i.e., f) should not cause a tweet to be added to a cluster. We will require that at least k dominant terms, where k is typically a small number (but more than one), should be in common between the tweet and the cluster to which it is added.

Phrases.

Phrases are features containing two or more terms (e.g., "Barack" and "Obama") that usually occur together, and sometimes they only have a semantic meaning when grouped together (e.g., "San" and "Francisco") which is otherwise lost when they appear individually. Treating them as separate features (i.e., the *bag of words* model) often produces a correct clustering. For TwitterStand, we are more concerned with spurious clusters arising when the multiple words that make up a single phrase end up with large TF-IDF scores, thereby resulting in tweets being added to a particular cluster because they share a phrase in common. For example, the phrase "Barack" and "Obama" may dominate the features of a cluster to such an extent that tweets may now be added to this cluster simply by containing the phrase "Barack Obama" in them, which may result in poor clustering. There are several ways of addressing this problem. A simple way of determining if words t_1 and t_2 belong to the same phrase is to see whether there exist many occurrences of t_1 close to t_2 within a tweet (not separated by a punctuation mark). Here we distinguish two kinds of relationships between words in a phrase. In the case of "Barack" and "Obama", the word "Obama" dominates the word "Barack" as it is more commonly used, sometimes even without the other one being present. In this case, we choose the more dominant one (i.e., "Obama"). On the other hand, in the case of "San" and "Francisco", both the features appear together all the time, and we simply merge them to form a single feature.

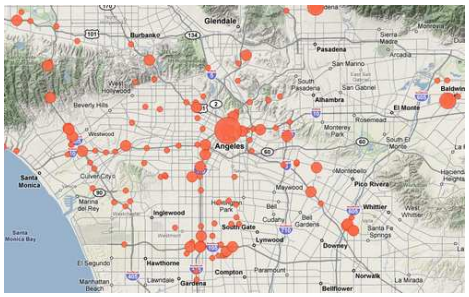

Figure 4: Geotagging of the tweets of accidents in Los Angeles, CA was made possible by incorporating the source location of the user.

4.4 Topic Geographic Focus

After clustering the tweets introduced to TwitterStand into topics, we now try to locate and extract geographic content from each cluster. In other words, we associate each cluster of tweets with a set of geographic locations that is determined to be its geographic focus. This process enables spatial exploration of the news in tweets by unifying explicit tweet content with its implicit geography. We do so by making use of both tweet content (i.e., text) and the source location of the user that generated the tweet. Figure 4 is an example of geotagging tweets of traffic incidents in Los Angeles, CA, which was made possible by incorporating the tweets content with the source location (i.e., Los Angeles) of the user. Using both content and source location is especially important for tweets due to their limited message size and minimally-edited nature. Once geographic content has been extracted from each tweet, we aggregate per-tweet geographic content to determine the cluster's overall geographic focus. Below, we describe our content and source location based geographic location extraction methods for a single tweet, and our topic focus method.

Using Tweet Content.

In terms of tweet content, we perform what is called *geotagging* [1] on each tweet's text, as well as any articles linked from the tweet (i.e., the tweet's artifacts). Geotagging consists of two steps. The first step, *toponym recognition*, means finding all instances of textual references to geographic loca-

Figure 5: Screenshot of TwitterStand with the left pane showing the result of the online clustering on tweets with their corresponding geographic focus shown in the right pane

tions, called *toponyms*, in the text, and is difficult because many names of places are also names of other entities (e.g., "Jackson" which can be a location but is also a common surname). The second step, *toponym resolution*, involves determining the correct geographic coordinates for each recognized toponym out of all possible interpretations, which are often numerous (e.g., choosing the correct "Paris" out of over 140 possibilities).

For toponym recognition, we use a combined strategy that uses tools built to solve related recognition problems in Natural Language Processing (NLP), namely Part-Of-Speech (POS) tagging [10] and Named-Entity Recognition (NER) [10]. POS tagging requires that each word in the text is assigned with its proper part of speech, as used in context, which is helpful for toponym recognition because toponyms are usually proper nouns. NER's goal is to extract typed entities from text such as persons, organizations, and locations, of which the latter can be used directly for toponym recognition. State-of-the-art statistical POS and NER methods use sentence structure and language features learned from a large corpus of annotated text. However, these POS and NER methods carry a significant caveat when geotagging tweets, because no annotated corpus of tweets yet exists. They are normally trained on corpora of full text documents such as news wire articles, which are very different from tweets in terms of length and content, and so may not correctly recognize the toponyms in tweets. In this case, we fall back on using the TF-IDF measure [10, 16], which extracts "interesting" phrases from text but ignores sentence structure.

After toponym recognition, we associate each recognized toponym with a list of geographic interpretations via name lookup into a *gazetteer*, or database of geographic locations and associated meta-data. Our gazetteer is based on the GeoNames [6] gazetteer, an open gazetteer constructed and maintained by volunteers around the world. The database contains almost 7 million entries, and multiple kinds of meta-data, such as feature type (e.g., country, city, river, mountain), population, elevation, and hierarchical containment information. We store and query a local copy of the gazetteer in a PostgreSQL database.

Once each toponym has a set of interpretations from the gazetteer, we perform toponym resolution to decide on the correct interpretation for each toponym. Our resolution procedure relies on a variety of heuristics inspired by the way humans specify and interpret toponyms. The main heuristics we use resolve toponyms using geographic containment, such as "College Park, Maryland", and prominent toponyms such as country names. These and other heuristics are used in our STEWARD [11] and NewsStand [18] systems. However, consider that these previous methods, and in fact any resolution method intended for full-sized text documents, will struggle with toponym resolution in 140-character tweets. This is because most such methods rely on multiple toponyms in the document providing evidence for each other, usually in terms of geographic distance, document distance, or geographic containment, and tweets normally contain at most only one or two toponyms. Therefore, rather than attempting a full resolution of all toponyms in the single tweet, we defer difficult resolutions to the topic focus step described below, which will consider all possible interpretations for a difficult toponym rather than a potentially erroneous resolution of the toponym. Note, however, at this stage we can also use the artifacts to aid in geotagging, as they are mostly news articles which are much longer than tweets. We can geotag artifacts that are news articles using the geotagger from STEWARD [11].

Using Tweet Metadata.

In addition to the textual content of each tweet, we also collect the meta-data of the user that created the tweet, and in particular the user's source location. This location is stored as free-form text, much like toponyms in tweet content (e.g., "College Park, MD", "Paris", or "England"). As a result, we can perform a much simplified geotagging process that omits the toponym recognition step (since the user's location is the only text under consideration, and it is known to contain a toponym) and directly resolve the location using our containment heuristic for toponym pairs (e.g., "College Park, MD") or prominence heuristic for single toponyms (e.g., "United States"). Of course, users are free to enter whatever location they wish, and they may enter text that is not a toponym at all. However, in most cases, users enter valid toponyms and user locations serve as a useful source of geographic evidence,

especially for tweets that do not contain any toponyms at all.

Computing Topic Focus.

Finally, after analyzing the content and user locations for all tweets in the cluster, we compute a geographic focus for the topic as a whole by ranking the geographic locations in the cluster. One basic measure of relevance used in our ranking is the frequency of occurrence of each geographic location in the cluster. The reasoning is that if a geographic location is important to the topic at hand, it would be mentioned in many tweets and linked articles belonging to the cluster. Also, for each toponym which had not been fully resolved in the toponym resolution step described earlier, we divide the toponym's frequency among all its possible interpretations. Another consideration is that geographic locations that are nearby to each other lend evidence to each other, so we give a higher relevance score to groups of locations that are mutually proximate. We combine these two measures to weight each geographic location in the cluster, and report the top few locations as the cluster's geographic focus.

4.5 User Interface Issues

TwitterStand's user interface is similar to that of NewsStand [18], in the sense that the screen is partitioned into two panes, with the left pane ranking the clusters in a decreasing order of importance, and the right pane containing a map associating clusters with their geographic region of interest. As users interact with the map interface with *pan* and *zoom* actions, the map is constantly refreshed with the most relevant clusters of news stories for the current viewing window, thereby keeping the window filled with clusters of news stories, regardless of position or zoom level. A given view of the map attempts to produce a summary of the clusters of news stories in the view, with the clusters selected using a combination of story significance and geographic spread of their geographic foci. Users interested in a smaller or larger geographic region than that visible on the map have the option to zoom in or out to retrieve more clusters of stories about that region.

The user interface described above is tied to another module that disseminates news as tweets on Twitter. In other words, TwitterStand sends periodic breaking news updates as tweets, such that a tweet contains the URL of its corresponding news topic on the user interface. Using this mechanism, users on Twitter can examine all the tweets that are related to a news topic, as well as the geographic locations, images, videos, keywords, and hashtags associated with them. We will provide an API for TwitterStand by which users can create personal customized TwitterStand accounts, where they can specify the kinds of tweets they would like to receive. For example, users will be able to choose the topic (e.g., general, politics, sports, entertainment, technology, science, and health) and geographic region of interest of news stories, as well as the rate at which TwitterStand should send out tweets, and formatting options such as inclusion or non-inclusion of hashtags, URLs, images, etc.

4.6 Topic Hashtags

An interesting aspect of how tweets are clustered lies in our ability to automatically discover a collection of hashtags corresponding to a news topic by simply aggregating and thresholding all the hashtags present in tweets belonging to a cluster c. For example, Figure 6 shows the hashtags associated with the 2009 Iranian elections, where the size of the font of a hashtag h in the Figure is made proportional to the number of times h appears in the cluster. Note that this clearly illustrates why the online clustering algorithm is superior to methods [7, 15, 19–21] that aggregate tweets simply on the basis of common hashtags, which invariably lead to fragmented and duplicate clusters. Instead, we use these topic hashtags associated with a cluster c in the following two ways. First, the hashtags aid the clustering process by reducing the ϵ values for those tweets and clusters that have one or more hashtags in common. Second, topic hashtags are used as search keys on the Twitter track interface to proactively search Twitter for more tweets belonging to a particular topic.

4.7 Friend Finder

Finding new friends (i.e., more appropriately, updating the set of friends) forms an integral part of TwitterStand. Studies (e.g., [8]) indicate that 10% of the most active users contribute up to 90% of the tweets. Moreover, given that Twitter is a dynamic medium, finding new friends that tweet about and create news. is critical to TwitterStand's success. It is interesting to observe that the BirdDog feed enables TwitterStand to obtain tweets from 200,000 users, which means that we need to have some automated means of finding new friends as well as pruning away those that become inactive. We do this by first pooling all the followers of the users in the Seeders feed, and then identifying the top 200,000 users among them that they have in common. The idea here is that any user who follows many of the users in the Seeders feed is likely to be interested in and tweet about news. As TwitterStand executes, we keep track of how many tweets from each of the users in both the BirdDog and the GardenHose feeds end up clustering with a news topic. If a particular user in the GardenHose feed has contributed a sufficiently large number of tweets to news topics over a given period of time, then we add the user to the BirdDog feed. Similarly, a prolific user in the BirdDog feed will end up being added to the Seeders feed if over time the user has contributed enough number of tweets to news topics. We also discard users both from the Seeders and the BirdDog feed if none of their tweets cluster, which would indicate that they may be inactive, or not so interested in news in that they do not contribute tweets to a particular topic sufficiently often.

5. CONCLUDING REMARKS

We have described a general technique for extracting a concept from a noisy medium. Although we have presented in the context of a news application, our techniques are far more general and can be applied in the extraction of other concepts, including but not limited to sporting events, products such as recalls, diseases such as monitoring swine flu outbreaks, celebrity movements including gossip, keeping up with distributed events with no centralized hierarchy, etc. All that is required to construct any of these systems is to use a different set of Seeders and an appropriate training set for the tweet classifier. The main issue was in dealing with noisy data, and we have described several productive strategies developed for overcoming noise. In particular, even though our input was quite noisy, we manually identified a small sampling of the input that was generally of a good quality, and used it to process the rest of the noisy data. Thus, TwitterStand works primarily due to our ability to understand how to deal with data of different qualities. Note that TwitterStand can be trivially combined with a traditional news aggregator by adding any news obtained by crawlers to the Seeders feed. Moreover, we can now add noisy feeds from many other domains, such as other social networking websites (e.g., Facebook), or blogs, by simply including them as noisy feeds. We used a naive Bayesian classifier to improve the quality of the noisy feeds

Figure 6: Hashtags associated with the cluster of the Iranian election from June 2009.

by throwing away a large portion of the noise tweets, while only losing a very small portion of news tweets. We employed a dynamic corpus to sensitize the classifier to current news. We observe that the sheer enormity of the data means that algorithms will have to be online in nature, which can be challenging. The online clustering algorithm that we presented in this paper is quite fast and robust for mitigating noise. In addition, we described methods for geotagging news, as well as a user-interface for displaying news.

It is important to observe that Twitter, or most likely a successor of it, is a harbinger of a futuristic technology that is likely to capture and transmit the sum total of all human experiences of the moment. Such efforts will result in the development of an intricate understanding of people, their lives, concerns, interests, and opinions. Leaving privacy concerns for the moment, this is a powerful experiment that has the potential to fundamentally change the world we live in. Thus, even though this technology seems a bit far fetched at the moment, and acknowledging the undeniable enormity of privacy and security concerns, we are nonetheless inevitably moving towards such an interconnected world. TwitterStand is an attempt to harness this emerging technology to gather and disseminate breaking news much faster than conventional news media. As Twitter gathers wider acceptance, TwitterStand will carry news in an even wider range of expressions, thoughts, opinions, and artifacts pertaining to news so that it is truly representative of the present moment. In our opinion, TwitterStand in its current form is a good first step towards this goal, and the lessons learned in this paper will lead to further improvements.

6. REFERENCES

[1] E. Amitay, N. Har'El, R. Sivan, and A. Soffer. Web-a-Where: Geotagging web content. In *Proc. of ACM SIGIR*, pages 273–280, Sheffield, UK, July 2004.

[2] CNN. http://www.cnn.com/2009/TECH/07/01/celebrity.death.pranks/index.html. Retr. July 1, 2009.

[3] N. Cohen. Twitter on the barricades: Six lessons learned. http://www.nytimes.com/2009/06/21/weekinreview/21cohenweb.html. Pub. June 20, 2009.

[4] Complete.com – Site profile for twitter.com. http://siteanalytics.compete.com/twitter.com/. Retr. July 1, 2009.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, New York, second edition, 2000.

[6] GeoNames. GeoNames. http://geonames.org/. Retr. June 17, 2009.

[7] HashTags. http://hashtags.org. Retr. July 1, 2009.

[8] B. Heil and M. Piskorski. New Twitter research: Men follow men and nobody tweets. http://blogs.harvardbusiness.org/cs/2009/06/new_twitter_research_men_follo.html. Pub. June 1, 2009.

[9] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *Proc. of SNA-KDD 2007 Workshop on Web mining and social network analysis*, pages 56–65, San Jose, California, 2007.

[10] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ, USA, 2000.

[11] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: architecture of a spatio-textual search engine. In *Proc. of ACM GIS*, pages 186–193, Seattle, WA, Nov. 2007.

[12] M. McGiboney. Twitter's tweet smell of success. http://blog.nielsen.com/nielsenwire/online_mobile/twitters-tweet-smell-of-success. Pub. Mar. 18, 2009.

[13] M. Milian. Twitter sees earth-shaking activity during SoCal quake. http://latimesblogs.latimes.com/technology/2008/07/twitter-earthqu.html. Pub. July 30, 2008.

[14] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, NY, 1997.

[15] Monitter. http://monitter.com/. Retr. July 1, 2009.

[16] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[17] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, pages 1–20, Boston, MA, Aug. 2000.

[18] B. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: A new view on news. In *Proc. of ACM SIGSPATIAL GIS*, pages 144–153, Irvine, CA, Nov. 2008.

[19] TweetMeme. http://tweetmeme.com/. Retr. July 1, 2009.

[20] Twist. http://twist.flaptor.com/. Retr. July 1, 2009.

[21] Twitter Search. http://search.twitter.com/. Retr. July 1, 2009.