2012 IEEE 28th International Conference on Data Engineering

TEDAS: a Twitter-based Event Detection and Analysis System

Rui Li #1, Kin Hou Lei *2, Ravi Khadiwala #3, Kevin Chen-Chuan Chang #4

#Computer Science, University of Illinois at Urbana-Champaign
1ruili1@illinois.edu
3khadiwal@illinois.edu
4kcchang@cs.uiuc.edu

*Computer Science Department, Brigham Young University
2tobias.garrick@byu.edu

***Abstract*— Witnessing the emergence of Twitter, we propose a Twitter-based Event Detection and Analysis System (TEDAS), which helps to (1) detect new events, to (2) analyze the spatial and temporal pattern of an event, and to (3) identify importance of events. In this demonstration, we show the overall system architecture, explain in detail the implementation of the components that crawl, classify, and rank tweets and extract locations from tweets, and present some interesting results of our system.**

I. Motivation

Recently, Twitter, a popular microblogging service, has become a new information channel for users to receive and to exchange information. Everyday, nearly 170 million tweets are created and redistributed by millions of active users.

Twitter has several unique advantages that distinguish it from news web sites, blogs, or other information channels. First, tweets are created in real-time. With the brevity guaranteed by a 140-character-message limit and the popularity of Twitter's mobile applications, users tweet and retweet instantly. For example, we could detect a tweet related to a shooting crime 10 minutes after shots fired, while the first news report appeared approximately 3 hours later. Second, tweets have a broad coverage over events. On Twitter, millions of general users, as well as verified accounts such as news agents, organizations and public figures, are constantly publishing new tweets. Every user can report news that is happening around him or her. Thus, tweets cover nearly every aspect of daily life, from national breaking news (e.g., earthquakes), local events (e.g., car accidents), to personal feelings. Third, tweets are not isolated; they are associated with rich information. For example, for each tweet, we can find an explicit time stamp, the name of the user, the social network the user belongs to, or even the GPS coordinates if the tweet is created with a GPS-enabled mobile device.

With these features, Twitter is, in nature, a good resource for *detecting* and *analyzing* events, which are the main concepts we will demonstrate.

Detecting Events: By taking advantage of both the speed and coverage of Twitter, we can detect events in a timely manner by listening to incoming tweets. As a tweet is often associated with spatial and temporal information, we can detect when and where an event happens. E.g., via monitoring an incoming tweet "Shooting outside the Irving mall." at 2:38pm on July 24, we were able to detect the crime immediately, and extract the location and the time of the crime as well.

Analyzing Events: People from different places may tweet about the same event (e.g, "Norway shooting") or the same type of event (e.g, "tornado"); by collecting tweets over time we can analyze two aspects from aggregate event information. First, we can discover the spatial pattern and temporal pattern of events. E.g., via analyzing the distribution of tweets related tornados, we can discover the major tornado regions. Second, we can identify the important events given a region and a time period. E.g., via analyzing the number of tweets related to Norway shooting, we can identify it as an influential event worldwide.

Recently, several systems (e.g., [1]) have been proposed to detect events from tweets, but most of them are missing the analysis component. In the literature, several systems (e.g., [2], [3]) are proposed to analyze events from blogs, but they may fail in processing tweets, which are short and noisy, and do not explore rich information (e.g., users's network) in Twitter.

II. Twitter based Events Detection and Analysis

In this paper, we propose a novel system that can detect and analyze events by exploring rich information from Twitter. It fully supports the three functions proposed above: (1) detecting new events, (2) ranking events according to their importance, and (3) generating temporal and spatial patterns for events. As an initial step, we focus on Crime and Disaster related Events (CDE), such as shooting, car accidents, or tornado, as they are important types of events.

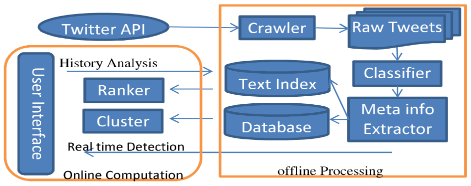


Fig. 1. System Architecture

Fig 1 shows the architecture of TEDAS. It contains two major parts: (1) offline processing and (2) online computing.

Offline processing retrieves, processes and stores CDE tweets. Using Twitter APIs, our CDE-focused crawler crawls potentially CDE-related tweets from Twitter. As crawled tweets may or may not be related to CDEs, they are fed into a classifier which determines whether a tweet is related to a CDE. A CDE-classified tweet is sent through a meta information extractor, which extracts temporal and spatial information from the tweet. The extracted meta information along with the original tweet is indexed by a text search engine and stored in a database system. The index and the database are used to support retrieving real time CDEs, as well as online analytical queries.

Online processing supports CDEs detection and answers users's analytical queries and generates visual results. The system consistently queries the index and the database to retrieve new arrived CDEs and display them. Given a query, which contains a spatial range (e.g, "Houston"), a temporal period (e.g., "July 2010"), keywords (e.g., "car accidents"), or their combinations, the system first fetches matched tweets and their meta information with the support of the text index and the database. As the system aims to identify important CDEs for the query, those tweets are ranked by a ranking model according to their importance. To extract spatial and temporal patterns for the query, a clustering model groups similar tweets into different geographic regions or temporal ranges. The results are sent to the interface, where a visualization of the results is provided.

Building such a system is non-trivial. Due to the limitation of Twitter API and the length of tweet, effectively implementing components, including crawling, classification, ranking, and extraction, requires new algorithms that fully explore the Twitter network. We will explain the algorithm for each component and present preliminary results in Sec IV.

III. Demonstration Scenarios

We implemented TEDAS based on Java and PHP with support of MySQL, Lucene, Twitter API, and Google Maps API. At the time of writing, the system has indexed over a million CDE tweets over two months and about a million users' information, and continues to index at rate of about 30,000 new CDE tweets per day. It supports detecting events related to crimes, accidents, and disasters.

To begin with, we demonstrate our system by presenting the three major functions. (1) As shown in Fig 2(a), the system is used to detect new CDEs (e.g., thunderstorm watch) from real-time incoming tweets, extract a GEO location for events (e.g., Kewaunee, WI), and plot them on a map. (2) As shown in Fig 2(b), users can input a location (e.g., Michigan) and a time period (July 2011), and the system will return CDE tweets (e.g., Grand Rapids shooting) at the given location during the time period, which are ranked according to their importance. (3) As shown in Fig 2(c), users can input a keyword (e.g., tornado) with an optional geographic range and time period as a query, and the system will find the spatial and temporal patterns for the query. The temporal pattern indicates that tornados happened on July 21, 2011, while the spatial pattern tells major tornado regions, such as Kentucky and Missouri.

In addition, we allow users to conduct a complete analysis through the interaction with these functions in the system. For example, if a user wants to analyze traffic accidents in July, he can start with inputting a query containing keywords "car accident" and a time range (e.g, July 1 to July 24). Our system will return results shown in Fig 3(a) as the spatial and temporal patterns for car accidents. Fig 3(a) clearly identifies that nearly all high car-accidents rate regions (shown in red) are major cities such as Houston, Los Angeles and New York. If a user clicks on the tweets button, a list of related tweets are listed as shown in Fig 3(b). We find a near-fatal car accident related to the husband of Khloe, a popular star in US, is ranked as top. The user can click on the tweet to see the detail. On the map, one can zoom in to Houston, as there are many accidents indicated by Fig 3(a). The system will update the map and show the spatial pattern about car accidents in Houston as Fig 3(c). Interestingly, we find that major car accidents areas in Houston are all intersections of highways.

IV. Challenges and Approaches

In this section, we discuss our implementation detail for several major challenging components in TEDAS.

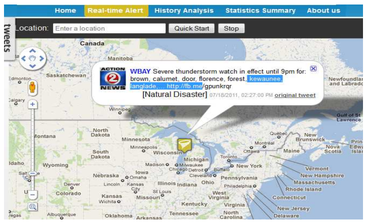
A. Crawling Tweets

The goal of our crawler is to crawl all CDE-related tweets with available Twitter's APIs. Twitter only provides two types of APIs for us to obtain public tweets. One returns a sample of all public tweets, and the other returns all the tweets that contain certain words (e.g., tornado) or those from some certain users (e.g., ABCNews).

We chose the latter API to design our crawler due to the following reasons. First, we can not obtain all CDE tweets with the first type of API. As mentioned previously, the second type of API returns all the tweets that satisfy given conditions, while the first type of API only returns a sample of tweets. Second, it is possible to retrieve most of CDE tweets with a set of well-defined keywords. We observe that similar types of CDEs share similar keywords. E.g., for shooting crimes, keywords such as death or police are common. For disasters, keywords such as alert or warning are common. As result, we can track a set of keywords, referred as tracking rules, to retrieve potential CDE tweets.

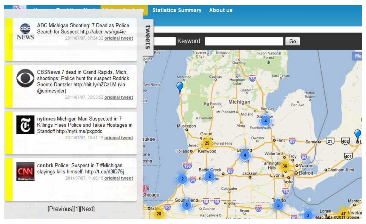
Ideally, a good rule will retrieve more CDE tweets and less non-CDE tweets. A good set of rules will cover most CDE tweets and less non-CDE tweets. Nevertheless, it is difficult to manually define a good set of rules. Instead, we adopted the bootstrapping idea to expand our tracking rule set automatically and iteratively. Our assumption is that as different rules (e.g., police or killed) are mixedly used in tweets (e.g, Police say at least 84 people were killed.), new rules (e.g., police) can be discovered based on existing rules (e.g., killed). Therefore, we can analyze the retrieved tweets to discover new rules.

Fig 4 illustrates our crawling strategy. We first manually select a set of rules (e.g., thunderstorm) as seeds, and use



WBAY Severe thunderstorm watch in effect until 9pm for: brown, calumet, door, florence, forest, kewaunee, langlade,... http://fb.me/gpunkrqr
[Natural Disaster] 07/18/2011, 02:27:00 PM original tweet

(a) Events Detection



ABC Michigan Shooting: 7 Dead as Police Search for Suspect http://abcn.ws/rgu4ie
2011/07/07, 07:34:32 original tweet

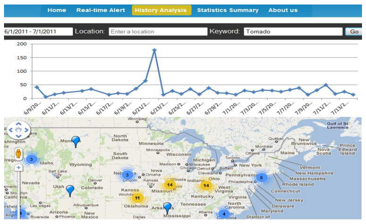
CBSNews 7 dead in Grand Rapids, Mich. shootings; Police hunt for suspect Rodrick Shonte Dantzler http://bit.ly/nZCzLM (via @crimesider)
2011/07/07, 05:53:52 original tweet

nytimes Michigan Man Suspected in 7 Killings Flees Police and Takes Hostages in Standoff http://nyti.ms/pxgzdc
2011/07/07, 10:41:15 original tweet

cnnbrk Police: Suspect in 7 #Michigan slayings kills himself. http://t.co/d3lD76j
2011/07/07, 11:06:15 original tweet

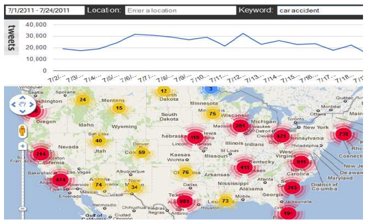
[Previous][1][Next]

(b) Important CDEs

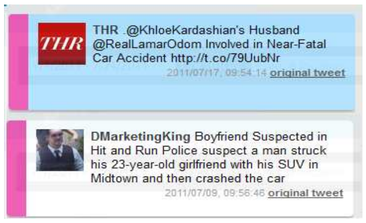


(c) Spatial and Temporal Patterns

Fig. 2. Three Major Functions



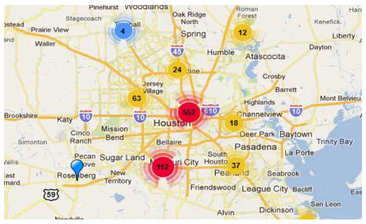
(a) Patterns for Car Accidents



THR .@KhloeKardashian's Husband @RealLamarOdom Involved in Near-Fatal Car Accident http://t.co/79UubNr
2011/07/17, 09:54:14 original tweet

DMarketingKing Boyfriend Suspected in Hit and Run Police suspect a man struck his 23-year-old girlfriend with his SUV in Midtown and then crashed the car
2011/07/09, 09:58:46 original tweet

(b) Important Car Accidents



(c) Car Accidents in Houston

Fig. 3. Complex Analysis Scenario

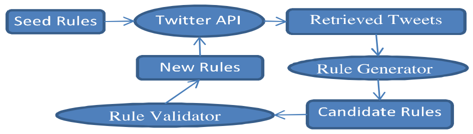


Fig. 4. Crawling Strategy

the streaming API to retrieve the matched tweets (e.g., M_1: "Severe Thunderstorm Warning is in effect for southwestern Bent until 4:15 PM.", M_2: "Incredible! Epic thunderstorm captured by photographer", and M_3: "Severe Thunderstorm Warnings Issued In Mid-State").

To extract new rules from retrieved tweets, a rule extractor is used to analyze retrieved tweets. The intuition behind the extractor is that a good rule (e.g., "Severe", "Warning") generally associates with CDE tweets (e.g, M_1, M_2) and rarely associates with non-CDE tweets (e.g., M_2). The extractor first extracts every unigram or bi-gram from tweets as candidate rules. For each candidate rule r, we define r's confidence, $Conf(r)$, as the ratio of between CDE tweets and non-CDE tweets that match r. Let $C(r)$ and $N(r)$ be CDE tweets and non-CDE tweets associated with r respectively. $Conf(r)$ is calculated as $\frac{|C(r)|+\alpha}{|N(r)|+\beta}$, where α and β are priors to punish rare terms. A rule r is selected if it passes a threshold.

Then, a rule validator is used to examine the usefulness of a new rule, as a new rule extracted from existing tweets may not bring new CDE tweets or it may even bring a lot of unexpected non-CDE tweets. Intuitively, we can evaluate the usefulness of a rule r using the ratio of new retrieved CDE tweets to new retrieved non-CDE tweets with some test runs. Specifically, let $C'(r)$, $N'(r)$ be the CDE and non-CDE related tweets retrieved by r in the test runs respectively, and $C'(S)$, $N'(S)$ be CDE and non-CDE related tweets retrieved by an existing set of rules S. The usefulness of a rule r is calculated as $\frac{|C'(r)|-|C'(r)\cap C'(S)|+\alpha'}{|N'(r)|-|N'(r)\cap N'(S)|+\beta'}$, where α' and β' are values to punish rare cases.

Finally, a set of new rules (e.g.,"warning") are added into the set. New tweets (e.g., "Weather Alert:Excessive heat warning through tomorrow 10pm.") will be collected and new rules (e.g., "excessive heat") will be generated iteratively. The procedure ends when no rule can be added into the set.

With the approach above, we crawl about $100,000$ tweets per day, among which 30% are CDE related tweets. For comparison, we evaluate a random sampling approach, the result of which contains only 0.05% CDE tweets in the crawled samples. With this information, we estimate we have crawled roughly 85% of all CDE tweets.

B. *Classifying Tweets with Social Features*

As not all crawled tweets are about CDEs, the classifier needs to determine whether a crawled tweet is related to CDEs or not. Although it is trivial to build a classification model based on a tweet's text, the model's performance is unacceptable, as tweets are limited to 140 characters and contain a lot of noisy information (e.g, misspelled words or short URLs). Thus, we explore additional features from Twitter in two directions.

Twitter-specific features: In Twitter, a tweet (e.g., @dcfireems: List of #MD cooling centers: bit.ly/prBr8x) has particular format. It may contain a short URL, which links to a page (e.g., "bit.ly/prBr8x"), a hash tag (e.g., "#MD"), which indicates a topic, or an "@" sign (e.g., "@dcfireems"), which means replying to somebody. These features provide extra signals for determining whether a tweet is CDE or not. E.g., if a tweet contains a link that points to a news page, it is likely to be related to CDEs and the content of the page can be used as additional text. If a tweet contains an "@" sign, it is likely to relate to personal communication.

CDE-specific features: Given M_1, M_2 and M_3 mentioned in Sec IV-A, we observed that although their content is similar, CDE tweets (e.g., M_1, M_3) often contain time (e.g., 4:15pm) or location (e.g, southwest bent), while non-CDE tweets (e.g., M_2) do not. In addition, a number is frequently mentioned in a CDE tweet (e.g., 7 people are injured) as well. Thus, we use those CDE-specific features for classification.

We trained a classification model which is tested to have 80% accuracy. It greatly improves the model with only text features, whose accuracy is only 60%.

C. Predicting the Location of a Tweet

Our extraction model is to assign a geographic location for each tweet. We find three resources that can help us to associate a tweet with a location. The first on is the GPS tag in a tweet. It is accurate but sparse. Only a few tweets contain GPS tags. The second one is content of a tweet. We can extract a location (e.g., Chicago) or a place (e.g., Time Square) from a tweet. However, not all the tweets contain location names explicitly in their content. The third one is the location of the creator of a tweet. When the first two resources are not available, we use it. However, not all users' have locations in their profiles. Hence, we need predict the users' location and assign it to their tweets.

To accurately predict a user's location, we explored rich information from the Twitter network. To predict a user A's location, we can explore A's historical messages as some of them have GPS tags and some of them mention explicit locations. We can also explore A's social network, as some of them have a particular location on their profiles.

To connect a user's location with locations from his tweets and friends, we have the following three observations. First, a user's location is more likely to appear in his tweets than other locations. Second, a user's friends tend to be closer with the user. Third, a user's location is mentioned at least once in his tweets or is the same with at least one of his friends.

With these observations, we can predict a user's location as the location from his friends or tweets that minimizes the overall distances between locations in his tweets and from his friends. Specifically, Let L_u be the location of a user u, M_u denote a set of locations mentioned in u's tweets, F_u denote a set of locations of u's friends, and $D(L_i, L_j)$ be the distance between two locations L_i and L_j. Thus, u's location L_u is the solution of the following optimization problem.

$$\begin{aligned} L_u &= \arg\min \Sigma_{L_i \in M_u} D(L_x, L_i) + \Sigma_{L_j \in F_u} D(L_x, L_j) \\ &\quad\ subject\ to\ L_x \in M_u \cup F_u \end{aligned}$$

With this model, we can predict a user's location from his tweets and friends with 63% accuracy.

D. Ranking Tweets

Our ranking model aims to identify important CDEs. However, similarity-based ranking functions (e.g., BM25) are infeasible because similarity cannot indicate the importance. Pagerank-like algorithms are infeasible neither due to the unexplicit links between tweets. Algorithms based on users' connections (e.g., expertrank) can derive users' authority, but the authority is not the only measure of the importance.

We develop our ranking model in a learning-to-rank approach, which learns a function to assign an score to each tweet, as it can systematically integrate a variety of signals, such as author's credibility and the number of retweets. To predict a tweet's importance precisely, we explored signals from various aspects, including content, user and usage.

Content Features: There are two kinds of content features. (1) Features that are defined according to whether a tweet contains some important words such as killed or death. Those words indicate whether a tweet is an important CDE or not. (2) Features that are defined based on specific tweet formats. As discussed in IV-B, a tweet may contain a URL or a hash tag. A tweet that contains a link or a hash tag may have a broad coverage, which may be also important.

User Features: A tweet from an authority is probably more significant than the one from a user with less credibility. Several signals can indicate the credibility of a user such as whether a user is a verified account (e.g., a news agent or a police department), how many followers a user has, the age of the account, and the number of tweets that a user has.

Usage Features: This is the chief kind of features which have not been explored in the literature. Twitter is a communication network, where information spreads over the network. We assume that the wider a tweet spreads, the more significant the tweet is. We capture the spread of a tweet both explicitly and implicitly. Features useful to indicate the explicit spread of a tweet are its re-tweet number and the number of favorites. As some users tweet regrading to the same event without retweeting, we need to capture the implicit spread. In particular, we measure it in two different ways: (1) how many tweets are similar to the tweet within a time range and location range, (2) how many tweets contain the hash tags used in the tweet.

Finally, we use a linear regression model to estimate the importance of a tweet based on the above features.

V. Contributions

To summarize, we made the following contributions: (1) Conceptually, we propose to explore a new source – Twitter for detecting and analyzing CDEs. Three interesting functions are discussed. (2) Technically, we developed an efficient CDE-focused crawler, and explored valuable and novel features from Twitter to classify and rank tweets, and to predict locations from tweets. (3) At the system level, we implement a realtime system that can detect and analyze CDEs based on millions of tweets and users.

References

[1] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes twitter users: real-time event detection by social sensors," in *WWW '10*, pp. 851–860.

[2] N. Bansal and N. Koudas, "Blogscope: spatio-temporal analysis of the blogosphere," in *WWW '07*, pp. 1269–1270.

[3] Q. Mei, C. Liu, H. Su, and C. Zhai, "A probabilistic approach to spatiotemporal theme pattern mining on weblogs," in *WWW '06*, 2006, pp. 533–542.