Framework for Real-Time Event Detection using Multiple Social Media Sources

Satya Katragadda
Center for Advanced Computer Studies
University of Louisiana at Lafayette
satya@louisiana.edu

Ryan Benton*
School of Computing
University of South Alabama
rbenton@southalabama.edu

Vijay Raghavan
Center for Visual and Decision Informatics
University of Louisiana at Lafayette
raghavan@louisiana.edu

Abstract

Information about events happening in the real world are generated online on social media in real-time. There is substantial research done to detect these events using information posted on websites like Twitter, Tumblr, and Instagram. The information posted depends on the type of platform the website relies upon, such as short messages, pictures, and long form articles. In this paper, we extend an existing real-time event detection at onset approach to include multiple websites. We present three different approaches to merging information from two different social media sources. We also analyze the strengths and weaknesses of these approaches. We validate the detected events using newswire data that is collected during the same time period. Our results show that including multiple sources increases the number of detected events and also increase the quality of detected events.

1. Introduction

Information sharing is a common feature across web-based social platforms today. Numerous event related information is shared through various social media websites like Flickr, Twitter, Tumblr, YouTube, etc. Twitter has 280 million active monthly users[1], generating over 500 million tweets daily[2]. Tumblr has 312 million blogs, with 139 billion posts on them posting 42 million posts a day[3]. This large user base makes these social media platforms some of the largest and fastest information sources [1]. Twitter and Facebook were used to spread information during the anti-government protests in Egypt during the Arab Spring [2, 3]. Information about an earthquake in Japan was tweeted within 2 seconds of the earthquake compared to 20 minutes for an alert to be issued[4]. Social media is also widely used by public officials for communication and outreach to the community [4].

Originally, research on detecting a change in narratives or new information was studied in Topic Detection and Tracking (TDT). This research used well-formatted sources like news articles, blog posts and academic papers [5, 6, 7]. The majority of the factors that make social media an important source of information for event detection, such as the variety of information, a huge volume of data and velocity of the data also makes it difficult to apply traditional event detection models. Further, the type of information posted on social media depends on the media platform; examples include status updates for Facebook, video messages for Vine, and pictures for Flickr. In addition to the above problems, most of the information posted on these platforms is informal, with misspellings and missing context; furthermore, the posts are usually short. These problems make it difficult to apply traditional TDT techniques to data extracted from social media data.

In recent days, there has been substantial work done on detecting events through Twitter [8]. Most of the work on event detection concentrates on Twitter due to its ease of access and quick propagation of information. Event detection has been tested on other plat-

*This work was done when the author was at University of Louisiana at Lafayette

[1] http://www.internetlivestats.com/twitter-statistics/

[2] https://about.twitter.com/company

[3] https://www.tumblr.com/about

[4] urlhttp://www.justmeans.com/blogs/japan-earthquake-on-twitter-social-media-trends-during-disaster

URI: http://hdl.handle.net/10125/41362
ISBN: 978-0-9981331-0-2

forms with varying results. Wavelet-based spatial technology is used to detect events from photo tags on Flickr [9]. Event detection systems are tested on different data sources like Google Plus, Twitter and Facebook [10]. Wikipedia is another, non-traditional source, for detecting events; in this case, unusual activity on a page the world can be used to identify and denote an event [11]. Wikipedia is also used to verify the events that are detected using Twitter [12].

As different sources generate different types of information, it would be advantageous to leverage the strengths of each source to improve event detection. First: Previous research indicates that some types of information are spread on traditional media faster than on social media and vice-versa [13]. Combining information from multiple sources can help detect an event faster. Second: Using multiple sources can increase in the quantity of information can lead to better defining the event. Third: Information propagated on social media is unreliable or contains claims that are not vetted by traditional journalists. Hence, combining information from various social media sources should enhance the event detection process.

One solution to combine the information from two sources of information is to combine various data sources into a single stream and process them as such. The problem with combining information from two data sources is two-fold. First, the information in these data sources is not homogenous, i.e. different data streams generate different kinds of information. For example, Twitter contains messages that are 140 characters long, Flickr is exclusively made up of pictures and associated tags, and newswire contains larger text articles. This makes directly combining the information difficult. Second, the velocity of data from these streams is not uniform, i.e. Twitter may generate tens of thousands of messages about an event which would only produce a small number of articles on the newswire. The data stream unifying models should be able to take these problems into account.

In this paper, we extend the Event Detection at Onset (EDO) model, which detects an event within 3-8 minutes after the event is mentioned on Twitter [14]. In particular, we are interested in combining the information extracted from both Twitter and Tumblr to detect events. The proposed model has two goals: First, design methodologies to effectively combine information from two different data sources within a real-time constraint (usually within a minute). Second: study the efficiency of proposed models to identify events from social media streams. We evaluate the events detected using multiple data sources against the events mentioned in the main stream news sources, which are validated by humans. A real world scenario is simulated using Twitter, Tumblr and traditional news media to evaluate the event detection process. The results demonstrate that the events detected using multiple data sources are better than those detected using a single data source.

The remainder of the paper is organized as follows. In section 2, we discuss some of the existing event detection work on both Twitter domain and using multiple data sources. In section 3, we describe the existing methodology to detect events and extensions to the model to adapt it to work with multiple data streams. In section 4, we provide information for the corpora used for evaluation and experimental setup. In section 5, we present our results and discuss the impact of various. Finally, in section 6, we provide our conclusions and discuss future extensions.

2. Related Work

This section provides a review of research done using multiple domains to aid in the detection of an event. While some work on detecting results from Twitter will be presented, the reader is referred to Aiello et al. [8] for a more through coverage of event detection via Twitter.

Ciglan et al. proposed Wikipop, a personalized event detection system based on Wikipedia page view statistics [11]. The Wikipop system presents to the user a set of Wikipedia articles that are popular based on his/her interest. The popularity of an article is based on the increase in page views of the article. The assumption behind their approach is events covered in public sources trigger an increase in the number of visits of corresponding Wikipedia articles. Ahn et al. also uses Wikipedia page views to detect events [15]. A set of articles whose daily page views for a fifteen day period substantially increases over the previous fifteen day period are identified. These articles are clustered using k-means and topic modeling to group similar articles together. Each detected cluster corresponds to an event. Li et al. proposed a model to identify events based on detecting events through tweets [16]. Events are detected by identifying phrases that are bursty in a given time period; clustering is then performed to group similar phrases based upon both content of the tweets containing the phrases and the temporal pattern of the phrases. However, most of the clusters that are detected are noisy. To eliminate noisy events, the phrases are tested for ”newsworthiness”. The newsworthiness of an event is calculated based on the importance of individual words in the phrase in the context of Wikipedia. A word is considered important if the word appears as anchor text in articles containing that word.

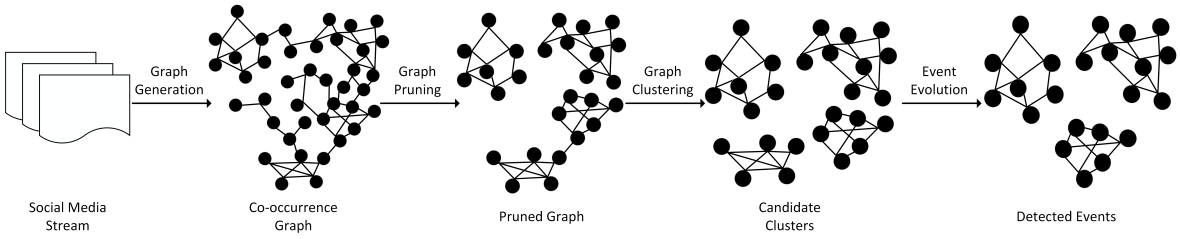

Figure 1: Workflow to Detect Events at Onset from a Single Data Stream

Osborne et al. use Wikipedia page views to improve the quality of events detected through First Story Detection [12]. Events are detected by building tweet clusters using local sensitivity hashing. Detected events from Twitter are cross-checked with Wikipedia page views to filter out wrongly detected events and improve the quality of discovered events. The main problem with this approach is that the lag between Wikipedia edits and page views is too high to aid in real-time event detection. According to the Osborne et al., the average lag between the Twitter stream and Wikipedia page views was 1.36 hours. Steiner et al. proposed an alternate model to detecting events using information from Wikipedia and social media [17]. The authors detect events by tracking the number of edit spikes on Wikipeida pages and then validating the event on social media.

Subasic et al. studied the effect of tweets and traditional news articles on each other [18]. The similarity is calculated based on studying underlying language models. The authors found that Twitter users neither create the news or peddle the news. Instead, Twitter users extend the news by commenting on them. Thus, text from social media can provide more information regarding the events as they are happening.

Most recent work on social media enhances the quality of events by combining information from various multimedia sites. Becker et al. examined how to extract information about related content from predictable events from Twitter, YouTube and Flickr [19]. Time and location information is extracted from multiple websites to gather information about the event. This information is used to query other social media websites to gather relevant information. They demonstrated that information from one social media document can be used to detect related documents from other social media sites, contributing to the diversity of information collected.

Abhik et al. extracted information from multiple social networking sites like Youtube, Flickr, and Twitter to identify new events or changes in already existing events [20]. They use the textual information to detect bursty words and these words are used to identify an event. Then meta-data like content creation time, location information is used to further identify sub-events. To our knowledge, these are the only works that use information from multiple social media sources to identify events. However, they all treat the information from all sources as equally important.

In the next section, we present three different models to merge information from multiple media sources.

3. Methodology

In this section, we briefly explain our chosen existing model to detect events from a single data stream. Then we follow up with different models to combine information from multiple social media sources.

3.1. Event Detection at Onset

The methodology to detect events on onset is taken from Katragadda et al. to detect events from real-time Twitter streams [14]. The workflow of the event detection model is presented in Figure 1. The Event Detection at Onset (EDO) model processes tweets in micro-batches of a minute each. Given the minute between batches, the EDO is constrained to limit its processing time. In the event detection system, the data undergoes three transformations from the data source stage to finally detect events: graph generation phase, graph pruning phase, and event detection phase. In the following paragraphs, we briefly explain the model.

During the graph generation phase, all the text from a single time period is tokenized into tokens where each token is a single word. The word tokens are transformed into a co-occurrence graph, where each node is a word token and an edge between nodes is the number of tweets in which the two words appear together. To avoid random spikes in word usage and to reduce noise, graphs from previous five time periods are aggregated into a single graph. The resulting graph is a highly dense and is pruned into a smaller size to efficiently cluster it.

The graph pruning step identifies two types of nodes in the graph: emerging words and important words.

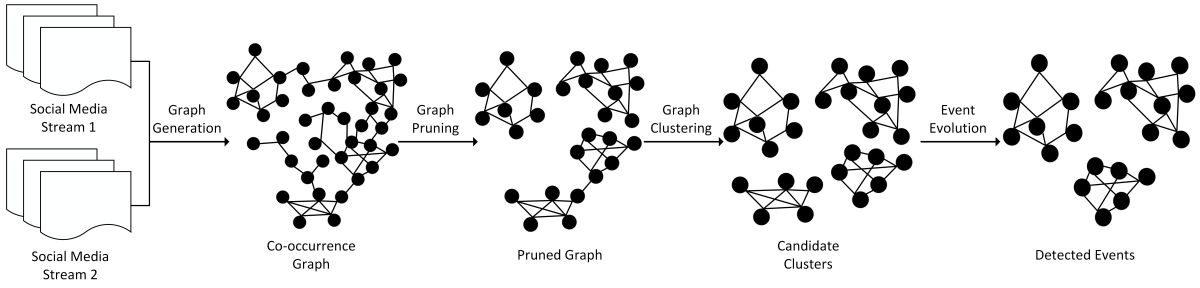

Figure 2: Workflow to Detect Events at Onset from a Multiple Data Streams: Graph Generation

These words are identified using Kullback-Leibler (KL) divergence score, which measures the change in the frequency of a word during the recent k time periods compared to its historical frequency in the previous s time periods. Kullback-Leibler divergence score, KL for word w at time t can be defined as,

$$KL_w^t = p \log_2 \left(\frac{p}{q} \right) \tag{1}$$

where p and q are the recent and historic probability distributions of word w respectively. They are defined as,

$$p = \frac{\sum_{i=t-k}^{t} f(w_i)}{k}, \quad q = \frac{\sum_{i=t-(k+s)}^{t-k} f(w_i)}{s} \tag{2}$$

where $f(w_i)$ is the frequency of word w during time period t.

Emerging words are identified as words that historically have not been used frequent but recently underwent a sudden surge in usage. A term is considered emerging if $KL_w^t > \alpha.KL_w^{(t-k)}$, where α is a user parameter. Important words are those that have been used by users fairly regularly, but have recently seen increased usage. A term is considered important if $KL_w^t > \beta$, where β is a user parameter. Important terms cannot be classified as emerging, but cannot be ignored. All the words in the graph that are not emerging or important are removed. All the important words that are not within two hops of emerging nodes are removed as well.

In the event detection phase, the pruned graph is clustered using voltage based clustering algorithm [21]. This algorithm clusters the graph in linear time which helps keep the processing time low. Each cluster from the clustering phase is considered an candidate event. Many of the candidate events are noise and are not credible events. More credible events stay active and evolve over time as new information is discussed as event progresses. In order to identify noisy events and to identify flow of event over time, event evolution is employed. In event evolution, current event clusters are compared to previous event clusters in order to identify those clusters that are similar. *KL* divergence score is used to identify the similarity of events over time. Instead of identifying the most diverse words, we try to identify the event clusters that are most similar to previous clusters in order to identify a continuing event. The evolution of cluster from previous time period to current time period that has at least 4 tokens in common is measured using the following formula.

$$KL_{c^t,c^{t-1}} = \frac{\sum_{i=0}^{n} w_i^{c^t} \log_2 \left(\frac{w_i^{c^t}}{w_i^{c^{t-1}}} \right)}{n}$$

where c^t is cluster at time period t and c^{t-1} is event cluster at *t-1* and n is the number of terms in cluster c^{t-1}. An event cluster is considered continuing only if $KL > \delta$. In this study, the value of d is set to 3; hence, a cluster is considered an event only if it continues over 3 time periods.

To adapt this model to include multiple data sources, we identified three different ways to merge the information from these data streams: during graph generation, after graph pruning and merge post-clustering. We will discuss each these three models in the next sub-sections.

3.2. Event Detection at Onset for Multiple Steams: Graph Generation Scenario

The most straightforward model is to combine both data streams into a single stream and treat it as such. The workflow for this model is represented in Figure 2. All the textual information from two data streams is collected during a single time period. Each data stream is processed separately, which includes data cleaning and tokenization. Special characters, URLs and other stream specific identifiers like retweets, etc. are stripped from text. Each textual stream is tokenized into individual tokens or words. A graph is generated from all the tokens available.

The main problem with this approach is real-time

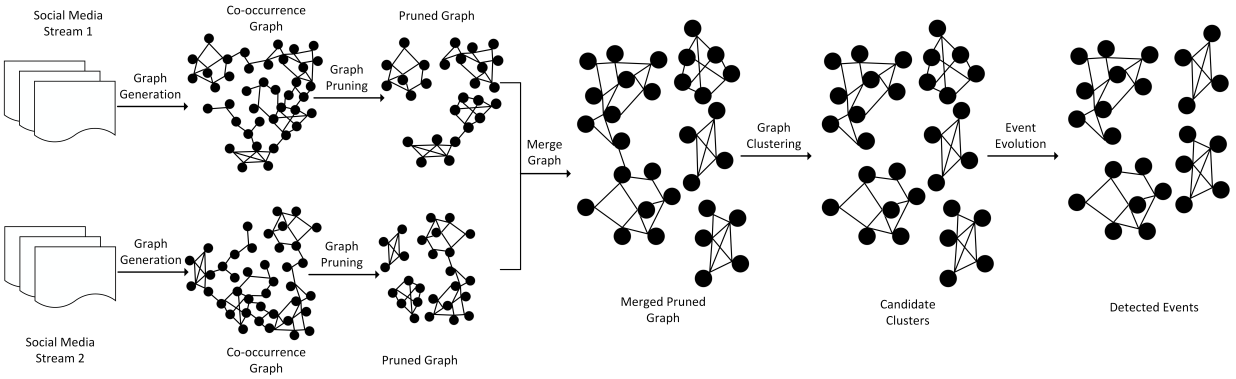

Figure 3: Workflow to Detect Events at Onset from a Multiple Data Streams: Graph Pruning

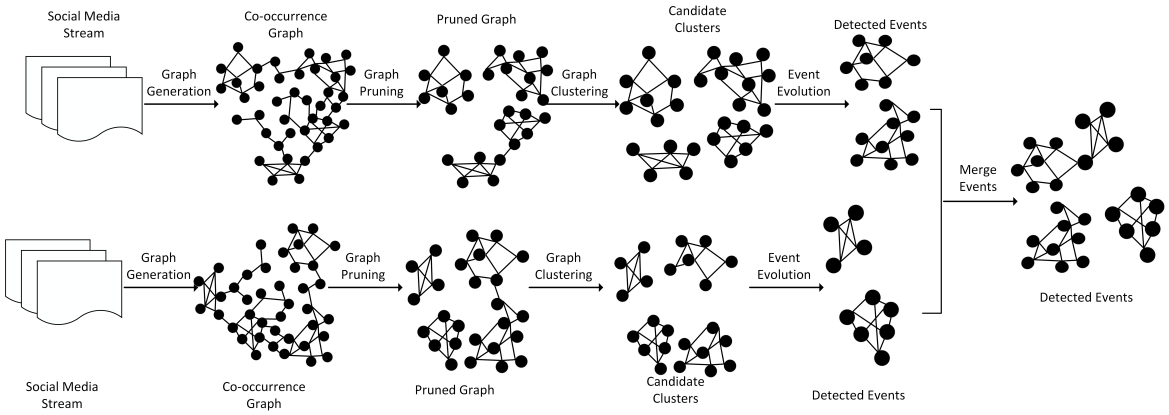

Figure 4: Workflow to Detect Events at Onset from a Multiple Data Streams: Post Clustering

data streams from different sources are different. For example, textual information from Flickr or Tumblr is in the form of tags associated with images or posts. The information from Twitter is in the form of small text message of 140 characters. If the information is extracted from newswire or YouTube description, the information is a large text article. Each of these data streams has their own characteristics that might be lost if they are all treated equally.

Another important problem is the velocity of information. For instance, Twitter users post 7255 posts per second whereas Instagram users upload 724 images at the same time. The disparity in velocity poses challenge in setting the user-defined parameters for identifying emerging and important words. If the parameters are set low in order to incorporate capture terms only appearing in the smaller (less velocity) data streams, noisy terms from streams of higher velocity will appear. This is due to the fact that the parameters are based on number of times terms appear. Hence, words appearing often in the higher velocity stream would be considered (by the parameters) as relevant. On the other hand, if the parameters are tuned to reduce the noise, the useful information found in smaller streams are often eliminated, as they don't occur enough times to be considered not-noise. The other two models to merge data streams takes these problems into account.

3.3. Event Detection at Onset for Multiple Steams: Graph Filtering Scenario

An alternative model for combining information from multiple data streams is to process each data stream separately and generate graphs for two data streams separately, while retaining the information within these data streams. The workflow for this model is presented in Figure 3.

Preprocessing and data cleaning is performed on each data stream separately. The graphs are filtered, where different set of parameters are used for each data source to retain important words from the data sources. Pruned graphs from both data sources are merged based

on common tokens between these two pruned graphs. The node weights, edge weights are updated to reflect the sum of node and edge weights in the individual graphs. The merged graph contains individual word tokens from Twitter and phrases from Tumblr, which are merged based on single tags from Tumblr. The merged graph is them clustered, where each cluster is an event. One issue with this model is the merged graph is denser than individual pruned graphs and hence the number of candidate clusters generated from the merged graph are less than those from individual pruned graphs. This results in low amount of noise, while also keeping the number of detected events low. In practice, this model's ability to detect events is an improvement from the previous scenario, but it does tend to eliminate smaller events by combining them with larger, more popular events.

3.4. Event Detection at Onset for Multiple Steams: Post Clustering Scenario

The third model to merge information multiple data streams is to treat each data stream as a single data stream and combines the candidate event clusters after they are detected. The workflow for this model is presented in Figure 4.

The graphs are generated, filtered and clustering is performed on individual data streams. The candidate events are then checked for evolution with events from previous time period for each data stream. The resulting event clusters from each stream are merged based on similarity. A cosine similarity model is used to identify the similarity between clusters from the validated event clusters. A vector of individual words are generated from clusters from Twitter and Tumblr (tags or phrases are broken down into individual words for this step). Each cluster is represented by the vector of normalized frequencies of words relative to the total frequency of all words in the cluster based on the following formula.

$$sim(c_{tweet}, c_{tumblr}) = \frac{c_{tweet} \cdot c_{tumblr}}{|c_{twitter}| \cdot |c_{tumblr}|} \quad (3)$$

Where c_{tweet} and c_{tumblr} are the clusters represented as vectors from Twitter and Tumblr respectively. Two clusters are considered to be related to the same event if $sim(c_{twitter}, c_{tumblr}) > \lambda$ and both clusters have at least 4 words in common between them. If more than one cluster from a data source is labeled for merging with a cluster from another data source, the cluster combination with larger similarity is merged.

4. Experimental Setup

In this section, we present the description of data used for analysis. Next, we explain the experimental setup for the datasets. We then compare the results from the proposed models to events detected from individual data sources.

4.1. Data

We collected posts from Twitter and Tumblr from April 30th, 2013 to May 6th, 2013. A query based model is used to retrieve data from Twitter and Tumblr using a list of keywords used by Department of Homeland Security (DHS) to monitor social media [22]. Twitter data is gathered using gardenhose using Twitter4j[5] and Java, which returns up to 1% of all tweets that passes through their system which contains at least one word in the query list. Our analysis finds that we never hit the 1% limit and we were able to capture all the tweets that contain at least one word from the query list. Tumblr on the other hand does not have an automated model to push posts, instead traditional retrieval models are used to search for relevant posts for queries. Jumblr[6] is used to access Tumblr api to retrieve most recent post that contains each word in the keyword list as a query one at a time. All the words are searched in a round-robin fashion to retrieve new posts for a keyword since the last search. Based on our analysis it takes around 2 minutes to search through the 187 keyword list. In total, we collected 2.1 million tweets and 348 thousand Tumblr posts during the first week of May when this data was collected. We had an average of 208 tweets and 38 Tumblr posts per minute.

4.2. Parameters

To evaluate the performance of combining multiple data streams, this model is compared against event detection performed the Twitter and Tumblr data individually. A detailed analysis on the effect of α and β was conducted by Katragadda et al. [14]. Increase in value of α and β increases the precision but also decreases the recall. Experiments were performed on this dataset for both Twitter and Tumblr data to determine the best case values of the different thresholds and best case scenarios were picked for various combination of parameters. Unless stated otherwise, the values of α and β are set to 5 and 100 for Twitter data stream, while the values for Tumblr data stream are set to α=2 and β=30.

Depending on the scenario, some of the parame-

[5] http://twitter4j.org/en/

[6] https://github.com/tumblr/jumblr

Table 1: Comparison of Results Different Models

	Twitter	Tumblr	Merge: Graph Generation	Merge: Graph Filtering	Merge: Post Clustering
Number of events detected	291	85	291	332	342
Number of events that were not detected	56	262	56	15	5
Noise (clusters created that are not events)	43	47	43	52	74
Precision	0.871	0.644	0.861	0.865	0.822
Recall	0.839	0.216	0.839	0.957	0.986
F-Score	0.855	0.323	0.85	0.909	0.897
Execution time (Seconds)	42	23	43	44	46

ters may change. For the merge during graph generation scenario, the default parameters for Twitter stream are adapted for the combined data stream to reduce the noise. For merge during graph filtering scenario, the default parameters for Twitter and Tumblr are adapted to account for the reduced velocity of data on Tumblr . This provides an easier mechanism to identify important words on Tumblr without them getting labeled as noise. For the scenario in which we merge detected events post clustering, the number of common nodes, as discussed in Section 3.4 must be 4. In addition, the value of λ is set to 0.25. All these values are set based on analysis of results for best-case scenario.

4.3. Validation

One issue that remains to be discussed is how we determine whether a detected "event" is real. To do this, we generate ground truth for the detected events using news stories during the time an event is active. Once an event was detected, all the words and phrases in the event cluster were used to search Google, Bing and Yahoo news aggregators for at least 2 hours after the event. The news that were collected during this process while not exhaustive was able to identify news that are related to the event. These detected events are annotated by 3 annotators using the news stories as a reference. A majority judgment is used to label each event cluster as an event or not. The inter annotator agreement between different annotators is measured using Fleiss' Kappa [23]. The annotator agreement is 0.67, which is considered substantial.

It is important to note that it is impossible to identify all the events mentioned on social media, it is equally impossible to identify all the events happening in the world at a given time. In order to study the efficiency of different models, we look at total number of true events that actually happened and noise, i.e. events detected that are not actually events in the world. Hence, the precision mentioned in Table 1 is a true precision while the recall is actually the number of events missed by a method that is actually detected by another method in the data.

5. Results

5.1. Comparison of Results

All the posts in the datasets were simulated using a client server architecture. Each post from Twitter and Tumblr are pushed to the client from the server based on creation time of that post. The tweets are collected in micro batches of a minute each, while the Tumblr posts are collected in a round robin fashion from results obtained from queries executed in the same time period. All the posts from the previous 5 minutes are compared to the posts made in the past hour to aid in real-time detection of the events.

In Table 1, we present the analysis for events detected on both Twitter and Tumblr individually as well as results for Merge: Graph Generation, Merge: Graph Filtering and Merge: Post Clustering. In the graph, *the number of events detected* show the total number of events detected by each method. The precision, recall and F-Score are calculated using the following formula.

$$Precision = \frac{Number\ of\ Events\ Detected}{Number\ of\ Events\ Detected\ +\ Noise} \tag{4}$$

$$Recall = \frac{Number\ of\ Events\ Detected}{Actual\ Number\ of\ Events} \tag{5}$$

$$F-Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{6}$$

Actual number of events is calculated by adding the number of events detected and the number of events not detected.

Table 2: Events Detected using Twitter and Tumblr

Event Clusters from EDO	**Event**
Twitter	
evacuated, dearborn, district, courthouse	Dearborn Court evacuated due to bomb threat
price, puts, hiding, cop, fugitive, cuba,fbi	FBI puts 2 million price on fugitive in cuba, who was charged with killing a cop
evacuated, nypd, raymond, bronx, threat, bomb	School in Raymond, bronyx evacuated due to bomb threat
airport, person, fired, shots, iahhouston	Shots fired at Houston airport
dropped, stuff, fall, recently, switzerland, scientists, antimatter, nuclear	Switzerland scientists drops antimatter
flash, illinois, watch, issued, cdt, nws, central, warning, thunderstorm	NWS waring to flash floods in Illinois
center, memphis, thunderstorm, watch, warning, issued, cdt, nws, severe, tornado	Tornado warning in Memphis
Tumblr	
Occupy Sandy, Forte Green, Occupy WallStreet, Mayday Protests	Occupy wall street sets up distribution center for mayday protests
Nobama, guantanamo, hunger strike, free prisoners	Obama called to free the prisoners from Guantanamo
Curtin Springs, smoke and fire	Picture of fire near Curtin Springs, Australia

Table 3: Types of Events, Detection and Verification Time

Type of Event	Events in Dataset	Detection Time (mins)	Time difference between Social Media and News (mins)
Armed Conflicts & Attacks	131	6.33	18.46
Arts, Culture & Entertainment	16	12.26	-34.32
Business & Economy	12	10.38	-7.19
Disasters & Accidents	97	6.54	13.21
Law, politics & Scandals	18	8.59	-3.41
Science & Technology	8	10.37	-9.39
Sports	65	6.18	22.51

The Event Detection at Onset model, when applied on Twitter and Tumblr datasets individually was able to identify 291 and 85 events respectively. The execution time for Twitter data took 42 seconds to identify an event for 1-minute micro batch of data, while Tumblr data took 23 seconds for the same time period. Tumblr generated less number of events compared to Twitter and a large number of noise clusters, i.e. the clusters that do not correspond to an events. This leads us to believe that Tumblr is not a good source of information by when it is used as the only source of information for breaking news.

Combining the information from both these streams at the Graph generation phase does not identify any extra events compared to just using the Twitter data, while increasing the execution time by a second. Our analysis show that the data from Tumblr is minuscule compared to data generated from Twitter and is considered noise by the EDO model. However, the other two models to combine the information were able to identify events that were missed just by Twitter or Tumblr.

Merge: Graph Filtering was able to identify 332 events of 347 in the combined dataset in 44 seconds. We were able to detect 39 more events by combining the datasets rather than using Twitter as the only source of information. The model generated 52 events that cannot be verified. The number of noisy events is larger than those detected by Twitter or Tumblr. In our opinion, the number of extra events detected by combining streams offsets the minor increase in the number of noisy events. The precision decreases when multiple sources are combined, but recall increases considerably compared to only Twitter and Tumblr. The F-Score also increases when two data sources are combined.

The final model Merge: Graph Clustering identified 342 events of the 347 verifiable events in the dataset. This model was the best model to identity more than 98% of the verifiable events in the dataset. The model also generated comparatively larger number of events that are considered noise. This is due to the number of noisy events that are generated by Twitter and Tumblr passed on as events, compared to the previous step, where these types are events are eliminated post graph filtering phase, which either generated a single

noisy cluster instead of multiple noisy clusters. The model also took 2 seconds longer to execute compared to Merge: Graph Filtering.

Finally, Merge: Graph Filtering and Merge: Graph Clustering identifies larger number of events compared to using a single stream for event detection or combining the streams at the start of the process. The main trade off between using these two models to detect events is the extra execution time compared to single streams.

Table 2, shows the clusters of events related to the event and actual event detected using our event detection system for both Twitter and Tumblr. All the clusters generated were evaluated by human to label each cluster as an event or not based on the words that form the cluster and the tweets and Tumblr entries that the clusters were based upon. All of these events were recognized as new events by the system. Certain words that are required to identify the context of the cluster is missing due to low scores in the graph pruning process. This would result in the event clusters to be incoherent. Tweets and context were also presented along with event clusters to give a better understanding of event clusters.

5.2. Types of Events

The detected events are classified into six categories based on their description on Wikipedia. Events that do not have a Wikipedia entry are classified manually by the three annotators. Table 3 provides a breakdown of total number of events in the dataset and average time taken to detect various kinds of events by different models and traditional news media. The EDO model is very effective at identifying emerging events across the world, mainly armed conflicts and attacks as well as disasters and accidents within 6.33 minutes and 6.54 minutes respectively. Traditional news media takes about 18.46 minutes to report news about armed conflicts and attacks, 13.21 minutes to report news about accidents and disasters. This is mainly because these are the type of events that break on social media and are then followed through by traditional news reports. Another type of events where social media leads traditional news outlets is sports reporting. The main reason for this is that information is generated on social media thousand times faster and propagated and the lag on traditional media can be attributed to lag in publishing the information rather than information gathering capability. Traditional news stories prefer to generate a larger news piece about a match rather than push minute-by-minute updates on a sporting event. Among all these types of events Twitter usually leads the conversation compared to Tumblr, but the latter contributes more in multimedia content. The other type of events like arts, culture and entertainment, business and economy, law politics and scandals, science and technology traditional news outlets lead the conversation and social media follows the conversation with dedicated groups of users propagating the news articles coupled with their own commentary. These are also the stories that are more popular on Tumblr, where most of these events are detected first compared to Twitter.

6. Conclusion and Future Work

In this paper, we proposed a novel, real-time approach to merge multiple streaming data streams using time-evolving graphs. Three different approaches are presented to merge multiple streams. We also tested these approaches and presented the strengths and weaknesses of each approach. We demonstrated that using information from multiple data streams increases the quality and quantity of detected events. This model can be applied to detect events and track changes over time.

The models in this paper can be extended to incorporate other sources like Instagram, SnapChat, and RSS feeds etc. to detect better events. The quality of detected events can also be increased by combining information from multimedia data like images, videos to associate them with detected events. Another extension to this work is to automatically validate the events using newswire data.

7. Acknowledgements

This work is supported in part by the award NSF/IIP - 1160958 from the Computer and Information Science and Engineering (CISE) Directorate of the National Science Foundation and in part by the Industry Advisory Board of Center for Visual and Decision Informatics. The authors would like to thank Robert Boland from Johnson & Johnson and Gary Edwards of Lockheed Martin for their valuable suggestions that contributed to this work.

8. References

[1] A. Dong, R. Zhang, P. Kolari, J. Bai, F. Diaz, Y. Chang, Z. Zheng, and H. Zha, "Time is of the essence: improving recency ranking using twitter data," in *Proceedings of the 19th international conference on World wide web*, pp. 331–340, ACM, 2010.

[2] K. Starbird and L. Palen, "(How) will the revo-

lution be retweeted?: information diffusion and the 2011 Egyptian uprising," in *Proceedings of the acm 2012 conference on computer supported cooperative work*, pp. 7–16, ACM, 2012.

[3] M. F. Alkazemi, B. J. Bowe, and R. Blom, "Facilitating the egyptian uprising: A case study of facebook and," *Cases on Web 2.0 in Developing Countries: Studies on Implementation, Application, and Use*, p. 256, 2012.

[4] A. Mills, R. Chen, J. Lee, and H. Raghav Rao, "Web 2.0 emergency applications: How useful can Twitter be for emergency response?," *Journal of Information Privacy and Security*, vol. 5, no. 3, pp. 3–26, 2009.

[5] J. Kleinberg, "Bursty and hierarchical structure in streams," *Data Mining and Knowledge Discovery*, vol. 7, no. 4, pp. 373–397, 2003.

[6] G. Kumaran and J. Allan, "Text classification and named entities for new event detection," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 297–304, ACM, 2004.

[7] Q. He, K. Chang, and E.-P. Lim, "Analyzing feature trajectories for event detection," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 207–214, ACM, 2007.

[8] L. M. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Goker, I. Kompatsiaris, and A. Jaimes, "Sensing trending topics in twitter," *IEEE Transactions on Multimedia*, vol. 15, no. 6, pp. 1268–1282, 2013.

[9] L. Chen and A. Roy, "Event detection from flickr data through wavelet-based spatial analysis," in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 523–532, ACM, 2009.

[10] M. Osborne and M. Dredze, "Facebook, twitter and google plus for breaking news: Is there a winner?," in *International AAAI Conference on Web and Social Media*, 2014.

[11] M. Ciglan and K. Nørvåg, "Wikipop: personalized event detection system based on wikipedia page view statistics," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, pp. 1931–1932, ACM, 2010.

[12] M. Osborne, S. Petrovic, R. McCreadie, C. Macdonald, and I. Ounis, "Bieber no more: First story detection using twitter and wikipedia," in *SIGIR 2012 Workshop on Time-aware Information Access*, 2012.

[13] S. Petrović, M. Osborne, and V. Lavrenko, "Streaming first story detection with application to Twitter," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 181–189, Association for Computational Linguistics, 2010.

[14] S. Katragadda, R. Benton, S. Virani, and V. Raghavan, "Detection of event onset using twitter," in *2016 International Joint Conference on Neural Networks*, pp. 1539–1546, IEEE, 2016.

[15] B. G. Ahn, B. Van Durme, and C. Callison-Burch, "Wikitopics: What is popular on wikipedia and why," in *Proceedings of the Workshop on Automatic Summarization for Different Genres, Media, and Languages*, pp. 33–40, Association for Computational Linguistics, 2011.

[16] C. Li, A. Sun, and A. Datta, "Twevent: segment-based event detection from tweets," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 155–164, ACM, 2012.

[17] T. Steiner, S. Van Hooland, and E. Summers, "Mj no more: using concurrent wikipedia edit spikes with social network plausibility checks for breaking news detection," in *Proceedings of the 22nd International Conference on World Wide Web*, pp. 791–794, ACM, 2013.

[18] I. Subašić and B. Berendt, "Peddling or creating? investigating the role of twitter in news reporting," in *Advances in Information Retrieval*, pp. 207–213, Springer, 2011.

[19] H. Becker, D. Iter, M. Naaman, and L. Gravano, "Identifying content for planned events across social media sites," in *Proceedings of the fifth ACM international conference on Web search and data mining*, pp. 533–542, ACM, 2012.

[20] D. Abhik and D. Toshniwal, "Sub-event detection during natural hazards using features of social media data," in *Proceedings of the 22nd international conference on World Wide Web companion*, pp. 783–788, International World Wide Web Conferences Steering Committee, 2013.

[21] F. Wu and B. A. Huberman, "Finding communities in linear time: a physics approach," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 331–338, 2004.

[22] "Analyst's desktop binder," *Department of Homeland Security National Operations Center Media Monitoring Capability Desktop Reference Binder*, pp. 20–23, 2011.

[23] J. L. Fleiss and J. Cohen, "The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability.," *Educational and psychological measurement*, 1973.