RISECURE: Metro Incidents And Threat Detection Using Social Media

Omer Zulfiqar[1], Yi-Chun Chang[1], Po-Han Chen[1], Kaiqun Fu[1], Chang-Tien Lu[1], David Solnick[2] Yanlin Li[2],
[1](omer95, bensonchang, pohan, fukaiqun, ctlu)@vt.edu
[2](dsolnick, yli)@wmata.com
[1]Department of Computer Science, Virginia Tech, Arlington, VA, USA
[2]Washington Metropolitan Area Transit Authority, Washington DC, USA

***Abstract*—Open and accessible public utilities such as mass public transit systems are some of the vexing venues that are vulnerable to several criminal acts due to the large volumes of commuters. Existing forms of threat or event detection for the rail-based transit systems are either not working in real-time or do not provide complete coverage. In this paper, we present RISECURE[1], an open-source system, that uses real-time social media mining to aid in the early detection of such possible events within a rail-based/metro system. The system leverages dynamic query expansion to keep track of any new emerging information about any particular incident. The Real Time Incident panel of the proposed system provides a comprehensible representation of the evolution of threatening transit events, which are further shown in the storyline modal for each respective station. The alert notification module of the system is capable of monitoring threats to the rail-based/metro systems in real-time. We demonstrate the system by including case studies involving incidents occurring within the Washington DC Metropolitan Area Transit Authority (WMATA) metro system to justify the effectiveness of our approach.**

***Index Terms*—Data Mining, Dynamic Query Expansion, Web Application**

I. Introduction

Public Transport Networks play a vital role in the functioning of any major city or even country. The fact that they are "open" and "public" makes them virtually open to anyone, bringing in huge volumes of people. Rail-based transit systems are one of the most widely used forms of public transportation around the world. In January 2019, the Washington Metropolitan Area Transit Authority (WMATA) stated that the Metro's weekday average ridership was at 626,000 rides [1]. Assuming every person makes a round trip, that's approximately 300,000 people a day riding the Metro during the week. Any threat or security-related issue faced by the Metro jeopardizes the people using the service. This could be anything from a minor rider emergency-related delays to any criminal/terrorist activity. However, putting the lives of the commuters in jeopardy, the potential loss of lives and the potential damage to the critical infrastructure of the system are things to worry about [2]. Along with social disruptions, this also brings in serious economic repercussions.

[1]https://wmata.cs.vt.edu/

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 © 2020 IEEE

Most current systems are restricted to only showing the expected arrival time for trains and/or disruptions due to maintenance/construction. Rider safety is a big concern and the absence of such a component motivated us to take on this project for WMATA. **So how can we aid in the early detection of these events to account for the deficiencies in the current system?** "If you see something, say something," is an ad slogan by the New York City Metropolitan Transportation Agency (MTA). The city cannot dispatch its police force on every train, bus, and subway, so it ran the campaign to ask commuters to be the eyes and ears of public safety [3]. In today's era social media platforms also allows people to become "human sensors". Social media websites like Twitter, have become a powerful and economical tool for extracting information. It has a sufficiently large pool of "Tweeters", one that is more diverse than any other specific incident crowd-sourcing tool. Twitter data has 2 key properties that make it highly suitable for this problem. *Promptness:* Compared to traditional media sources tweets are often posted rapidly after an incident. [4]. This data can be captured in a timely manner and can further span the entire transit network. *Geolocated:* According to the latest statistics on Twitter usage, 80% of tweeters post from mobile devices [5].

Here we present RISECURE, a tool that leverages Social Media Data and uses tweets as surrogates to extract information relevant to any possible security events/incidents within a Metro system. It includes an alert notification system that pings the user or security professional of any instance of new or updated information regarding the event. This is not an approach to predict or determine the likelihood of an incident, it is just a resource to aid in earlier detection, gain situational awareness and early deployment of resources to contain the situation and bring it under control. We provide additional value by demonstrating case studies on historical events and events caught in real time by our system within the Washington D.C Metro system(WMATA).

The major contributions of RISECURE are:

- **Social Media Mining:** Acquisition of Twitter data to store data on the events and use it to extract candidate tweets using keywords detection, and using Dynamic Query Expansion to track any new and emerging chatter on the incident via Dynamic Query Expansion.

- **Real Time Storyline:** A sorted timeline that keeps track of the incident(s), allowing the users to stay updated from start to end. When the status of an incident changes or new tweets related to the incident are detected, users can see the update immediately.
- **Alert Notifications:** This will be an extension of the Data Mining processes. Once identified and detected, an event will be passed on to the user interface, and the user will be notified.
- **Web and Mobile Platform Generation:** A convenient provision of the Data Mining model providing users with an effective visualization of the location of the event along with any necessary information in the form of a timeline.

II. System Overview

In this section, we illustrate the system architecture of RISECURE, as pictured in Figure 1. In the next few subsections, we will be going over every component of the model in more detail.

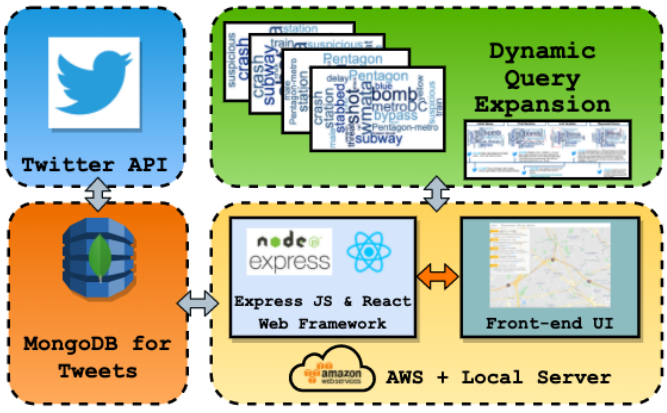

Fig. 1. System Architecture

Data Acquisition: This is broken down into three separate stages; Data Acquisition, Dynamic Query Expansion and Station Entity extraction. Users or "Tweeters" who tweet publicly generate a huge amount of data that is available for researchers to use. To retrieve historic data, we used the Twitter search API. To avoid the limitation of Twitter's API, the tool GetOldTweets3 was used to collect the data samples. The initial data was acquired using a set of seed keywords to query and filter tweets related only to the Metro stations or the metro itself. Table I below shows our initial set of Metro related seed keywords.

TABLE I
SEED KEYWORDS USED FOR INITIAL QUERY

Category	Seed Keywords
Metro	*metro, subway, train, station, wmata, rail, metroDC*
Threat / Incident	*gun, shot, suspicious, threat, struck, police, crash, weapon, attack, bomb*

After data acquisition, the data was cleaned up and tokenized. The data preprocessing was done using the Natural Language Toolkit (NLTK) library. We used this pre-processed data to analyze and search for the keywords related to any threats or incidents to further filter the dataset.

Application Server: This is the core server component of the application. We use AWS cloud and MongoDB to help integrate the data acquisition module and the backend database. After the data has been acquired, the AWS Lambda function will fire and send an API request to our backend service to update the data in our database. For the backend, we use Express.js with node.js as a web server framework following the RESTful API principles.

Application and Mobile Interface: This is the major component of user interactions and operations. The web application was built based on the React.js framework and Google Map API. Besides, we use Progressive web application(PWA) to construct our mobile app. PWA can be installed on the user's device much like native apps and provide cross-platform compatibility for iOS and Android.

III. Features

The RISECURE platform crawls and extracts any tweets related to the Metro system using the Twitter API. The generalized dynamic crawling process developed and utilized can be applied to multiple cities. The location is extracted using named entity recognition, which is then passed to the Google Geocoder, which helps us pinpoint the incident location on the map.

A. *Dynamic Query Expansion*

Dynamic Query Expansion evaluates inputs and reformulates the query result to improve retrieval performance. After acquiring the candidate tweets, we can use data to extract the representative keywords for a specific threat event. This helps keep track of the emerging information for the event as it progresses. Besides, we select some high-frequency keywords, as shown in Table I, as our initial seed query S based on analyzing the historical data of threat-related tweets.

Algorithm 1: Dynamic Query Expansion Algorithm

Input: A time-ordered sequence of Tweets $\langle T_0, T_1, \ldots, T_t \rangle$, Seed Query S
Output: Expanded Query Q
Set $Q_0 = S = F_0, w(F_0) = 1, k = 0$
repeat
  $k = k + 1$;
  $w(F_k) = idf(F_k) \cdot C \cdot w(T_{(k-1)})$; // use $w(T_{(k-1)})$ weight to compute $w(F_k)$
  $w(T_k) = \Phi \cdot C' \cdot w(F_k)$);
  repeat
    $swap(min(w(T_k)), MAX(w(T - T_k)))$;
    $\sigma = min(w(T_k)) - MAX(w(T - T_k))$;
  until $\sigma \leq 0$;
until $w(F_k) = w(F_{(k-1)})$;
$Q = F_k$;

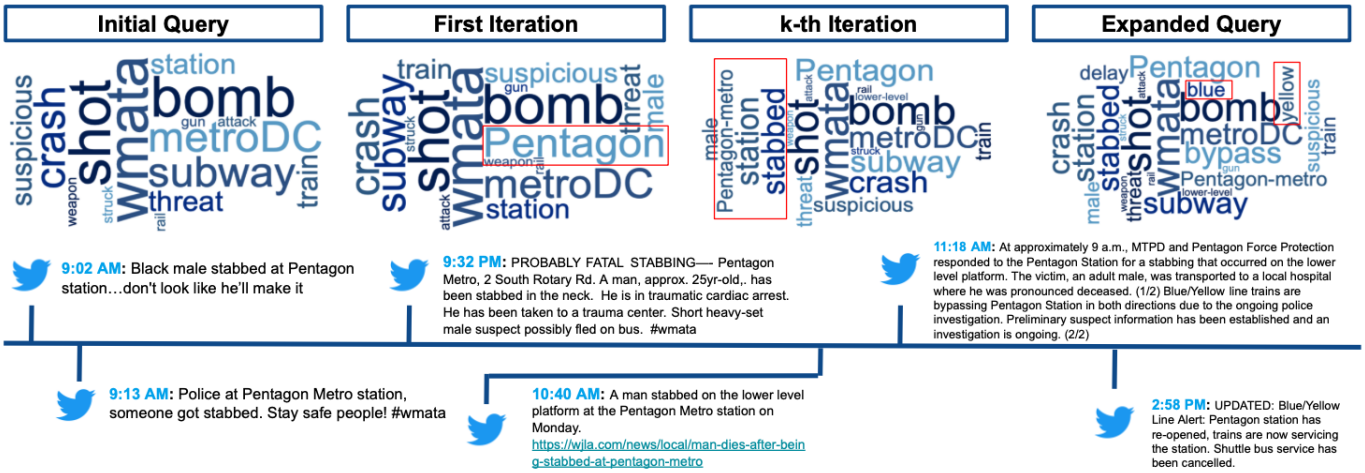

Fig. 2. Dynamic Query Expansion for Pentagon Metro Stabbing Case Study

To select the representative keywords, we use the algorithm based on dynamic query expansion (DQE) techniques [6], [7]. Given a time-ordered sequence of Tweets $\langle T_0, T_1, \ldots, T_t \rangle$ and Seed Query S, we could retrieve the new expanded query Q to represent this event. F_k is the feature node. W is the set of weights for nodes where higher weights denote a higher degree of relation between the node (either a tweet or a feature) and threat-related theme. We can calculate the weight of F_k by Inverse Document Frequency(IDF) and weight of $T_{(k-1)}$. C is the adjacency matrix.

For each iteration, dynamic query expansion compares the minimum weight of the related tweet node and the maximum weight of unrelated tweet node, selecting the one with a higher score and putting it in the result. After the kth iteration, it converges to the stable representative keywords. After the stable status is reached, we can assume that the highest weighted keywords could describe the event. We retrieve this result and represent it on our application. The dynamic query expansion for the Pentagon Metro Stabbing Case study is shown in Figure 2. After more event-related tweets are collected, we can see how keywords transform from an initial query with equal weight to the expanded query with more representative keywords.

B. Geo-Tagging

Not all tweets contain geo-location information. To extract the location from the tweets we have currently developed a location dictionary for the Washington DC metropolitan area. In this dictionary we assign specific codes to metro stations. These station codes are mapped in two ways; firstly to the localities and secondly directly to locations of the stations. We search tweets for the locality or the station name. If both are found, more weight is given to the name of the station. If the tweet only contains the locality, then we map the tweet out to all the stations that serve the locality and continue to search incoming data for location updates. If it contains the name of the station, we map the tweet to the location of the station. Figure-3 below shows a visualization of these two steps. The first tweet is mapped to multiple stations with downtown Arlington. A few minutes later a newer tweet is mapped to the exact location station.

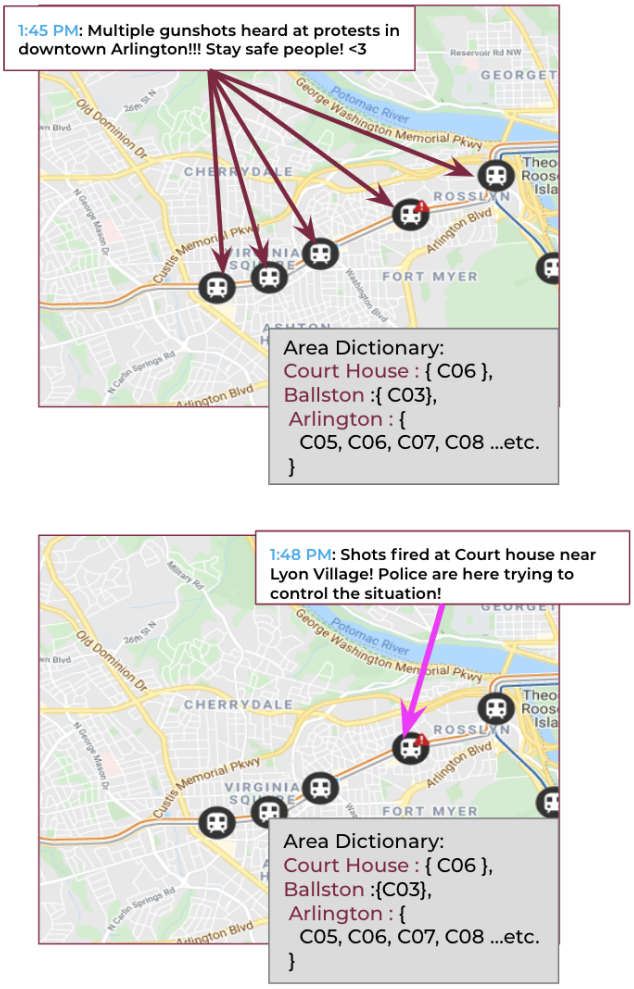

Fig. 3. Location Extraction

C. Threat Detection and Visualization

The Incidents are are accessible from the UI through 3 major components: the real time storyline panel, the station marker and the alert notification pop ups.

*1) **Real Time Incidents Panel**:* The real time panel on the left provides the user with the latest information about any occurring incidents at any station. Tweets related to incidents are collected by timestamp and are used to construct a real-time storyline. Each incident related tweet is tagged under a specific category which is displayed on the yellow label. The user is also provided a link to the original tweet itself. Figure 4 provides a concept of the real-time panel.

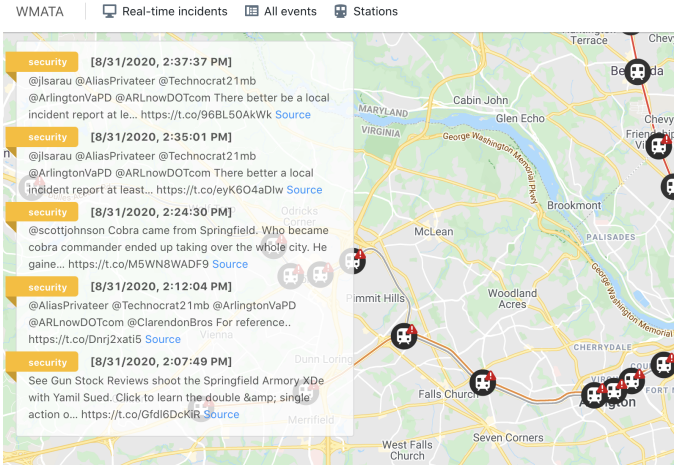

Fig. 4. Real Time Incidents Panel

*2) **Alert Notification System**:* The alert notification system allows users to subscribe to multiple stations and the system provides an immediate alert notification when an incident is detected. The alert notification system also updates the latest follow-up information once the authority validates the authenticity of the event. Figure 5 illustrates the scenario of our application pushing an alert notification for first event-related tweets posted and then the verified event notification.

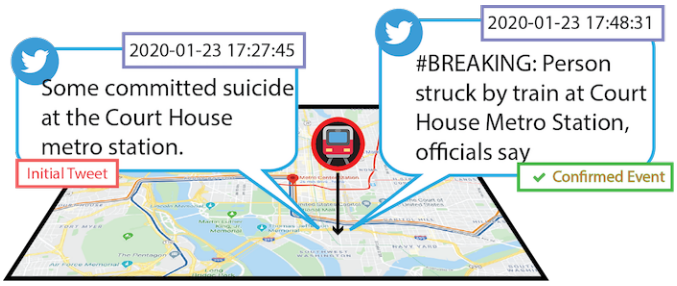

Fig. 5. Alert Notification System

*3) **Station Marker**:* Station markers with a red warning sign indicate a disruption at the station due to a security incident. Clicking on the station marker will display two small pop ups. One shows the details of the station itself and the other displays a timestamped storyline which contains events specific to that particular station. This allows users to navigate to the station of their choice on the map and stay updated with any recent incidents at that station. Figure-5 shows a concept of this component.

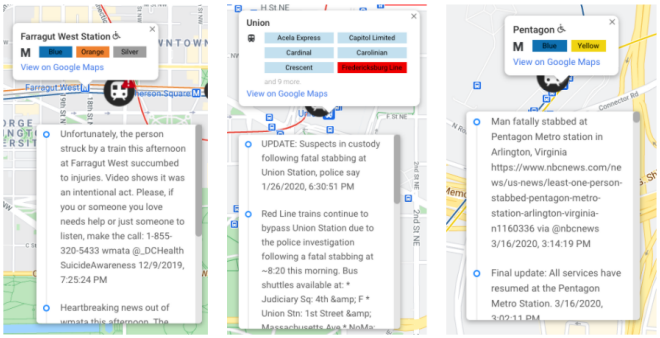

Fig. 6. Station Event List

IV. Case Studies

We found three cases to demonstrate our application. For the first case study we setup a pipeline of tweets sorted by earliest to latest to simulate a mock version of the Twitter streaming API on our local machine. The last two incidents were detected by our system. All these notifications and updates pop on the real time panel and the event list for the respective station(s) as shown in Figures 3 and 5. The first tweet received sends a notification to application, notifying the user of the incident as shown in Figure 4. That very first tweet was the result of the data collected by our initial seed keywords from Table I. These cases verify the promptness property of Twitter data mentioned in the introduction.

A. Shaw-Howard Metro Shooting 07/19/2019

The initial outbreak of incident was made on Twitter at 12:58 pm, when a user tweeted about witnessing someone getting shot at Shaw Howard Metro Station. Once we had our initial data, the Dynamic Query Expansion module performs multiple iterations on new and incoming data to expand it's set of keywords(query) for the particular shooting incident. As the query expansion algorithm gathers more data from incoming news and updates about the incident, the application is updated notifying the user of any major related news. After the first tweet, the first news report was on FOX news DC at 1:34 pm, almost 40 minutes after the incident.

B. Pentagon Metro Stabbing 03/16/2020

The application received an initial tweet of an African American male being stabbed at Pentagon Station at 9:02 am. Over the course of the next few hours as new information comes in, the query expansion is at work. As seen in Figure-2 the algorithm is able to add Pentagon to it's expanded query after the initial tweet, helping us identify the location of the incident. We can also observe that the first news outbreak on the local ABC7 news(WJLA) was at 10:40 am, over an hour and a half after the first tweet. This verifies the promptness of the Twitter data and the effectiveness of the query expansion

component. At 2:58 pm users were informed about the station reopening.

C. Union Station Stabbing 01/26/2020

The initial tweet at 8:37 am mentioned that red line trains were bypassing Union station. At 9:12 am another tweet emerged mentioning that the police were searching for suspects involved in a stabbing incident. Over time as the query expansion is at work new information keeps coming in about the incident and updates about the Red line itself. The first news update was at 10:04 am by ABC7, reporting news about an ongoing search for the suspect. As time passed by, more and more people tweeted about the closure and at 2:30 pm the users were notified of the station reopening.

V. Conclusion

RISECURE is an open-source and automated system that is capable of early detecting transit security-related incidents by using Social Media data mining and dynamic query expansion techniques. The effectiveness of the proposed RISECURE system is demonstrated in this paper using real world cases from the WMATA system. Through the UI we visualize the location of the incidents, develop a notification system for new incident(s) and major updates, and provide a real time storyline for the incident(s). For real-world deployment in transit systems such as metro rails, our proposed approach can serve as a supplementary resource to aid in earlier detection, gain situational awareness, and early deployment of resources to contain the situation. We foresee a great potential to take this platform to a nationwide level where it can help improve the rider experience for the public transit systems.

References

[1] "Metrorail ridership grew by 20,000 trips per weekday in 2019," Jan 2020. [Online]. Available: https://www.wmata.com/about/news/2019-Metrorail-ridership.cfm

[2] W. L. Waugh Jr, "Securing mass transit: a challenge for homeland security," *Review of Policy Research*, vol. 21, no. 3, pp. 307–316, 2004.

[3] B. De Longueville, R. S. Smith, and G. Luraschi, "" omg, from here, i can see the flames!" a use case of mining location based social networks to acquire spatio-temporal data on forest fires," in *Proceedings of the 2009 international workshop on location based social networks*, 2009, pp. 73–80.

[4] X. Zhang, Z. Chen, W. Zhong, A. P. Boedihardjo, and C.-T. Lu, "Storytelling in heterogeneous twitter entity network based on hierarchical cluster routing," *2016 IEEE International Conference on Big Data (Big Data)*, pp. 1522–1531, 2016.

[5] S. Aslam, "Twitter by the numbers: Stats, demographics fun facts," Feb 2020. [Online]. Available: https://www.omnicoreagency.com/twitter-statistics/

[6] L. Zhao, F. Chen, J. Dai, T. Hua, C.-T. Lu, and N. Ramakrishnan, "Unsupervised spatial event detection in targeted domains with applications to civil unrest modeling," *PloS one*, vol. 9, no. 10, 2014.

[7] R. P. Khandpur, T. Ji, Y. Ning, L. Zhao, C.-T. Lu, E. R. Smith, C. Adams, and N. Ramakrishnan, "Determining relative airport threats from news and social media," in *Twenty-Ninth IAAI Conference*, 2017.