Enhancing Local Live Tweet Stream to Detect News

Hong Wei
Department of Computer Science
University of Maryland
College Park, Maryland 20742
hyw@cs.umd.edu

Jagan Sankaranarayanan
UMIACS
University of Maryland
College Park, Maryland 20742
jagan@umiacs.umd.edu

Hanan Samet
Department of Computer Science
University of Maryland
College Park, Maryland 20742
hjs@cs.umd.edu

ABSTRACT

Twitter captures invaluable information about real-world news, spanning a wide scale from large national/international stories like a presidential election to small local stories such as a local farmers market. Detecting and extracting small news for a local place is a challenging problem and the focus of this work. The main challenge lies in identifying these small stories that correspond to a local area of interest, which are typically harder to detect compared to national stories in the sense that there may be just a handful of tweets about a local story. A system, called Firefly, is proposed that overcomes the data sparsity and captures thousands of local stories per day from a metropolitan area (e.g., Boston). The key idea lies in combining the enhancement of a local live tweet stream in Twitter, the identification of "locality-aware" keywords, and using these keywords to cluster tweets. Experiments show that the proposed system has a significantly higher recall over a set of representative local news agencies, and at the same time, outperforms the baseline approach TwitterStand. More importantly, the results also demonstrate that our system, by utilizing the enhanced local live tweet stream, discovers much more local news than the methods working only on geotagged tweets, i.e., those with embedded GPS coordinate values.

CCS CONCEPTS

•**Information systems** →*Data streaming; Clustering;*

KEYWORDS

Twitter, Live Tweet Stream, News Detection, Local News, Geotagging, Apache Spark

ACM Reference format:
Hong Wei, Jagan Sankaranarayanan, and Hanan Samet. 2018. Enhancing Local Live Tweet Stream to Detect News. In *Proceedings of 2nd ACM SIGSPATIAL Workshop on Analytics for Local Events and News, Seattle, WA, USA, November 6, 2018 (LENS'18),* 10 pages.
DOI: 10.1145/3282866.3282868

1 INTRODUCTION

The popularity of Twitter arises from its capability of letting users promptly and conveniently contribute tweets on a wide variety of subjects such as news, stories, ideas, and opinions. As a result, with people discussing what is happening outside in the real world by posting tweets, an invaluable amount of information on the real world news is hidden in Twitter. Therefore, many researchers have devoted remarkable efforts to discover this knowledge. For example, TwitterStand [1–5] is a news tweet processing system that aggregates tweets from a sparsely sampled tweet source to detect news.

However, this approach is too brute-force for smaller-scale local news where every single tweet matters because such types of news may only span a very limited number of tweets. Figure 1 shows a news story about the "Westborough Education Foundation" that happened at around 6:30 PM on Oct 24, 2016 at Westborough, MA. We only found 6 tweets (8 if retweets are included) about this news by the time we captured the screenshot, and none of them is geotagged, i.e., containing a pair of geographical lat/lon coordinate values. No access to full tweets in Twitter makes data sparsity pervasive in Twitter's publicly accessible tweets, and further compromises the possibility of collecting all 6 tweets about this news. The challenge in capturing such news lies in being able to find these tweets, cluster them into a news story, and subsequently displaying it on a map.

Figure 1: A local news in Westborough, MA on Oct 24th, 2016

In this paper, we are interested in detecting news (a set of tweets) that are being discussed by local people from a given place (e.g., Boston city), and meanwhile emphasizing on finding local news. The term "local news" refers to a news event that happens at or is of great interest to the given place. For instance, the news story in Figure 1 may only be of interest to the local community and not much further beyond. Local news can sometimes escalate to be of national/international interest such as when it is dramatic (e.g., Boston Marathon bombing in May 2013). We want to capture both these types. Other national and international stories that are discussed by local people (e.g., a presidential election) are also in by providing a local perspective to larger news stories. Our focus is primarily the former two classes of stories, and later in our experiments we evaluate how well we do with and without considering these national and international news stories.

Identifying the news stories that are of great interest to a place requires a combination of approaches. It requires first finding users that reside and tweet about our place of interest. To find such users, we implement an efficient online social network-based Twitter user geotagging approach, which is to approximate the location of a Twitter user by examining the publicly-known locations of his social friends (neighbors). The publicly-known location, termed the *profile-location*, is provided in a Twitter user's profile, but is only available for around 20% (in our case, 32%) of Twitter users [6]. This makes the procedure of geotagging Twitter users indispensable in our system. With the help of this scheme and its efficiency, our system, Firefly, keeps trying to find as many as possible active Twitter users from a given area and putting their posting statuses (tweets) to a local live tweet stream to largely increase its number of local tweets.

LENS'18, Seattle, WA, USA

DOI: 10.1145/3282866.3282868

Next, there is a larger problem of clustering these local tweets so that news can be captured. For example, some features like bursty words [7, 8] or TF-IDF [1, 9] that are commonly used to group tweets together might not work well with small local news because such news span over a very limited number of tweets, and thus words in them hardly bring about burstiness or yield distinguishing TF-IDF scores. Another category of methods that only exploit geotagged tweets such as [10, 11] would simply miss the news example in Figure 1 because few of its tweets are geotagged.

In this paper, we utilize an idea of "locality-aware keywords" to capture the changes in word-usage patterns caused by a news of limited local interest from the perspective of individual people. Essentially, the locality-aware keywords in each tweet are a set of words that are used only recently by this tweet's publisher and also at the same time only appear in a limited number of other Twitter users' tweets. Such locality-aware keywords correspond to the aspects of a local news being "novel" as its nature of being new, as well as having a small spread span among Twitter users. Take the one in Figure 1 for example, "Westborough", "Education", "Foundation", "Trivia" and "Bee" are considered as locality-aware because they are new words used by this set of people.

To capture news from the enhanced local live tweet stream, we keep identifying and updating locality-aware keywords from tweets that are in the latest 6-hour sliding time window (The choice of 6-hour window is in recognition that television media usually has four times of locally-oriented news broadcast in one day and thus is an appropriate lifetime of local news), and group tweets together that share at least a number of locality-aware keywords to form news clusters. Finally, in the UI, a Twitter timeline is created to post the news we detect. We also estimate the geographic focus of detected news (tweets clusters) to display them on maps.

The main contribution of this paper is summarized as follows:

- We implement an efficient online Twitter user geotagging procedure on Apache Spark, which takes less than 3 seconds to geotag Twitter users appearing in 1000 tweets. Such efficiency is essential to maintaining the liveness of the enhanced local tweet stream and furthermore the timeliness in news detection.
- Our enhanced local live tweet stream easily covers up a typical metropolitan area. For example, in Boston, we are tracking 176K Twitter users, which is considered sufficient since Boston has a population of 646K[1] and that one-fifth of the USA population are active Twitter users[2].
- The design of locality-aware keywords emphasizes the word usage characteristics of small, local news from the view of Twitter users who are discussing them (e.g., only a small number of people talk about them and use words they didn't use before).
- We evaluate our system against a set of representative local news agencies as well as a few baseline approaches. The results show we achieve the highest news coverage and at the same time, outperform the baseline approaches. More importantly, our method detects hundreds of more local news in comparison with the methods that solely utilize the existing Twitter's publicly available tweet stream.

The rest of this paper is organized as follows. Section 2 summarizes the related work. Section 3 details the design and implementation of our system. Section 4 describes the experimental evaluation of our methods. Section 5 contains concluding remarks as well as directions for future work.

[1] http://www.census.gov/popest/about/terms.html

[2] https://www.statista.com/statistics/274564/monthly-active-twitter-users-in-the-united-states/

2 RELATED WORK

There is a large body of related work that deals with extracting useful patterns (e.g., news, events) from social media, Twitter in particular. Two recent surveys Atefeh and Khreich [12], and Abdelhaq [13] provide an excellent description of different techniques. We review some of the related work that deals specifically with the problem of detecting local events. There are two broad categories of methods for taking location into consideration when performing detection tasks, namely: *location-anchored* and *event-anchored*. The essential difference is whether event or location is the primary clustering key. For example, event-anchored methods first detect an event and then determine its location, while location-anchored methods examine if an event happens at a certain location.

Location-Anchored Methods: Among the location-anchored methods are two popular approaches: *model dimension extension* and *geographical space tessellation*. Model dimension extension treats geographical information as an additional variable to the existing models. For example, in calculating similarity between documents while performing a clustering algorithm, geographical distance between tweets can be incorporated in the clustering algorithm [14] to form potential events [11, 15–17]. Hong et al. [18], Zhou and Chen [19] and Wei et al. [20] treat geographical regions as latent variables in their generative topic model.

Geographical space tessellation fills the map with small, non-overlapping cells. The motivation here is that local news or events, which usually have an limited geographical area impact, should fall in the same or nearby cell(s). *Grid* tessellation is the simplest yet most commonly used way of subdividing the geographical space into small equal-sized cells [21–25]. In reality however, the geographical distribution of social media documents is not homogeneous, frequently requiring the consideration of adjacent cells in the analysis. To alleviate this issue, a few strategies are proposed including resizing the cells, connecting nearby cells if they share similar features, or utilizing an adaptive hierarchical tessellation structure [26]. For example, Krumm and Horvitz [10] et al. discretize the space with a hierarchical triangular mesh. Magdy et al. [27, 28] describe a system called Mercury for querying top-k spatio-temporal queries on microblogs in real-time using a pyramid structure.

After tessellation, the social media documents or features are aggregated into small cells according to their inferred geographical information. Next, an intuitive way to detect the existence of any anomaly at a specific location is to count aggregated documents or other feature entities like keywords to see if their number exceeds a certain threshold. Counting, however, is easily plagued by distribution heterogeneity both temporally and spatially. Therefore, various anomaly detection techniques have been explored. For example, Xu et al. [29] employ a probabilistic model that recovers spatio-temporal signals using a Poisson point process estimation to deal with sample bias and data sparsity problems. Others exploit the usages of a discrepancy paradigm which compare between previous data (to build up a baseline) and the newly observed data [10, 22, 30, 31].

Nevertheless, such methods have heavy dependence on the availability of social media documents containing geographical information. Such geographical information, however, is very rare in Twitter, with geotagged tweets accounting for less than 1% [24, 32, 33]. Some works have proposed to estimate a geographical location for a non-geotagged tweet. The intuitive approach towards this problem is to geotag nominal locations (place names) embedded in the content of a microblog to get its possible longitude/latitude coordinates by aligning against existing gazetteer databases or services, e.g. GeoNames [13, 24, 34, 35]. While another set of works try to assign a

geographical location to a non-geotagged tweet by its poster's location [32, 36, 37] , which might be initially estimated through a social network based procedure [6, 38–41] or content-based methods [42–51].

Event-anchored Methods: This class of methods, after identifying events, leverages an additional step of spatial analysis to determine the locations where they are happening. For example, TwitterStand [1], after clustering tweets to identify events, estimates each news cluster's geographical focus by making use of both geographical information in the content of the tweet and by the source location of the users. This geographic focus is computed as a whole by ranking the geographic locations in the cluster. One basic measure of relevance used in their 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 event at hand, the it would be mentioned in many tweets and linked articles belonging to the cluster. In addition, they also give a higher relevance score to groups of locations that are mutually proximate by considering that geographic locations that are nearby to each other lend evidence to each other. To infer and track the location of detected earthquake or typhoon events, Sakaki et al. [52] resort to Kalman filtering and particle filtering by treating each Twitter user as a sensor.

Even though all event related documents are exploited (not just the ones with location information) in event-anchored methods, their data sources still suffer from sparsity to detect small, local events. For example, TwitterStand's data source, which then claimed to sample around 10% of all tweets but now only 1%, is still too few for small-scale events that might only span $3 \sim 5$ tweets in total.

Therefore, realizing it is the local data sparsity that undermines the opportunities for researchers to discover small-scale events in Twitter, our system proposes to enhance the public local live tweet stream for an area by i) identifying as many Twitter users as possible that are from that area and then ii) tracking the tweets that they publish in real-time. Weng and Lee [53] similarly track a number of users in Singapore to detect news but only at a small scale, i.e., 1K Twitter users. In contrast, we identify and track 176K users in Boston. Our work is also different from Albakour et al. [54], which directly chooses several areas in London to collect tweet data, and tries to detect events for each of these areas separately [54]. Their method doesn't solve the problem of local data sparsity by using Twitter's Streaming API, i.e., statuses/filter with parameter "locations", in our experiment, is still very sparse and thus makes a very limited contribution to local news detection.

3 SYSTEM

In this section, we present the design and implementation of our event detection system, Firefly, as illustrated in Figure 2. Including the User Interface, Firefly consists of 5 major modules, which are described below sequentially.

3.1 Online Twitter User Geotagging via Spark

The goal of this module is to keep estimating the geographical locations for more Twitter users, and thus to maintain a large pool of geotagged Twitter users. In so doing, for a given geographical area like the Boston Metropolitan area, our system can easily retrieve a large body of Twitter users in it. Tracking tweets posted by these users significantly enhances our local live tweet stream.

The motivation behind geotagging Twitter users is that the profile-location information for specifying where a Twitter user comes from is only sparsely available in public data. Therefore, inspired by studies [55, 56] that online social friendships are often formed over short geographic distances, a social network-based Twitter user geotagging method is proposed in [6], which approximates a user's location by examining the publicly-known locations of his online friends (neighbors). This method is reported to have state-of-the-art city-level accuracy when geotagging a large body of Twitter users and, more importantly, doesn't require sophisticated natural language processing in comparison with tweets content-based methods [42–46], making it more suitable for online geotagging.

To be specific, the social network-based geotagging problem is addressed from the point of view of solving an optimization problem, i.e., inferring user locations is solved by finding

$$\arg\max_{\boldsymbol{f}} \boldsymbol{g}(\boldsymbol{f}) \text{ s.t. } f_i = l_i,\ \forall i \in L \tag{1}$$

where $\boldsymbol{f} = (f_1, f_2, f_3 ... f_n)$ represents location estimation for each user $1...n$, and L denotes the set of users who opt to make their locations l_i public. The total variation is formulated as $\|\nabla \boldsymbol{f}\| = \sum_{ij} w_{ij} * d(f_i, f_j)$, where $d(\cdot,\cdot)$ measures geographical distance.and w_{ij} weighs the friendship between user i to user j, which essentially reflects how many times user i reciprocally interacts with j such as retweeting, mentioning etc. Note that, an edge between i and j in the graph is bidirectional and only formed if both i and j have actively initiated at least one interaction with each other, and we use reciprocal neighbors or friends to term such edges.

The above minimization problem could be solved by calculating, for each user, the *L1-multivariate median* from his reciprocal neighbors' locations. The value of *L1-multivariate median* [57], which acts as a user's estimated (geotagged) location and is denoted by l^{L1mm}, essentially finds a point that minimizes the sum of its distances to the users' reciprocal neighbors. For a user j, its *L1-multivariate median* l_j^{L1mm} is mathematically defined as,

$$l_j^{L1mm} = \arg\min_{l} \sum_{l_i \in L_j} w_{i,j} * d(l, l_i) \tag{2}$$

where, L_j contains the locations of j's reciprocal neighbors. In the implementation, Equation 2 can be solved through a coordinate descent procedure. Upon completing the calculation of location estimate, for a user j, how far l_j^{L1mm} deviates from his reciprocal neighbors determines whether he accepts l_j^{L1mm}. This deviation, called *Geographical Dispersion*, is defined as,

$$GD(L_j) = median_i\ w_{i,j} * d(l_j^{L1mm},\ l_i) \quad \text{s.t. } l_i \in L_j \tag{3}$$

For example, user j will accept his estimated location if $GD(L_j)$ is less than a given threshold, γ. In our experiments, we set $\gamma = 100$ km, which is suggested as a suitable trade-off between geotagging coverage and accuracy for the city-level scenarios [6].

One drawback of [6] lies in indiscriminately utilizing all available location information from reciprocal friends to calculate a candidate location estimation in Equation 2, while some of them might be noisy points as discussed in [39]. For example, as illustrated in Figure 3 (where each circle represents a reciprocal friend and the number in each circle denotes the weight to that friend), a user from Boston has 9 reciprocal friends with available location information, 4 of them (red circles) are relatively far away from Boston and can be seen as noisy points or outliers because incorporating them into Equation 2 is likely to yield a location estimation that does not satisfy the geographical dispersion constraints γ, and thereby fails to geotag this Twitter user. Inspired by the observation in [39] that the location of a friend is usually more reliable if a user has multiple friends from that or nearby location, we propose a single-linkage-clustering based outlier removal procedure to get rid of potential noisy points. As presented in Algorithm 1, this procedure works as follows. Take the locations of a user j's reciprocal neighbors, L_j, as the input, we first perform the Single Linkage Clustering with

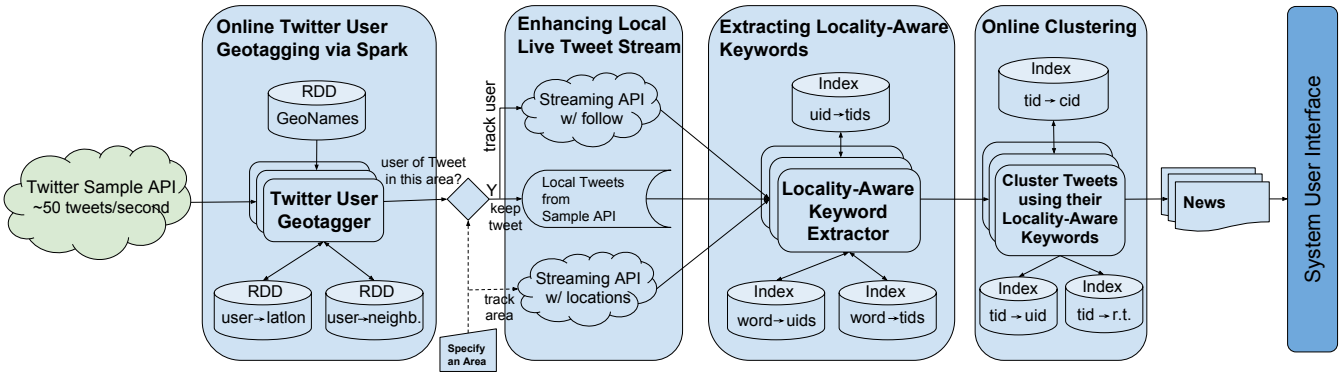


Figure 2: System Architecture of Firefly

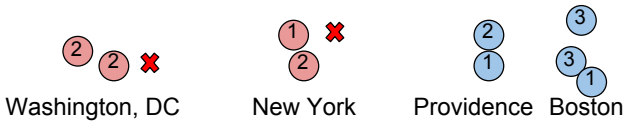


Figure 3: An illustration of outliers in the locations of friends.

geographical dispersion γ being the distance threshold. During the clustering, two location points in L_j that are within γ are grouped into the same cluster; and two clusters are merged if a pair of points from each of them are within γ. Next, we select the cluster with maximum sum of weights and use it as new L_j in Equation 2 to calculate the location estimation.

Algorithm 1: Outlier Removal

Input: The locations of user j's reciprocal neighbors $-L_j$
Output: A list of locations after removing outliers $- L'_j$

1: A set of clusters $C = \{C_1, C_2, C_3...\} \longleftarrow$ Single Linkage Clustering on L_j with distance threshold γ;
2: $L'_j = \arg\max_{C_k \in C} \sum_{l_i \in C_k} w_{i,j}$

Another improvement over [6] is a minimum size constraint for L'_j because too few location information might be considered as weak evidence [40]. In other words, we refuse to calculate l_j^{L1mm} for user j if $|L'_j|$ is less than a given threshold λ. The experimental results show that such a constraint for λ might effectively improve the accuracy of geotagging in the sparse social networks where users have only a few reciprocal friends, especially the ones with locations.

Publicly-Known Locations of Twitter Users In Twitter, there are two sources to know a user's location: profile-location or the GPS coordinates embedded in his tweets. The profile-location is often in the form of place names like "College Park, MD" and can be aligned with databases like GeoNames to decode its geographical latitude/longitude coordinates. In order to assign a unique pair of latitude/longitude coordinates, for a user having multiple GPS points available in his tweets, we compute the *L1-multivariate median* for these points and similarly check the geographical dispersion to decide whether to use this median or not. At last, for a Twitter user who has a valid profile-location as well as a valid *L1-multivariate median* calculated from his tweets, we opt to use his profile-location if this location is within γ of the median; otherwise, his two sources of location information seem to be conflicting with each other and thus wouldn't be utilized. Algorithm 2 outlines our online Twitter user geotagging procedure, which utilizes a streaming computing platform Spark Stream by maintaining 4 RDD variables [58–61]. Resilient Distributed Dataset (RDD), is a distributed memory abstraction which gives Spark the ability to perform fast in-memory map-reduce operations. IndexedRDD extends *key-value* RDD by enforcing key uniqueness and pre-indexing the entries for efficient look-up operations. In practice, RDD could be seen as a table in the database. The IndexedRDD variable for GeoNames, location→latlon, is to align the profile-location, e.g., "Boston, MA", to decode its latitude/longitude coordinates, e.g., [42.3584, -71.0598]. The RDD variable, location→user keeps a reversed index from a user to his profile-location to perform join operation in Spark. The RDD variable, user→twGPS, stores the GPS coordinates embedded in a user's tweets. The RDD variable, user→neighb., stores the neighbor-ships. Finally, the IndexedRDD variable, user→latlon, caches the geotagged user to retrieve users in an area.

To quickly start-up our online geotagging procedure, i.e., fill in the RDD variables, we boost our algorithm with one year of tweets data collected from the Twitter Sample API statuses/sample. We discretize this live tweet stream into 23-second intervals using DStream in Spark to perform the online Twitter user geotag. For an incoming user, we first look-up his geographical coordinates in user→latlon; if this fails, then we try to align his profile-location (if provided) to GeoNames; otherwise, we retrieve a list of his reciprocal neighbors' locations to estimate his location.

Algorithm 2: Online Twitter User Geotagging via Spark

Input: Twitter's Public Live Tweet Stream – G; 1 year of tweets collected from Twitter Sample API – T
Output: Geotagged Twitter Users

1: **Boosting Phase**:
a. Load location→latlon from GeoNames; and extract location→user, user→neighb., and user→twGPS in T;
b. user→latlon $\longleftarrow$ location→user join location→latlon;
c. Update user→latlon with users whose lat/lon can be calculated upon user→twGPS using Equation 2 and 3.
2: **Online Geotagging**:
a. Init a Spark DStream D in G w/ a 23s time window;
b. Update user→neighb. and user→twGPS with D;
c. **for each** user u in D who is not in user→latlon and fails to align profile-location in location→latlon and fails to calculate a lat/lon in user→twGPS **then do**
i). get u's reciprocal neighbors' coordinates L_u by joining u, user→neighb. and user→latlon;
ii). $L'_u \leftarrow$ Outlier-Removal(L_u)
iii). calculate l_u^{L1mm} by L'_u **if** $|L'_u| \geq \lambda$;
iv). u accepts l_u^{L1mm} **if** $GD(L_u) \leq \gamma$;
end for

3.2 Enhancing Local Live Tweet Stream

Given a geographical area, this module tries to collect as many tweets as possible from three sources: two of Twitter's statuses/filter

Streaming API – "follow" and "locations"[3], and tweets filtered from another Twitter Sample API statuses/sample [4], which returns a small random sample (usually 1%) of all public tweets. The Statuses/filter "follow" real-time returns the postings of a list of specified Twitter users (5,000 at most) as they publish tweets; while "locations" tracks the tweets falling in a geographical area either according to tweet's embedded GPS coordinates or place names.

After specifying an area A, our system first retrieves a set of Twitter user who fall inside A using IndexedRDD variable user→latlon built in Section 3.1, and collects their live tweets via statuses/filter "follow". Our experiments in Section 4.2 show that doing so dramatically increases the number of local tweets and thereby boosting the number of detected local news in our system. Meanwhile, statuses/filter "locations" is also initiated to collect tweets with embedded GPS coordinates or place names falling inside A. Finally, we also keep one's tweets captured from Twitter Sample API if he is from A. As the system runs, we also keep following the newly found Twitter users belonging to A to track their real-time tweets.

3.3 Extracting Locality-Aware Keywords

"Hot" news or events in Twitter often cause, temporally or spatially, noticeable changes (e.g., word usage and increase in the number of related-tweets) in Twitter, thereby encouraging the exploitation of anomaly detection techniques such as the discrepancy paradigm [10, 22, 30] which makes a comparison between previous data (to build up a baseline) and the newly observed data to discover anomalies. These techniques are often addressed only from the perspective of detecting anomalies in the entire set of tweets (e.g., a set of tweets collected or aggregated together either geospatially or temporally), and in so doing might miss small-scale local news. Again, the data sparsity might make the problem worse. For example, to detect the news in Figure 1 is like finding a needle in a haystack from tweets because such a story, with only 6 tweets, hardly affects the word usage pattern in that evening at Westborough, MA.

However, if we look at the news story in Figure 1 from the view of individual people involved, such a small news poses noticeable changes in their word-usage pattern. For example, "Westborough Education Foundation Trivia Bee" are recently used words for 3 the Twitter users in that afternoon.

Therefore, given the sparsity of local news tweets, we utilize the following observations to capture such news. First, instead of looking for bursty or frequently used words with respect to a corpus of tweets from different Twitter users, we focus on the newly-used words with respect to the tweets from a single Twitter user. In other words, for a Twitter user, we are only interested in the words recently used by him. Such newly-used words correspond to the aspect of local news being "novel" as its nature of being news. Second, to reflect the aspect of local news being discussed by a limited number of people, we look for the words that are only used by a limited number of Twitter users, instead of the ones intensively used by people. Therefore, for a given tweet, we identify the words exhibiting the above two properties and call them *locality-aware keywords* in the sense that they are aware of the characteristics of local news. For example, in Figure 1, "Westborough", "Education", "Foundation", "Trivia" and "Bee" are considered as locality-aware because they are new words used by this set of people.

Inspired by this, we recognize a word (only non-stopwords) in a tweet to be locality-aware by looking at 3 measures: how many times this tweet's publisher uses it, how many other users are using it and how many tweets contain it. To ensure the local news we detect

[3]https://dev.twitter.com/streaming/reference/post/status/filter

[4]https://dev.twitter.com/streaming/reference/get/statuses/sample

Algorithm 3: Online Extracting Locality-Aware Keywords and Online Clustering to Detect News

Input: the latest 6-hour sliding window in enhanced local live tweet stream – S; the locality-aware constraints – R_{TF}, R_{DF} and R_{CF}; the threshold values m, n and r

Output: news, i.e., clusters of tweets

1: load hash variables uid→tids, word→uid, word→tids, tid→uid, tid→cid, tid→r.t. in last time window;

2: **while** true **do**

a. pull a tweet from S, get its non-stopword tokens W, tweet id t, and user id u;

b. Locality-Aware Keywords $W_L \longleftarrow \emptyset$

c. **Extracting Locality-Aware Keywords Procedure:**

for each word $w \in W$ **do**

i). calculate TF_w, TF'_w, DF_w, CF_w, CF'_w from uid→tids, word→uid, word→tids;

ii). $W_L \longleftarrow W_L \cup \{w\}$ **if** TF_w, TF'_w, DF_w, CF_w and CF'_w meet with R_{TF}, R_{DF} and R_{CF};

iii). update word→uids, word→tids by inserting w and its corresponding u and t;

end for

d. update uid→tids by inserting u and t;

e. update tid→r.t. by inserting t and retweet number;

f. **Online Clustering Procedure:**

for each $W_L^m \in$ subsets of W_L with size m **do**

i). retrieve Q – the ids of tweets containing all words in W_L^m, from word→tids;

ii). retrieve the user set U in Q using tid→uid;

iii). **continue if** $|U| < n$;

iv). extract the largest group of tweets C from Q with the same cluster id c (a null c means that the tweets in C haven't formed a cluster yet);

v). calculate RT_C, which is the sum of retweet number of each tweet in C, using tid→r.t..

vi). **if** $|C| \geq \lceil \frac{|Q|}{2} \rceil$ **and** $|U| \geq \lceil r * RT_C \rceil$ **then**

$c \longleftarrow$ generate a new id **if** c is null;

assign t to cluster c in hash tid→cid;

report t as a *news* tweet to system UI;

break;

end if

end for

g. Remove obsolete tweets from uid→tids, word→tids, tid→uid, tid→cid and update word→uids;

end while

are up to date, all these measures are computed in the latest 6-hour sliding time window from the enhanced local live tweet stream. If we treat a user's tweet as a sentence, then all his tweets in time order form a document, and all the tweets in the latest time window consist of a corpus. This is different from the idea of TF-IDF used in [1, 9] which treat each single tweet as a document.

We term the above 3 measures as *term frequency*, *document frequency* and *corpus frequency*, i.e., *TF*, *DF* and *CF*, respectively. Here we assume that a word appears at most once in a tweet (or counts only once if more), which is reasonable given the 140-character limit. For a given word w in the tweet posted by user u, these measures are computed as: $TF_w = |T_u \cap T_w|$, $DF_w = |U_w|$, and $CF_w = |T_w|$. T_u denotes the tweets of user u, T_w denotes the tweets containing word w, U_w denotes the users who recently used the word w. Our heuristic is that, in order for a word w to be locality-aware, it should have a smaller TF_w (i.e. how many times it has been used recently by a Twitter user), which indicates that word w might be newly used by this user and thereby captures a news's "novelty"; DF_w (i.e., how many Twitter users have been using word w recently)

should have a limited range (like [3, $\frac{|U_S|}{20}$] specified in parameter settings in Section 4.4.1, where U_S is the set of Twitter users), to reflect the local news's characteristic of having a limited spread among people; and also CF_w should be small to avoid commonly used words like "day" and "people" etc. In our implementation, to account for the heterogeneity of the rates of publishing tweets for different users and for the number of tweets collected at different times and different places, we also use the relative frequencies of TF_w and CF_w, i.e., $TF'_w = \frac{|T_u \cap T_w|}{|T_u|}$, $CF'_w = \frac{|T_w|}{|T_S|}$, where T_S represents all current tweets. The constraints for TF, TF', DF, CF and CF' — denoted by R_{TF}, R_{DF} and R_{CF} — are discussed in Section 4.4.1.

3.4 Online Clustering to Detect News

As presented in Algorithm 3, we take into account the following two aspects to group tweets together. First, the tweets need to share at least a number m of locality-aware keywords to be grouped together. Second, at least n different Twitter users must exist in a cluster. Existing methods usually neglect the importance of these two aspects. For example, GeoBurst [11] measures the semantic similarity between two tweets by performing random walks on their keyword co-occurrence graph to calculate the average probability that one tweet reaches another. However, without requiring a minimum number of keywords in a tweet, two tweets containing and sharing very few keywords could be mistakenly considered semantically coherent even if they are not on the same topic. In addition, TwitterStand [1] groups tweets together as long as they are similar enough in the TF-IDF vector space and in so doing, might form noisy clusters out of a single Twitter user's repeated tweets.

Therefore, in our method, to cluster an incoming tweet, we first retrieve a set of tweets sharing at least m locality-aware keywords. If these tweets were contributed by less than n Twitter users, or the majority of the tweets don't locate in the same cluster, then we don't group this new tweet and try another set of m locality-aware words. We also require that a news spreads among more local people. In Twitter, the spread extent of a tweet is provided by its retweet number, i.e., how many other Twitter users retweet it. We now define, for a given news cluster C, its spread extent RT_C to be the sum of the retweet number of each tweet in it. And the local spread ratio $spread_{local}$ is computed by $\frac{|U|}{RT_C}$, where U is the users contributing to C. In our experiments, we set $spread_{local} \geq r = 0.4$ to account for the local tweets that we might not capture.

The details of calculating the above measures are presented in Algorithm 3. Generally, Firefly uses 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 for real-time detection of news from a local live tweet stream.

3.5 System User Interface

As shown in Figure 4, our user interface consists of two parts: a Twitter Timeline[5] and a Google Map based Web Application [1]. The Twitter Timeline allows a user to view a list of tweets collected for various purposes, such as real-time monitoring of a Twitter user's updates or searching for the latest tweets on a specific topic. Therefore, in order to demonstrate the latest news that we detect in real-time, a Twitter Timeline [6] is created via Twitter Collections API, which is very convenient for other Twitter users to view and even subscribe to. Note that the Collections API only allows for a user to retain a few thousand of tweets and automatically delete the oldest ones if it has too many tweets. To display the events that we detect

[5]https://support.twitter.com/articles/164083

[6]https://twitter.com/bostonnewslocal/timelines/878280225074950144

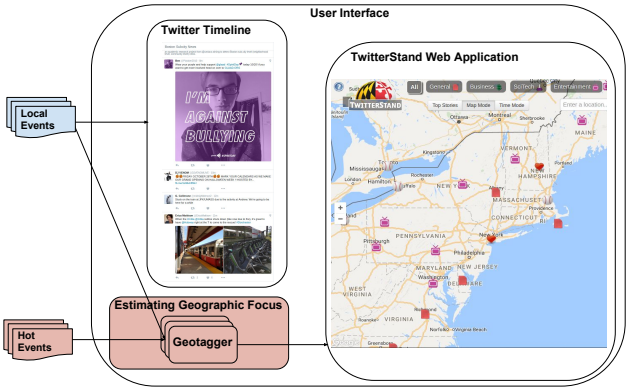


Figure 4: System User Interface

on the Google map-based web application, we utilize a procedure to estimate the geographical focus for a news cluster in [1], which makes use of both the geographical information in tweet content and the source location of the users in an event cluster. After geotagging an event cluster, the Google map-based web application displays a marker for this event at its coordinates.

4 EXPERIMENTS

4.1 Online Processing Settings and Efficiency

Our system adopts sliding time window techniques to meet the demand for online processing of a live tweet stream. The experiments are evaluated on a Spark cluster of 5 computing nodes where each node has two 6-core Intel Xeon E5-2620 v3 CPUs and 128GB RAM.

For Online Geotagging, we utilize Spark Stream to discretize the live tweet stream from the Twitter statuses/sample API into intervals of 23 seconds, which is the average time to accumulate 1000 tweets. Similarly, a 6-hour sliding time window is applied on the enhanced local live stream for locality-aware keyword extraction and online clustering. The 6-hour window size is intuitively set in recognition of the fact that television media usually has four times of a locally-oriented news broadcast in one day. The day of Jan 16, 2017 is chosen to evaluate our system for news detection with respect to the Boston metropolitan area i.e., the rectangle area [42.008339, -71.803026, 42.732923, -70.577545].

In our experiments, we find the major overhead is the Boosting Phase in Algorithm 2, which takes around 76 minutes to finish. But this procedure runs only once to start up the system and does not affect the timeliness of subsequent procedures. After the Boosting Phase, the online geotagging procedure takes an average of 3 seconds to process 1,000 tweets from the Twitter statuses/sample API, and geotags an average of 47 unknown-location Twitter users per second. Afterwards, Algorithm 3 processes 70 tweets per second on average (which is also the approximate arriving rate of tweets in enhanced local live stream) and reports about 3 tweet clusters per minute.

4.2 Twitter User Geotagging via Spark

4.2.1 Boosting Dataset. To boost the startup of geotagging Twitter users, we utilize a set of tweets collected between 09/2015 and 09/2016. This dataset consists of 2,876,822,081 tweets, 102,382,292 users and 824,303,126 pairs of neighbor-ships. Among these users, 31,250,047 have valid location source (successfully aligning profile-location to GeoNames or having embedded GPS coordinates) and are used to build-up the variable user→latlon. Accordingly, variable user→neighb. builds from the extracted neighbor-ships. Filtering down to only reciprocal neighbors, we have a reciprocal graph of 24,946,962 vertices (8,787,152 of them have lat/lon coordinates) and 54,550,871 bidirectional edges.

4.2.2 Effectiveness. In lack of a ground-truth for Twitter users' locations, we exploit the *boosting dataset* to evaluate the effectiveness on coverage and accuracy. Specifically, for the 8,787,152 Twitter users with lat/lon coordinates in the reciprocal graph built in Section 4.2.1, their lat/lon coordinates are obtained from their profile-location or GPS coordinates in their tweets, and are thus treated as ground-truth. We then perform a leave-p-out validation by randomly sampling 10% (i.e., 878,715) of these Twitter users to evaluate the coverage and accuracy. The coverage is to calculate how many Twitter users in the sampling set would get geotagged, while accuracy is to calculate the mean distance error between the ground-truth and their estimated location.

Our experiment shows that with $\gamma = 100$ km, $\lambda = 2$, 13.6% (i.e., 119,505 out of 878,715) test users get geotagged with a mean error of 228.66 km and a median of 27.93 km, which as shown in [6], is accurate at city-level for majority of test users.

Effect of Outlier Removal We now exclude the step of outlier-removal in Algorithm 2 to geotag the 10% test Twitter users with γ fixed at 100 km and λ at 2. This brings us a lower 7.1% coverage with a larger mean error of 279.81 km, showing that removing outliers significantly increases the chances for test users to get successfully geotagged while without compromising accuracy.

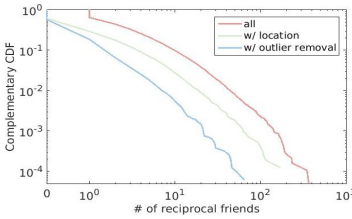


Figure 5: CCDFs.

λ	Coverage (%)	Error (km)
1	53.3%	1900.54
2	13.6%	228.66
3	6.5%	251.34
4	3.9%	213.62
5	2.5%	191.65
10	0.6%	234.72
20	0.1%	187.16

Table 1: Effect of λ.

Effect of λ (the Minimum Number of Reciprocal Friends with Valid Locations) We first plot the distributions on the number of reciprocal friends of these 10% Twitter users in Figure 5, as well as the ones with locations and the ones that have survived from the outlier removal step. Figure 5 shows that lots of the Twitter users have very few reciprocal friends that have locations. In such a sparse reciprocal graph, it may not be fair to decide the location for a Twitter user only based on very few of his friends locations.

To investigate the sensitivity of the minimum constraint parameter λ in Algorithm 2, we set $\gamma = 100$ km and use different values of $\lambda = \{1, 2, 3, 4, 5, 10, 20\}$ for the 10% sampling test users and list the corresponding coverage and mean errors in Table 1. The results show that although $\lambda = 1$ is able to geotag more than half of the test users, it brings about an acceptably large error; $\lambda = 2$ seems to reach the best trade-off point between coverage and accuracy; while larger λ values have similar accuracy, they have relatively low coverage.

4.3 Enhanced Local Live Tweet Stream

At the start of the day on Jan 16, 2017, 176,007 users are found in the input Boston bounding box. Among them, 101,409 provide valid location source (profile-location or GPS), and the remaining 74,598 are geotagged using Algorithm 2. Following these two sets of users to track their real-time postings comprises of the two sources of Streaming API w/ "follow" I and II as listed in Table 2, respectively[7].

Table 2 first shows how many local tweets (i.e., the tweets that fall in the given area or are published by people there) as well as how many cluster tweets (i.e., the tweets that compose the detected clusters) each source contributes to our enhanced local live tweet stream. The table shows that Twitter Streaming API statuses/filter "follow" (I and II) yield the most of tweets. More importantly, Table 2 further shows that tracking Twitter users who don't have valid location sources also make significant contributions just like tracking the users with valid location sources. This reinforces the important role that the online Twitter user geotagging procedure plays in our system.

[7]Multiple API tokens are used because one only follows up to 5000 users.

Table 2: Contributions of Different Local Live Tweet Sources

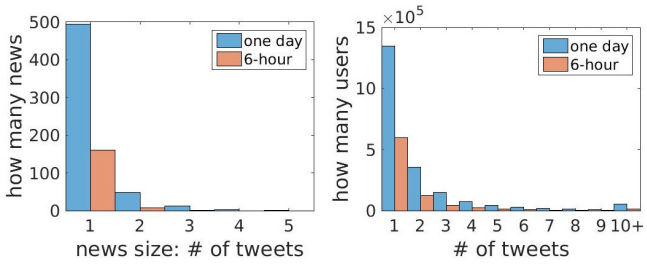
Source	# of tweets		# of news	
	Local tw.	Cluster tw.	Involved	Excl. (acc. %)
Sample API	6,182	638	167	21 (35.5%)
Str. API w/ loc.	76,983	2,123	359	76 (52.9%)
Str. API w/ fol. I	2,986,291	23,120	2,489	1,241 (58.5%)
Str. API w/ fol. II	1,730,889	16,654	1,857	609 (52.7%)
Total	4,800,345	43,535	N/A	N/A

In addition, Table 2 also lists the number of "Involved" news (i.e., how many news a source's tweets have participated in forming) and the number of "Exclusive" news (i.e., how many news a source's tweets have exclusively formed, in other words, these news are formed by tweets only from this source), along with its accuracy of positive local news (the accuracy evaluation method is detailed in Section 4.4.4). The result shows that the majority of news events are generated using the tweets in Streaming API statuses/filter w/ "follow", indicating that by tracking local Twitter users, our method is able to find much more news than solely using the Twitter's publicly available tweet streams.

4.4 Local News Detection

In this section, we evaluate the performance of our system on detecting news from the enhanced local live tweet stream using *mutual recall* and *precision*. Mutual recalls are evaluated between our system and a set of local news media agencies, together with a few baseline approaches. As for precision, we recruited 3 volunteers to individually judge the detected news and collect the results using the strategy of majority votes.

4.4.1 Parameter Settings. Note that although some of the following parameter settings depend on the specific input city, they are simply statistics and easy to infer for other places. There are 3 constraints for a word to be locality-aware: R_{TF}, R_{DF} and R_{CF}. For R_{TF}, the main goal is to capture a local news's nature of being new and to reflect a person's word-usage anomaly, by requiring both TF and TF' to be small (of course, at least greater than 0). This means that, an upper boundary needs to be imposed on TF. To obtain an empirical value of this, we collect the tweets posted by the Twitter accounts listed in Table 3 (note that the Twitter account of @fox25news has changed to @boston25 in April 2017), and perform an analysis, for each individual agency, of how many of its tweets are about the same news. The results, presented in Figure 6a, show that an agency usually tweets only 1 or 2 tweets (5 at most) about the same news. The situation is similar when the time period narrows down to a 6-hour (e.g., from 15:00 to 21:00). We therefore set the upper bound of TF to 5. Figure 6b reminds us that this value could work for most of Twitter users as they usually post less than 10 tweets, either in one day or in a 6-hour time window, This value, however, seems too strict for Twitter users who publish 10 or more tweets and perhaps keep posting updates on the same news event. We therefore turn to TF' to relax the constraint of TF, and set a threshold value of 0.3 for TF'. To summarize, we have $R_{TF} := (|T_u| < 10 \ \wedge \ TF \leq 5) \vee (|T_u| \geq 10 \ \wedge \ TF' \leq 0.3)$.



(a) Histogram of # of tweets in a news by each individual news agency. **(b) Histogram of # of tweets posted by each individual Twitter user.**

Figure 6: Histograms of # of tweets.

R_{TF} alone, however, is not enough because it would mark most of the words for most of Twitter users as locality-aware. We further utilize DF to explore another characteristic of local news: being "limited spread". Recalling the fact that one-fifth of the population are active Twitter users, we set $R_{DF} := 3 \leq DF \leq \frac{|U_S|}{20}$, where U_S are all the users in the time window S. Our argument is that when $DF = 3$, there might be an equal number of users reporting the same activity in Twitter. This further indicates that in reality, there might exist an ongoing news event that involves 15 people. Likewise, we set the upper boundary to 1% of the population, which is around $\frac{|U_S|}{20}$. The distribution of detected cluster size in Figure 8a further validates our assumption.

Finally, there is an additional constraint R_{CF} to get rid of commonly used words. Our analysis on the CF's of most common non-stopwords in English shows that they have a min CF of 0.57% (max: 2.7%, mean: 1.6% and median: 1.8%) . Also considering that the average number of tweets published by a Twitter user is around 2 (e.g., in Figure 6b, 2.30 and 1.82 for one day and 6-hour) and DF's upper bound, we set the upper bound of CF to $\frac{|U_S|}{10}$. Therefore, R_{CF} is set as $R_{CF} := CF \leq \frac{|U_S|}{10} \wedge CF' \leq 0.57\%$, which helps us to successfully recognize words like "trump", "martin", "luther", "day" and "people" as not locality-aware.

We then have 3 more threshold values to set for online clustering in Algorithm 3. For the least number of overlapping words between two tweets to cluster together, we set $m = 5$ because it is usually large enough to cover a news's "who", "what" and "where" information, e.g., the bold words in the example event of Figure 1. In our experience, a larger m makes clustering tightly cohesive yet might split the same news story into several clusters; while a smaller m might not fully reveal a story's own trait and groups non-related things together. To be consistent with R_{DF}, we set the least number of people in a cluster $n = 3$. At last, we set the local spread ratio threshold $r = 0.4$ to deal with the tweets we might miss.

4.4.2 Local News Media and Baseline Approaches. **Reputable Local News Media Agencies** We select 9 Boston local news agencies, as listed in Table 3 in the form of "@ScreenName", to collect their news tweets as a groud-truth dataset to compare with. As most of the tweets of these agencies are of good quality, we perform a simple single-linkage cluster algorithm to extract news from them. That is, for a single news agency, as long as any two of its tweets share $m = 5$ non-stopwords, we group them together, and throw away tweets with less non-stopwords.

Baseline Approaches We also compare our method with the following four baseline approaches.

- *TwitterStand*: TwitterStand [1] groups news tweets into cluster of tweets to form news stories using a TF-IDF based similarity.
- *TwitterStand-3*: By default, TwitterStand only reports a cluster as a news story if it has more than 10 tweets. In this setting, we relax the minimum number of tweets to 3, out of the consideration of fairness for TwitterStand to be able to detect news of small scale.
- *EvenTweet*: EvenTweet [22] first identifies temporal bursty keywords and spatial local keywords and then clusters them together according to their spatial density distribution. The spatial density distribution is calculated based on a 50×50 grid tessellation.
- *GeoBurst*: GeoBurst [11] first generates candidate events by identifying pivot tweets based on geographical and semantic similarities and then ranks the candidates according to their spatiotemporal burstiness to filter out noisy ones.

As summarized in Section 2, TwitterStand (or TwitterStand-3) is an event-anchored method and therefore is fed with the same enhanced local live tweet stream in Firefly to detect news, while the last two are location-anchored methods which only take geotagged tweets (33, 966 tweets with embedded GPS coordinates) as input.

4.4.3 Mutual Recalls. The mutual recalls are computed by examining how many news in the news agencies or baseline approaches have been found by our system and *vice versa*. We claim a news cluster $c_{\mathbf{X}}$ in agency $\mathbf{X}$ recalls a news cluster $c_{\mathbf{Y}}$ in agency $\mathbf{Y}$ if there is a tweet in $c_{\mathbf{X}}$ and another tweet in $c_{\mathbf{Y}}$ that share at least 5 non-stopwords. The results are summarized in Table 3, in which a news agency's "@Screen Name" is to represent its tweets news. Also, to make the table compact, we give each agency an order denotation in the column headers. Below the column headers are the number of news found in an agency or our system Firefly. So for a cell, it shows how many news row $\mathbf{X}$ covers over column $\mathbf{Y}$.

Table 3 shows that Firefly achieves high recalls against most of news agencies. For example, we successfully detect news like "Stabbing Reported at Stoughton Home of UMass Boston Chancellor", "Dog killed by coyote in Gloucester, police issue warning" and "A woman caught in the line of fire in Lyn" etc which are also reported by "@7News". In contrast, a very large portion of news in Firefly don't receive coverage from any of the listed news agencies, e.g., "There is a growing collection of lonely hand warmers at Fallon Field in #Roslindale", "Hockey star Kacey Bellamy took a break from prepping for the 2018 Winter Olympics to chat with @BrooksSchool girls hockey team today!" and "Just a portion of the many people that volunteered today to build STEM kits for Boston schools" etc.

In contrast, the default settings of TwitterStand have much lower recalls across the 9 local news agencies. Although relaxing its cluster size to have minimum of 3 tweets yields many more clusters, it doesn't yield clearly higher recalls. We conjecture that in doing so, TwitterStand-3 is reporting many small clusters for the same news due to the fragmentation problem in its online clustering [1]. For example, the 409 news of TwitterStand are covering 1, 607 news of TwitterStand-3. This also explains TwitterStand-3's extremely asymmetric mutual recalls over the local news agencies.

It comes as no surprise that EvenTweet and GeoBurst, both of which only run on sparsely available geotagged tweets, have low recalls across the local news agencies too. This is essentially because geotagged tweets cover very limited news in our dataset. For example, none of the news tweets posted by local news agencies contain geotagged tweets. Similarly, in all the tweets clusters generated by our system Firefly, only 633 of them contain geotagged tweets and only 107 of tweets clusters are formed by only geotagged tweets. This shows that by utilizing non-geotagged tweets, we are able to detect much more local news than methods EvenTweet and GeoBurst and further reinforces the importance of enhancing local live tweet stream by finding and tracking local Twitter users.

Table 3: The Mutual Recalls between Firefly, Baseline Approaches and the 9 reputable Boston Local News Agencies

	Order	A	B	C	D	E	F	G	H	I	J	K	L	M	N
# of news		3364	409	2331	184	179	61	21	128	95	52	64	11	15	110
Firefly	**A**	3364	305	1213	135	164	48	21	85	32	41	49	6	10	69
TwitterStand	**B**	200	409	1607	71	46	7	5	13	15	4	4	0	2	12
TwitterStand-3	**C**	215	395	2331	51	66	8	6	16	19	4	5	0	5	15
EvenTweet	**D**	236	218	292	184	151	6	6	3	3	15	2	0	3	15
GeoBurst	**E**	132	64	202	126	179	7	1	3	3	7	5	0	3	5
@7News	**F**	49	73	212	2	13	61	3	4	7	2	6	0	1	9
@BostonDotCom	**G**	21	22	47	1	1	3	21	6	5	0	2	1	0	3
@BostonGlobe	**H**	85	83	210	2	6	4	6	128	4	3	1	2	0	5
@bostonherald	**I**	38	82	179	1	3	7	5	4	95	0	7	1	1	5
@CBSboston	**J**	41	64	149	2	8	2	0	3	0	52	1	0	0	4
@fox25news	**K**	49	23	64	2	5	6	2	1	7	1	64	0	2	2
@GlobeMetro	**L**	6	0	0	0	0	0	1	2	1	0	0	11	0	0
@metroBOS	**M**	11	27	62	2	5	1	0	0	1	0	2	0	15	0
@WCVB	**N**	69	95	217	7	13	9	3	5	5	4	2	0	0	110

4.4.4 Precision. We asked 3 human judges to independently examine the 3,364 clusters of tweets detected in Firefly. As shown in Figure 7, each candidate news is a set of tweets with their urls. The set of tweets are selected by having the most non-stopwords, retweet numbers and overlapping words with each other and no more than 5 tweets. The drop-down list provides 3 available options: "Positive", "Neutral" and "Negative", which are used by the judges to answer the question: "Are the three or more tweets describing the same local news?". The instructions given to judges are summarized as follows:

Each candidate news has a set of tweets, followed by their urls. Please read the tweets and answer if they are talking about the same news. A local news, here, refers to an event that happens in Boston Metropolitan area. For example, local news can be about traffic, weather, missing persons/pets, farmer's market, yard-selling and book-selling, happy hour of bars and restaurants, crimes, protests, gatherings, award-nominations, and parties, meetings, celebrations, conferences, sports games etc. You can utilize the tweets' urls to get more information such as where the news happened. If you can't determine where it happened, choose "Negative". National/international news are recognized as "Neutral". News that happened in another place, like sports held in another city, should be "Negative". Also if you don't think the presented tweets are representing a news, select "Negative".

Figure 8b presents the distribution of judges' answers of the 2,574 events out of 3,364 that received a majority of "Positive"s or "Negative"s. Among the 2,574 events, 73.6% had 2 or more "Positive"s and were consented to be local news. The median number of tweets and median number of users in these local news are only 7 and 6, respectively, as shown in Figure 8a. We also discovered that most of the clusters with a majority of "Negative" were formed by a set of people tweeting like "My fitbit for 1152017 6145 steps and 29 miles traveled". This surprised us because this crowd behavior meets our constraint for local news. In addition, out of 3,3364 news we detect, 649 received 2 or more "Neutral"s and were considered to be national or international news.

Table 4: Proportions of Different Types of Tweet Clusters

	≥2 Positives	≥2 Neutrals	≥2 Negatives
Firefly	1,894 (56.3%)	649 (19.3%)	680 (20.2%)
TwitterStand	14 (3.4%)	306 (74.8%)	76 (18.6%)
TwitterStand-3	123 (5.28%)	1302 (55.9%)	816 (35.0%)
EvenTweet	44 (23.9%)	27 (14.7%)	90 (48.9%)
GeoBurst	52 (29.1%)	21 (11.7%)	70 (39.1%)

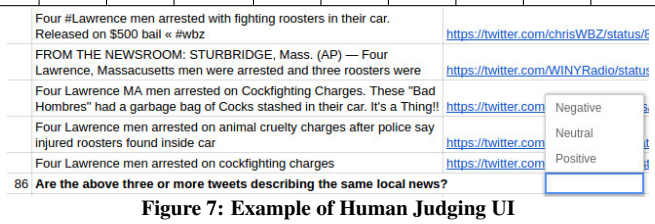
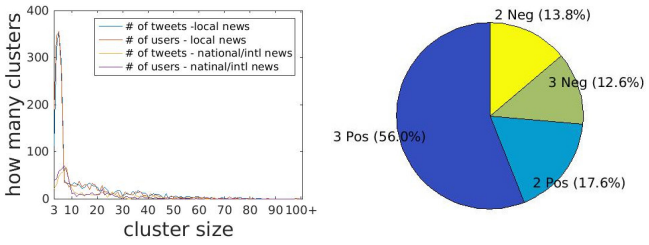


Figure 7: Example of Human Judging UI



(a) Distribution of tweets and users. **(b) Distribution of human evaluation.**

Figure 8: Distribution of news cluster sizes and human evaluation

Next, we evaluate the clusters in TwitterStand, TwitterStand-3, EvenTweet and GeoBurst in the same way and list their proportions of news receiving more than 2 "Positives", 2 "Neutrals" and 2 "Negatives" respectively in Table 4. In comparison, among the 409 clusters in TwitterStand, only 14 are identified as local news. The low proportion of local news in the default settings of TwitterStand is caused by its constraint that at least 10 tweets to form a cluster. Although relaxing this limit to 3 tweets in TwitterStand-3 captures more local news, its non-news proportion increases much more by falsely recognizing some repeating tweets from Twitter users as news, e.g. "@healylike". In contrast, by only exploiting the sparsely available geotagged tweets, EvenTweet and GeoBurst are only able to detect a small number of positive local news. Similarly, in Firefly, out of the 107 clusters that are formed by only geotagged tweets, 47 of them receive ≥ 2 "Positives" and are considered positive local news. This further illustrates that making only use of geotagged tweets will miss the majority of local news reported in non-geotagged tweets.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a system called Firefly to detect news for a given geographical area. In order to deal with the infamous sparsity problem in publicly available Twitter data, Firefly first enhances the local live tweet stream by identifying a large body of Twitter users in an area to follow via an online geotagging procedure and thereby

significantly increases the amount of tweets generated from that area. With the enhanced local live tweet stream, we propose a method to identify locality-aware keywords and further use them to cluster tweets together to detect news. Comparing with news extracted from a set of local news agencies' tweets, our system achieves the highest recalls, and at the same time, outperforms the baseline approach TwitterStand regarding both recall and precision in detecting local news, and more importantly, is able to detect much more local news than the approaches that only use geotagged tweets.

A small portion of news might be present in two or more clusters if these news don't get updates in a time period that exceeds 6-hour, which is the main reason whey Table 3 is not symmetric for Firefly. A remedy to this problem in the future might be to simply lengthen the time window or to keep a pool of news clusters before the current sliding time window and keep them active if they receive updating tweets. In addition, the importance of various users should be addressed differentially. For example, reporter or news agencies should be more trustworthy to publish news. Additionally, as the human verification yields a ground-truth of local news, a learning procedure might be explored to help determine the parameter values in extracting locality-aware keywords and online clustering. We leave these questions for our future work.

6 ACKNOWLEDGEMENT

This work was also supported in part by the NSF under Grants IIS-13-20791 and IIS-1816889.

REFERENCES

[1] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling. TwitterStand: News in Tweets. SIGSPATIAL '09.
[2] N. Gramsky and H. Samet. Seeder Finder: Identifying Additional Needles in the Twitter Haystack. LBSN '13.
[3] A. Jackoway, H. Samet, and J. Sankaranarayanan. Identification of Live News Events Using Twitter. LBSN '11.
[4] H. Samet, J. Sankaranarayanan, M. D. Lieberman, M. D. Adelfio, B. C. Fruin, J. M. Lotkowski, D. Panozzo, J. Sperling, and B. E. Teitler. Reading News with Maps by Exploiting Spatial Synonyms. *Commun. ACM*, 2014.
[5] M. D. Lieberman and H. Samet. Supporting Rapid Processing and Interactive Map-based Exploration of Streaming News. SIGSPATIAL '12.
[6] R. Compton, D. Jurgens, and D. Allen. Geotagging one hundred million Twitter accounts with total variation minimization. BigData '14.
[7] E. Kwan, P.-L. Hsu, J.-H. Liang, and Y.-S. Chen. Event Identification for Social Streams Using Keyword-based Evolving Graph Sequences. ASONAM '13.
[8] M. Mathioudakis and N. Koudas. TwitterMonitor: Trend Detection over the Twitter Stream. SIGMOD '10.
[9] H. Wei, J. Sankaranarayanan, and H. Samet. Finding and Tracking Local Twitter Users for News Detection. SIGSPATIAL '17.
[10] J. Krumm and E. Horvitz. Eyewitness: Identifying Local Events via Space-time Signals in Twitter Feeds. SIGSPATIAL '15.
[11] C. Zhang, G. Zhou, Q. Yuan, H. Zhuang, Y. Zheng, L. Kaplan, S. Wang, and J. Han. GeoBurst: Real-Time Local Event Detection in Geo-Tagged Tweet Streams. SIGIR '16.
[12] F. Atefeh and W. Khreich. A Survey of Techniques for Event Detection in Twitter. *Comput. Intell.*, 31(1):132–164, February 2015.
[13] H. Abdelhaq. *Localized Events in Social Media Streams: Detection, Tracking, and Recommendation*. PhD thesis, Heidelberg University, Heidelberg, Germany, November 2015.
[14] Q. Li, A. Nourbakhsh, S. Shah, and X. Liu. Real-Time Novel Event Detection from Social Media. ICDE '17.
[15] C. Zhang, L. Liu, D. Lei, Q. Yuan, H. Zhuang, T. Hanratty, and J. Han. TrioVecEvent: Embedding-Based Online Local Event Detection in Geo-Tagged Tweet Streams. KDD '17.
[16] M. Walther and M. Kaisser. Geo-spatial Event Detection in the Twitter Stream. ECIR '13.
[17] A. Boettcher and D. Lee. EventRadar: A Real-Time Local Event Detection Scheme Using Twitter Stream. GreenCom '12.
[18] L. Hong, A. Ahmed, S. Gurumurthy, A. Smola, and K. Tsioutsiouliklis. Discovering Geographical Topics in the Twitter Stream. WWW '12.
[19] X. Zhou and L. Chen. Event Detection over Twitter Social Media Streams. *VLDB*, 23(3):381–400, June 2014.
[20] W. Wei, K. Joseph, W. Lo, and K. Carley. A Bayesian Graphical Model to Discover Latent Events from Twitter. ICWSM '15.
[21] A. Skovsgaard, D. Sidlauskas, and C. S. Jensen. Scalable top-k spatio-temporal term querying. ICDE '14.
[22] H. Abdelhaq, C. Sengstock, and M. Gertz. EvenTweet: Online Localized Event Detection from Twitter. volume 6 of *PVLDB '13*.
[23] K. Y. Kamath, J. Caverlee, K. Lee, and Z. Cheng. Spatio-temporal Dynamics of Online Memes: A Study of Geo-tagged Tweets. WWW '13.
[24] K. Watanabe, M. Ochi, M. Okabe, and R. Onai. Jasmine: A Real-time Local-event Detection System Based on Geolocation Information Propagated to Microblogs. CIKM '11.
[25] C. Jonathan, A. Magdy, M. F. Mokbel, and A. Jonathan. GARNET: A holistic system approach for trending queries in microblogs. ICDE '16.
[26] W. Kang, A. K. H. Tung, F. Zhao, and X. Li. Interactive hierarchical tag clouds for summarizing spatiotemporal social contents. ICDE '14.
[27] A. Magdy, M. F. Mokbel, S. Elnikety, S. Nath, and Y. He. Mercury: A Memory-Constrained Spatio-temporal Real-time Search on Microblogs. ICDE '14.
[28] A. Magdy, A. M. Aly, M. F. Mokbel, S. Elnikety, Y. He, S. Nath, and W. G. Aref. GeoTrend: Spatial Trending Queries on Real-time Microblogs. SIGSPATIAL '16.
[29] J.-M. Xu, A. Bhargava, R. Nowak, and X. Zhu. Socioscope: Spatio-temporal Signal Recovery from Social Media. ECML PKDD '12.
[30] T. Lappas, M. R. Vieira, D. Gunopulos, and V. J. Tsotras. On the Spatiotemporal Burstiness of Terms. volume 5 of *PVLDB '12*. VLDB Endowment.
[31] Q. He, K. Chang, and E.-P. Lim. Analyzing Feature Trajectories for Event Detection. SIGIR '07.
[32] C. Budak, T. Georgiou, D. Agrawal, and A. El Abbadi. GeoScope: Online Detection of Geo-correlated Information Trends in Social Networks. PVLDB '13.
[33] Z. Liu, Y. Huang, and J. R. Trampier. LEDS: Local Event Discovery and Summarization from Tweets. SIGSPATIAL '16.
[34] G. Valkanas and D. Gunopulos. How the Live Web Feels About Events. CIKM '13.
[35] S. Roller, M. Speriosu, S. Rallapalli, B. Wing, and J. Baldridge. Supervised Text-based Geolocation Using Language Models on an Adaptive Grid. EMNLP-CoNLL '12.
[36] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller. Twitinfo: Aggregating and Visualizing Microblogs for Event Exploration. CHI '11.
[37] M. Quezada, V. Peña Araya, and B. Poblete. Location-Aware Model for News Events in Social Media. SIGIR '15.
[38] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang. TEDAS: A Twitter-based Event Detection and Analysis System. ICDE '12.
[39] Y. Yamaguchi, T. Amagasa, and H. Kitagawa. Landmark-based User Location Inference in Social Media. COSN '13.
[40] C. A. Davis Jr., G. L. Pappa, D. R. R. de Oliveira, and F. de L. Arcanjo. Inferring the Location of Twitter Messages Based on User Relationships. *Trans. GIS*, 15 (6):735–751, 2011.
[41] A. Sadilek, H. Kautz, and J. P. Bigham. Finding Your Friends and Following Them to Where You Are. WSDM '12.
[42] Y. Chen, J. Zhao, X. Hu, X. Zhang, Z. Li, and T.-S. Chua. From Interest to Function: Location Estimation in Social Media. AAAI '13.
[43] Z. Cheng, J. Caverlee, and K. Lee. You Are Where You Tweet: A Content-based Approach to Geo-locating Twitter Users. CIKM '10.
[44] J. Mahmud, J. Nichols, and C. Drews. Home Location Identification of Twitter Users. *ACM TIST*, 5(3):47:1–47:21, July 2014.
[45] B. Han, P. Cook, and T. Baldwin. Text-based Twitter User Geolocation Prediction. *J. AIR*, 49(1):451–500, January 2014.
[46] N. Dalvi, R. Kumar, and B. Pang. Object Matching in Tweets with Spatial Models. WSDM '12.
[47] G. Li, J. Hu, J. Feng, and K. l. Tan. Effective location identification from microblogs. ICDE '14.
[48] M. D. Lieberman and H. Samet. Multifaceted Toponym Recognition for Streaming News. SIGIR '11.
[49] M. D. Lieberman and H. Samet. Adaptive Context Features for Toponym Resolution in Streaming News. SIGIR '12.
[50] M. D. Lieberman, H. Samet, and J. Sankaranayananan. Geotagging: Using Proximity, Sibling, and Prominence Clues to Understand Comma Groups. GIR '10.
[51] H. Samet. Using Minimaps to Enable Toponym Resolution with an Effective 100% Rate of Recall. GIR '14.
[52] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors. WWW '10.
[53] J. Weng and B.-S. Lee. Event Detection in Twitter. ICWSM '11.
[54] M.-D. Albakour, C. Macdonald, and I. Ounis. Identifying Local Events by Using Microblogs As Social Sensors. OAIR '13.
[55] Y. Takhteyev, A. Gruzd, and B. Wellman. Geography of Twitter networks. *Social Networks*, 34(1):73 – 81, 2012.
[56] D. Mok, B. Wellman, and J. Carrasco. Does Distance Matter in the Age of the Internet? *Urban Studies*, 47(13):2747–2783, 2010.
[57] Y. Vardi and C.-H. Zhang. The multivariate L1-median and associated data depth. *Proc. NAS*, 97(4):1423–1426, 2000.
[58] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenker, and I. Stoica. Resilient Distributed Datasets: A Fault-tolerant Abstraction for In-memory Cluster Computing. NSDI '12.
[59] A. Dave. IndexedRDD: Efficient Fine-Grained Updates for RDDs. *Spark Summit '15*.
[60] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster Computing with Working Sets. HotCloud '10.
[61] M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica. Discretized Streams: Fault-tolerant Streaming Computation at Scale. SOSP '13.