Real-time Timeline Summarisation for High-impact Events in Twitter

Yiwei Zhou[1] and **Nattiya Kanhabua**[2] and **Alexandra I. Cristea**[3]

Abstract. Twitter has become a valuable source of event-related information, namely, breaking news and local event reports. Due to its capability of transmitting information in real-time, Twitter is further exploited for timeline summarisation of high-impact events, such as protests, accidents, natural disasters or disease outbreaks. Such summaries can serve as important event digests where users urgently need information, especially if they are directly affected by the events. In this paper, we study the problem of *timeline summarisation of high-impact events that need to be generated in real-time*. Our proposed approach includes four stages: classification of real-world events reporting tweets, online incremental clustering, post-processing and sub-events summarisation. We conduct a comprehensive evaluation of different stages on the "Ebola outbreak" tweet stream, and compare our approach with several baselines, to demonstrate its effectiveness. Our approach can be applied as a replacement of a manually generated timeline and provides early alarms for disaster surveillance.

1 Introduction

Social media services, such as Twitter, have become a popular platform for communication in everyday life and in the time of crisis. In case of critical situations, Twitter demonstrates its usefulness when users urgently need information, especially if they are directly affected by *major events*, for example, disease outbreaks or natural disasters. Due to the prevalence of events reporting and collective attention in Twitter, numerous works have leveraged tweets for detecting real-world events, e.g. [34, 5].

Real-world events in Twitter can be classified into two main categories, based on the level of impact. For instance, the events "Gulf of Mexico oil spill", "Ebola outbreak" and "Zika virus outbreak" are regarded as *major events* that have high impact, which can influence people worldwide; whereas the events "Charlton Road Closure for London Marathon" and "Three people were released from a lift at Pescod Square" refer to *local events*, with a low impact on specific groups of people.

The consumption of event-related stories in Twitter can be a tedious task that requires cognitive effort, due to the overwhelming amount of texts, as well as the presence of noisy and redundant information. Moreover, a large proportion of tweets are mundane discussions, irrelevant to real-word event detection. In case of tweets reporting about an event of interest, they might contain a large amount of near-duplicates, in which the main content conveys the same meaning, with slightly different word usages [34, 11, 17].

In this paper, we focus on *sub-events* detection of a known *major event*, to automatically generate a real-time timeline for the *major event* in a format as in `https://en.wikipedia.org/wiki/Ebola_virus_epidemic_in_West_Africa_timeline`, which is the most widely used timeline for internet users to understand the "Ebola outbreak" temporally.

The generation of timeline summaries can ease the comprehension of major events in news stream [31] and social media, such as Twitter. Our timeline summaries consist of sub-events or key incidents relevant to a given major event. The sub-events show the status of the ongoing major event. They earn compatible attention at a similar scale with their associated major event, but this attention can only last for a few days, or even shorter, as they will be superseded by the following sub-events. For example, "On March 24, two suspected cases in Liberia are announced by the Liberian Ministries of Information, Culture, Tourism, and Health. The government had also stated that Ebola had 'crossed over into Liberia,' but did not confirm the information." is a sub-event of the major event "Ebola outbreak". By using a chronological order, a timeline can represent the temporal development of the major event. Thus, our main task is to detect sub-events and to provide concise and non-redundant summaries. Furthermore, a timeline must be generated in real-time, in order to help users follow recent updates about the high-impact events, according to their interest.

Few researches have been done in the area of sub-event detection and timeline generation, which include [25, 16, 18]. Our approach is different from former ones in the following ways: we differentiate between real-world events reporting tweets and other tweets, by applying only event-independent features; we propose an online incremental clustering algorithm, to handle different levels of duplicated tweets reporting on the same sub-event, which makes real-time timeline generation possible; considering the evolvement characteristic of major events, we propose a post-processing step, to improve clustering performance and reduce computational cost. As we only use event-independent features overall, the approach can be easily adapted to other major events. We perform a thorough evaluation of the proposed approach on the "Ebola outbreak" tweet stream, and verify its advantages based on several evaluation metrics, over the baseline approaches.

Our approach can be an efficient supplement or even replacement of the user-generated timeline. Its real-time characteristic not only can eliminate the lag between user-generated timeline on Wikipedia and news reports [14], but also can help to generate early alarms for disasters.

[1] Department of Computer Science, University of Warwick, UK, email: Yiwei.Zhou@warwick.ac.uk
[2] Department of Computer Science, Aalborg University, Denmark, email: nattiya@cs.aau.dk
[3] Department of Computer Science, University of Warwick, UK, email: A.I.Cristea@warwick.ac.uk

2 Real-time Timeline Summarisation for High-impact Events in Twitter

In this section, we present a real-time timeline summarisation approach for real-world major events in Twitter, as show in Fig. 1. As a preprocessing step, we POS-tag the tweets in the English tweet stream, which mention the pre-known target entity(ies) related to the studied major event. The POS-tags can provide features for the subsequent stages. This is achieved by the CMU Part-of-Speech Tagger [24] for tweets. According to [24], it can achieve more than 90% accuracy on various tweets datasets, which fully satisfies our needs. After preprocessing, we filter out tweets in the stream that are not *real-world events reporting tweets*. Moreover, we apply an online incremental clustering algorithm to cluster the near-duplicate tweets reporting on the same sub-event, in real-time. Furthermore, we adapt a post-processing step, to generate more precise results and remove clusters reporting terminated sub-events. We update the summaries of sub-event clusters, as long as there are new tweets to be included, and order these summaries chronologically, which constitute the timeline of the major event.

2.1 Extraction of Tweets Reporting Real-world Events

In [34, 11, 17], researchers have pointed out that about 50% of the tweets on Twitter are not relevant to real-world events. In our approach, we first filter all the tweets in the stream by the major event's relevant entity(ies), to reduce the number of irrelevant tweets. We further differentiate the tweets that report real-world events from the tweets that express personal feelings, or pointless "babbles", to avoid the "mundane" and "polluted" information [5].

We train a binary classifier, to determine if one incoming tweet is a real-world events reporting tweet or not. We explore the differences in expression patterns between real-world events reporting tweets and other tweets. These expression patterns, which are event-independent, form the features set. Event-dependent features, such as the n-grams, are excluded. One set of features are Twitter syntax features that are commonly used in tweet-related classification, which include: the number of hashtags in the tweet, the number of at-mentions in the tweet. Another set of features are indicators of other users' reactions to this tweet, which include: the number of retweets of a tweet, whether the tweet has been "favourited", as we expect that Twitter users are apt to have different reaction patterns when reading about tweets reporting real-world events, from other tweets expressing personal feelings. Compared with real-world events reporting tweets, Twitter users are more likely to include informal language, such as emoticons and abbreviations, in tweets expressing personal feelings. Moreover, Twitter users like to use interjections, exclamation marks, question marks in personal feeling expressing tweets, to stress the tone used. On the contrary, fact-related information, such as numbers, URLs and locations are frequently mentioned in real-world events reporting tweets. We further include all these above features into the features set. The number of emoticons, abbreviations, interjections, numbers and URLs in the tweet can be obtained through the CMU POS-tags. The number of exclamation marks and question marks can be obtained by simple character matching. We calculate the number of locations mentioned in the tweet by checking the inclusion of pre-specified location names in the noun phrases obtained after POS-tagging. The pre-specified location names are extracted through gazetteer lookup, the scope and granularity of which can be configured based on the characteristics of the major event, to improve efficiency.

We do not include the user profile features and the occurrence of a tweet's geo-tag information into the features set, as it has been shown that those features cannot help to improve the classifier's performance through cross-validation. This may be due to the fact that the major events usually attract the attention of all kinds of Twitter users, from public accounts of news agencies to regular personal accounts, no matter where their physical locations are. Besides that, the Twitter's retweet function and the "Tweet Button" on webpages make it much easier for Twitter users with different backgrounds to report real-world sub-events related to the major event.

2.2 Sub-event Detection in Tweet Stream

Because of the huge volume of daily posts on Twitter, a large percentage of them can be seen as redundant, as they only report on the sub-events that are already reported by other tweets. From our observation, for most of the time, the tweets reporting the same sub-event are near-duplicate tweets. In [30], researchers distinguished near-duplicate tweets on 5 levels, which were: *exact copy*, *near exact copy*, *strong near-duplicate*, *weak near-duplicate* and *low-overlapping*. For *exact copy*, *near exact copy*, *strong near-duplicate* and *weak near-duplicate* tweets, the main parts are identical or almost the same. For *low-overlapping* tweets, they only have a couple of common key terms, but greatly vary in word usages and expression patterns.

In [30], researchers treated the near-duplicate detection as a classification problem and the classifier had to make a decision on every pair of tweets that were possible to be near-duplicates. Their near-duplicate detection strategy worked well on a small scale, but it needed human annotation of tweets from various domains to train the classifier, and its computation complexity was really high. On the other hand, traditional online clustering algorithms, based on the similarities of TF-IDF representation vectors of tweets' textual content [25, 7], inevitably have the following drawbacks: (i) The iteratively updated IDF information can be biased, if the number of processed tweets is not large enough, or the processed tweets are not randomly sampled from the whole tweets dataset; (ii) Tweets are shorts texts, the role of some rare terms can be dominating, when calculating the similarities between tweets using their TF-IDF representation vectors; (iii) Researchers usually reduce the dimensionality of the representation vectors, by selecting the tokens with high IDF values only, but it is questionable to equal rareness with importance, especially when the IDF information is not reliable, as some valuable information can be easily lost; (iv) By setting a reasonable threshold, this kind of online clustering algorithms may have acceptable performance on *exact copy*, *near exact copy*, *strong near-duplicate* and *weak near-duplicate* tweets, but they can hardly deal with *low-overlapping* tweets, which occupy 18.8% of all kinds of near-duplicate tweets, according to [30].

Another drawback of most current clustering algorithms for event detection in tweets is that they do not consider the variances in word usages, which are highly frequent because of tweets' short and informal characteristics, and their rich syntax features. For example, tweet t_1: "#Senegal sends medical teams to border with #Guinea after an outbreak of Ebola there. #Sierra Leone, much closer to the epicenter, hasn't." and tweet t_2: "Senegal has sent a medical team to all its main border crossing points with Guinea after an outbreak of Ebola... `http://fb.me/6WqOZ2b3h`" are talking about the same sub-event. They contain some key terms that vary in representation forms, but have the same meaning, such as *#Senegal* and *Sene-*

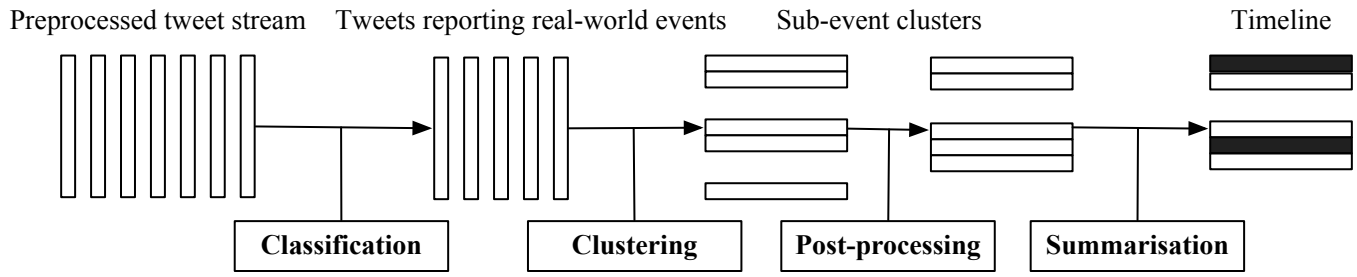

Figure 1. Timeline generation of tweet stream.

gal, *medical teams* and *medical team*, *Guinea* and *#Guinea*, which should be treated as the same terms. However, traditional clustering algorithms treat these key terms, which share the same meaning, but only vary slightly in representation forms, as different terms. As a result, the similarity of these two tweets decreases, and they cannot be included in the same cluster when the *clustering threshold* is high. However, alternatively blindly lowering the *clustering threshold* can cause the decrease of the *clustering precision* (defined in Section 3.3).

To solve the above problem and reduce the dimensionality of tweets' representation vectors, as well as increase the *clustering precision* and decrease the *compression ratio* (defined in Section 3.3), we propose an online incremental clustering algorithm, as shown in Algorithm 1.

Algorithm 1: Sub-event Detection on Tweet Stream

input : t, tweet stream; E, target entity(ies)
output: $ProcessingClusters$, clusters of tweets reporting the same *sub-event*

```
ProcessingClusters = ∅;
foreach Tweet t_i in t mentioning E do
    Preprocess t_i;
    if t_i is reporting a real-world sub-event then
        Initialise a cluster c_i with t_i, U_i (useful URLs in t_i), and
        K_i (key terms in t_i);
        foreach cluster c in ProcessingClusters do
            if cluster c has common useful URL with c_i then
                MergeClusters(c, c_i);
            else if GetSimilarity(c, c_i) > clustering
            threshold then
                MergeClusters(c, c_i);
            else
                add c_i to ProcessingClusters;
            end
        end
    end
end
```

Algorithm 1 incrementally clusters the tweets based on common URL(s) and key terms sharing the same meaning. We try to reduce its computational cost as it needs to process the incoming tweets in real-time.

When a new tweet comes, after preliminary target entity(ies) filtering, preprocessing and classification, we obtain the tweets reporting real-world sub-events belonging to the pre-known major event. To eliminate the noise, we only use some *key terms* in one tweet to represent this tweet, which are: noun phrases, verbs, hashtags, URLs, numbers and at-mentions. The choice is made based on the observation that for tweets reporting the same sub-event, these key terms would be the same or almost the same, but the other parts of the tweets, such as conjunctions, adjectives and adverbs, often vary. In this way, the dimensionality of the tweets' representation vectors is reduced. Since we already have the POS-tags after the preprocessing step, we only need chunking to extract the key terms, and lemmatisation to transform the verbs from their various inflected forms to their original forms.

It is of high probability that tweets containing URLs are closely related to the content of the linked webpages [1]. Some researches have used this kind of tweets as the summaries or highlights of the sub-events reported by the linked webpages [33]. This has shown that the benefits of the assumption that tweets containing URLs represent highlights of the sub-events reported by the linked webpages, overweigh the risks. Based on the above assumption, new tweets are incorporated into the processing sub-event cluster with which it shares common URL(s). Two sub-event clusters are considered to report on the same sub-event, if they contain common URL(s). We do not take the full actual content of the webpages into account, to avoid the inclusion of noisy information. Because of the characters limitation of Twitter, the URLs contained in the tweets are mostly shortened in various ways, to save space. After retrieving the original URLs, we consider the URLs that contain nothing but domain and category information, such as `http://NBCNews.com` and `http://www.nbcnews.com/news` to be *useless*, as this kind of URLs provide no information about the sub-events. One original URL would only be consider as *useful*, if it contains the concrete address of a real-world sub-event reporting webpage.

The prioritised *URL-based clustering strategy* can help to enrich the processing sub-event cluster with the key terms that report the sub-event from different angles. On the other hand, if an incoming tweet does not contain any common URL with any processing sub-event cluster, it is still possible to be incorporated into one processing sub-event cluster. This is achieved by the *threshold-based clustering strategy*. As mentioned before, the problem that key terms appear in slightly different forms, but have the same meaning, widely occurs in tweets. For example, the occurrences of "#Liberia", "Liberian" and "Liberia's" have the same effect as the occurrence of the country name "Liberia". To deal with this problem, we treat two different key terms as the same key term, as long as the Jaro-Winkler metric between them is above a threshold. This method can further reduce the dimensionality of the tweets' representation vectors. As in our approach, each dimension in tweets' representation vectors represents a set of key terms that share the same meaning, rather than only one key term. Jaro-Winkler metric is specially designed for short strings matching, which is based on the length of the longest common prefix, the number and order of the common characters between two strings [10]. In [10], researchers replaced the exact token matching

with approximate token matching, based on the Jaro-Winkler metric; in [9], researchers compared various personal name matching techniques, and the Jaro-Winkler metric was one of the techniques with the best performance. We set the Jaro-Winkler metric threshold to 0.9, following [10]. Based on the above description, we define $GetSimilarity(c_1, c_2)$ in Algorithm 1 as follows:

$$GetSimilarity(c_1, c_2) = J_{JW}(K_1, K_2) = \frac{K_1^{'} \cap K_2^{'}}{K_1^{'} \cup K_2^{'}} \quad (1)$$

where: c_1 and c_2 denote two clusters, K_1 and K_2 denote the key terms mentioned in c_1 and c_2. We replace the traditional Jaccard similarity metric based on exact matching (J) with a Jaccard similarity metric based on the Jaro-Winkler matching (J_{JW}). In Eq. 1, $K^{'}$ represents sets of key terms, with all the key terms in the same set sharing the same meaning. A new key term can be incorporated into one of the sets, as long as the Jaro-Winkler metric between this new key term and any one of the key terms that are already in the set is above 0.9. For two sets of key terms, k_i and k_j, they are viewed as belonging to the same set, if the Jaro-Winkler metric between one key term from k_i and one key term from k_j is above 0.9.

All information about the processing sub-event cluster, such as the above mentioned key terms and URL(s), will be updated, as long as it incorporates new tweets.

In [36], researchers pointed out that clustering algorithms utilising the Jaccard similarity metric achieved better performance than the ones utilising the cosine similarity metric, because of the sparsity of tweets. Similar to [39], we choose the Jaccard similarity metric when evaluating the similarity between two tweets' representation vectors. Since only key terms in the tweets are considered, we do not have to face the problem that Jaccard similarity metric cannot deal with terms with different levels of importance.

2.3 Post-processing of Detected Sub-events

Algorithm 2: Post-processing of Detected Sub-events

```
input  : ProcessingClusters; M, termination threshold; D,
         target period
output: ProcessingClusters, TerminatedClusters,
         clusters of tweets reporting the same sub-event
TerminatedClusters = ∅;
foreach Day i in D do
    ProcessingClusters =
    HierarchicalClustering(ProcessingClusters);
    foreach cluster c in ProcessingClusters do
        if c has not incorporated sub-event updates for M days
        then
            move c from ProcessingClusters to
            TerminatedClusters;
        end
    end
end
```

The proposed online incremental clustering algorithm (Algorithm 1) inevitably has some drawbacks. First, the fact that an incoming tweet is incorporated into one processing sub-event cluster as long as certain conditions are met, ignores the possibility that there are other processing sub-event clusters, which may also meet these conditions. Second, the information in clusters is dynamic; so one cluster can become more similar to some other clusters, after incorporating some tweets. To solve the above problems, we apply a more rigid and computationally-intensive hierarchical clustering algorithm on processing sub-event clusters. We consider all the tweets in the new generated clusters reporting on the same sub-event.

A hierarchical clustering algorithm needs the distance matrix of all the items to be clustered as the input, thus it is not suitable for online scenarios. However, after the online incremental clustering, the number of items to be clustered (processing sub-event clusters) is much lower than the original number of tweets, which greatly reduces the computational overhead. Moreover, since the hierarchical clustering algorithm aims at fixing the miss outs of Algorithm 1, it has lower priority, and thus can be processed offline, at the end of each day, or during any less busy time. We use a similar strategy as in Algorithm 1 to compute the distance matrix of the processing sub-event clusters, as the input of the hierarchical clustering algorithm: for two processing sub-event clusters, their distance is 0, if they mention common *useful* URL(s); their distance is the Jaccard distance of their representation vectors, otherwise. We use the same *clustering threshold* in Algorithm 1 as the *cutting threshold* in the hierarchical clustering algorithm, to guarantee that a similar standard is applied. We choose single-linkage hierarchical clustering, aiming at merging clusters that contain the closest pair of sub-event clusters into a new cluster. In this way, we can deal with the following scenario: if there exist sub-event clusters reporting on the same sub-event from different angles in one intermediate cluster; another intermediate cluster can be further merged with this intermediate cluster, as long as it contains sub-event clusters reporting on the sub-event from any angle.

Since sub-events last shorter and have narrower influence, we also consider temporal features of processing sub-event clusters, to further reduce the computational cost of Algorithm 1, and the hierarchical clustering algorithm. We set up the following rule: a sub-event can be seen as terminated, as long as there is no new tweet reporting on this sub-event for M days, since the sub-event cluster's last incorporation. If one sub-event has terminated, its identity will be changed from *processing sub-event cluster* to *terminated sub-event cluster*. We discard the possibility that an incoming tweet reports on a terminated sub-event, thus it is not possible for terminated sub-event clusters to incorporate new tweets. We also discard the possibility that a processing sub-event cluster reports on the same sub-event with any terminated sub-event cluster, thus terminated sub-event clusters are not considered by the hierarchical clustering algorithm. M is dependent on the prior knowledge about the major event. This rule can improve the efficiency of the whole approach, but it inevitably compromises the overall performance. Thus we recommend setting M based on the detailed application scenario and not to a number less than 15, based on the regular lasting period of sub-events.

2.4 Timeline Summarisation

We extract timestamps and summaries of the sub-events described by the clusters, and rank them in chronological order, to generate the real-time timeline. Based on the fact that all the tweets in the same cluster are near-duplicate tweets, we select the most representative tweet in each sub-event cluster as the summary of the described sub-event, as in [19, 18, 29]. Both temporal and textual features are considered when generating items for the timeline, as shown in Algorithm 3.

We select the tweets mentioning the highest number of key terms with different meanings, as the *candidate representative tweets*. This

Algorithm 3: Generation of Items for the Timeline

input : c, a cluster of tweets reporting the same *sub-event* ($ProcessingClusters + TerminatedClusters$)
output: s_c, the summary of this *sub-event*; T_c, the timestamp of this *sub-event*; T_{s_c}, the posted timestamp of the summary tweet

$MaxSimilarity = 0$, $T_c = CurrentTime$, $T_{s_c} = CurrentTime$;
foreach *new incorporated tweet* t_i *in cluster* c **do**
  if *extracted timestamp from* t_i*'s text* $< T_{t_i}$*(*t_i*'s posted timestamp)* **then**
    $T'_{t_i} \leftarrow$ extracted timestamp from t_i's text;
  else
    $T'_{t_i} \leftarrow T_{t_i}$;
  end
  Initialise a cluster c_i with t_i's key terms K_i;
  if `GetSimilarity`$(c_i, c) > MaxSimilarity$ **then**
    $MaxSimilarity \leftarrow$ `GetSimilarity`(c_i, c);
    $s_c \leftarrow t_i, T_{s_c} \leftarrow T_{t_i}$;
  end
  if `GetSimilarity`$(c_i, c) = MaxSimilarity$ *and* $T_{s_c} < T_{t_i}$ **then**
    $s_c \leftarrow t_i, T_{s_c} \leftarrow T_{t_i}$;
  end
  if $T_c > T'_{t_i}$ **then**
    $T_c \leftarrow T'_{t_i}$;
  end
end

is out of the reason that the representative tweet should contain as much information as possible. From the candidate representative tweets, we select the one that has the most recent posted timestamp (T_{t_i}) as the summary of this sub-event. This is due to the fact that the summary should contain the newest update of the sub-event.

As for the timestamp of the sub-event, we combine the extracted timestamps from temporal expressions in tweets from the dateparser[4] with the tweets' posted timestamps, similar to [28]. This is because users are likely to post tweets reporting past sub-events. For example, the tweet "Good news! No confirmed cases of Ebola recorded by the Sierra Leone government in their 20 March daily report. `http://reliefweb.int/report/sierra-leone/ebola-outbreak-updates-march-20-2015` ..." is posted on 21st March 2015, one day after the occurrence of the sub-event. The timestamp of the sub-event is set to be the earliest posted timestamp of all the tweets in its corresponding cluster, only if no earlier timestamp can be extracted from the tweets; otherwise we use the extracted timestamp instead.

3 Evaluation

3.1 Dataset Description

We applied our proposed real-time timeline summarisation approach on Ebola Tweets dataset of the TREC Dynamic Domain Track[5]. This dataset contains 165,000 tweet-ids, while only 90,823 of them, which were posted during a period from 31 Jan 2014 to 23 Mar 2015, can be accessed. It should be noted that only a small percentage of tweets in this dataset are related to the "Ebola outbreak". We processed the downloaded tweets in the order of their posted timestamps, to simulate the tweet stream. The known major event for our evaluation is "Ebola outbreak". We filtered out all the non-English tweets, and used "Ebola" as the target entity, in order to filter out tweets that were not related to the considered major event.

[4] `https://dateparser.readthedocs.org/`
[5] `http://trec-dd.org/`

3.2 Evaluation of Extraction of Real-world Events Reporting Tweets

We utilised CrowdFlower, a crowdsourcing website, to annotate 3,000 tweets, which were randomly sampled from the dataset, into two categories: *real-world events reporting tweets* and *other*. Only 2,103 *real-world events reporting tweets* and 333 *other* tweets were left, after we filtered out all the annotated tweets with confidence lower than 0.9. Since there was a big difference between the numbers of items from these two categories, we balanced the dataset, to avoid bias. We used grid search to find the most suitable parameters for a Support Vector Machine (SVM) classifier based on the average F1 score. A SVM classifier using RBF kernel, with the kernel coefficient (γ) set to 0.3125 and penalty parameter (C) set to 8 achieved the best performance. The precision, recall and F1 score of the classifier generated through 10-fold cross validation is shown in Table 1.

Table 1. Performance of the extraction of real-world events reporting tweets.

Metric	Other	Real-world	Average
Precision	0.828	0.761	0.795
Recall	0.730	0.850	0.790
F1 Score	0.779	0.805	0.792

A recall of 0.850 was achieved on the *real-world events reporting tweets* category using this classifier, which meant about 85.0% of the *real-world events reporting tweets* related to the major event can remain after this stage, which fully satisfied our needs.

Even though we used the "Ebola outbreak" dataset to train the classifier, only event-independent features were employed, as illustrated in Section 2.1. Thus, the classifier's performance will be less affected than other classifiers that are employing event-dependent features, when categorising tweets related to other major events.

We extracted 7,069 *real-world events reporting tweets* in English about the "Ebola outbreak" without performing any clustering algorithm, after processing the whole tweet stream.

3.3 Evaluation of Sub-event Detection

The cosine similarities between tweets' TF-IDF representation vectors have been widely applied in several online clustering researches [3, 8, 40], where both the centroids of clusters and IDF weights of the terms were iteratively updated. We implemented the threshold-based online clustering algorithm utilising cosine similarity metric between tweets' TF-IDF representation vectors (denoted by Cosine-TFIDF) as a baseline. Another baseline we implemented was a similar algorithm as Cosine-TFIDF but using the Jaccard similarity metric instead (denoted by Jaccard). We also compared the performances of the proposed algorithm with and without the post-processing step.

Unlike [28, 21], we define a stricter way to measure the *clustering precision*. The *clustering precision* is defined as the percentage of *positive clusters* in all the generated clusters that *contain more than one tweet*, after processing the whole tweet stream. A cluster can

only be counted as a *positive cluster*, if *all* the tweets in the cluster describe the same *sub-event*. The judges consisted of two student volunteers and one of the authors. For each sub-event cluster, we provided the judges with all the tweets in the cluster, and asked them to annotate it as a *positive cluster* or a *negative cluster*. To avoid bias, the judges were kept unaware about any configuration information for each unannotated cluster.

Since we lacked the ground truth about all the sub-events during the "Ebola outbreak", it was infeasible to calculate the recall. In [35], researchers defined *reduction ratio*, as the ratio of the size of the original dataset to the size of the reduced dataset. Similarly, in [22], researchers defined *compression ratio* as the ratio of the size of the summarised text documents to the size of the original text documents. Both of the above evaluation metrics were used to evaluate the compression ability of clustering algorithms. Similarly, we define the *compression ratio* for our application as:

$$CR = \frac{N_c}{N}. \tag{2}$$

where: *CR* is the *compression ratio*; N_c is the number of the generated clusters, regardless of the number of tweets in the cluster; N is the total number of the tweets in all clusters. After clustering, all the tweets in the same cluster can be compressed into one summary, as they all described the same sub-event and were near-duplicate tweets. When two clustering algorithms reach the same *clustering precision*, the lower the *CR* is that one algorithm achieves, the stronger the cluster algorithm's ability is in clustering near-duplicate tweets describing the same sub-event.

We experimentally set the parameter M in Algorithm 2 to 30, based on the observation that for "Ebola outbreak" related tweets, there was hardly any tweet discussing a sub-event, if this sub-event had not been updated for 30 days.

Table 2 shows the performance comparison of our algorithm and the baselines, after processing the whole tweet stream, where *CT* denotes *clustering threshold*, *CP* denotes *clustering precision*, and *CR* denotes *compression ratio*. We tuned the *clustering threshold* in the range of [0.3, 0.9], with 0.1 increments.

Table 2 demonstrates that the Cosine-TFIDF algorithm has the highest clustering precisions for all the clustering thresholds. However, its high compression ratios show that the Cosine-TFIDF algorithm is quite weak in detecting all kinds of near-duplicate tweets describing the same sub-event. Because of the reasons mentioned in Section 2.2, online clustering algorithms based on cosine similarities of the tweets' TF-IDF representation vectors are not suitable for clustering near-duplicate tweets.

For the other three algorithms, both of the proposed clustering algorithms with and without the post-processing step perform much better than the online clustering algorithm based on Jaccard similarity, in both compression ratio and clustering precision, when the clustering threshold is below 0.9. On one hand, our proposed clustering algorithm only considers the key terms, which can eliminate some noises introduced by tweets mentioning common adjectives and adverbs, but about different objects. On the other hand, our proposed clustering algorithms replace the exact token matching with approximate key term matching based on the Jaro-Winkler metric, and applies the URL-based clustering strategy, both of which contribute to the large increase in compression ratio. When the clustering threshold is 0.9, the online clustering algorithm based on Jaccard similarity can only detect tweets that are exactly the same or almost the same, thus it achieves a higher clustering precision but much lower compression ratio than the proposed clustering algorithms.

The choice between the proposed clustering algorithms with and without the post-processing step should be made based on the real-life application, after some consideration on the balance between compression ratio and clustering precision. The proposed clustering algorithm with the post-processing step achieves slightly better performance in compression ratio than the one without the post-processing step for any clustering threshold, at the price of slightly compromised clustering precision. When setting the clustering threshold to 0.5, the proposed clustering algorithm without the post-processing step's clustering precision is only 0.8% higher than the one with the post-processing step, but its compression ratio is 2.0% higher than the latter one. This is when we recommend choosing the proposed clustering algorithm with the post-processing step over the one without the post-processing step.

We selected the proposed clustering algorithm with the post-processing step and set the clustering threshold to 0.5 for our following evaluations, as that was when the proposed clustering algorithm had the lowest compression ratio when the clustering precision was above 90.0%.

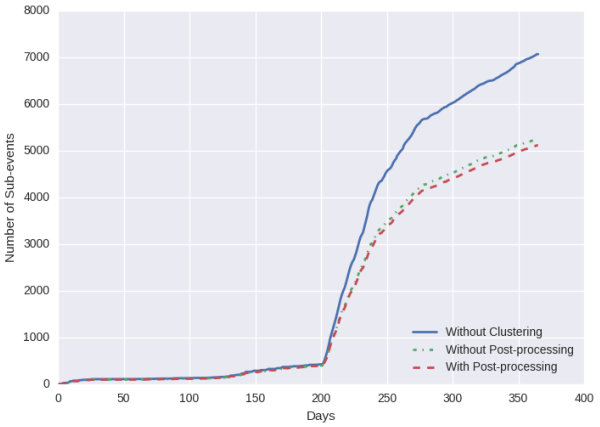

Figure 2. Number of detected *sub-events* for each day during the target period with different settings.

After setting the *clustering threshold* to be 0.5, we use Figure 2 to further illustrate our proposed clustering algorithm's effectiveness in detecting near-duplicate tweets.

3.4 Evaluation of Sub-event Summarisation

We further evaluated Algorithm 3's performance on the "Ebola outbreak" tweet stream. We selected one representative tweet for each sub-event cluster, considering both the amount of information and novelty, based on Algorithm 3. We provided the judges all the generated clusters of tweets, and let them choose one tweet for each cluster that can best represent the sub-event this cluster described. For 82.0% of all the clusters, our summarisation algorithm made coherent choices with human judges. This is significantly better than a baseline *Newest First* algorithm, which took the most recently posted tweet as one cluster's summary and achieved 60.5% in precision.

3.5 A Case Study of the Generated Timeline

Due to space limitation, we randomly sampled the timeline of the "Ebola outbreak" generated with our approach, as shown in Table 3.

Table 2. Performance comparison of different algorithms.

	Cosine-TFIDF		Jaccard		Without Post-processing		With Post-processing	
CT	**CP**	**CR**	**CP**	**CR**	**CP**	**CR**	**CP**	**CR**
0.3	84.0%	94.0%	67.0%	65.7%	80.0%	60.9%	77.9%	59.2%
0.4	94.4%	97.1%	80.0%	74.6%	85.9%	70.4%	83.9%	68.3%
0.5	97.8%	98.6%	85.0%	78.1%	90.8%	74.4%	90.0%	72.4%
0.6	100.0%	99.2%	88.8%	81.5%	93.0%	76.4%	92.0%	75.6%
0.7	100.0%	99.3%	92.9%	85.1%	94.9%	77.9%	93.5%	77.6%
0.8	100.0%	99.4%	93.9%	90.1%	95.4%	79.9%	95.4%	79.8%
0.9	100.0%	99.5%	100.0%	99.2%	95.9%	80.8%	95.6%	80.8%

Table 3. Example timeline generated for "Ebola outbreak".

Date	Location	Timeline
2014.03.23	Guinea	No serious med infrastructure in the area for response. "Guinea confirms Ebola as source of epidemic http://aje.me/1gU7kpU via @AjEnglish"
2014.03.24	Senegal, Liberia, Guinea, Sierra Leone	#Senegal & #Liberia mobilise medics to ward off #Ebola spreading in #Guinea. #SierraLeone much closer to epicenter doing/saying nothing.
2014.03.24	Sierra Leone	#EbolaFever has hit eastern Sierra Leone. Fast action needed please madam #MinisterofHealthandSanitation. This is very serious. God help us.
2014.03.26	Guinea	#Guinea says it has contained #Ebola outbreak in its southeast, but death toll rises and people are scared Reuters http://in.reuters.com/article/2014/03/26/guinea-ebola-idINL5N0MN50D20140326 ...
2014.03.26	Guinea	@WHO does not recommend any travel, trade restrictions to #Guinea & neighbouring countries in respect to this #Ebola outbreak #AskEbola
2014.03.26	Guinea, Liberia	#Ebola virus kills 90% of those it strikes - 63 people have died so far in #Guinea in latest outbreak, 5 in #Liberia http://www.bloomberg.com/news/2014-03-25/ebola-victims-face-90-death-risk-drugs-start-to-emerge.html ...
2015.03.11	UK, Sierra Leone	And now a UK Military Health Worker battling #Ebola in #sierraleone http://edition.cnn.com/2015/03/11/europe/uk-military-ebola/index.html
2015.03.13	Liberia	WHO Confirms No Ebola Case in Liberia in Two Weeks - http://AllAfrica.com http://goo.gl/fb/aDB55B #LIBERIA
2015.03.20	Liberia	#Liberia confirms first #Ebola case in weeks, just as the authorities were beginning the countdown to an Ebola-free nation.
2015.03.20	Sierra Leone	Good news! No confirmed cases of Ebola recorded by the Sierra Leone government in their 20 March daily report. http://reliefweb.int/report/sierra-leone/ebola-outbreak-updates-march-20-2015 ...

After processing the whole "Ebola outbreak" tweet stream, we compared our automatically generated results with the manually generated Wikipedia timeline for the Ebola outbreak in West Africa. We employed the same timeline format as in https://en.wikipedia.org/wiki/Ebola_virus_epidemic_in_West_Africa_timeline, and used all the country names extracted in Section 2.1 from tweets in the same cluster as the locations of the sub-events. We did not use the tweets' geo-tags or user profile locations, because unlike tweets reporting *local events*, most of the tweets reporting real-world *major events* were posted by Twitter users from all over the world, rather than from the neighbourhood of the *local events*.

There were 201 sub-events listed in https://en.wikipedia.org/wiki/Ebola_virus_epidemic_in_West_Africa_timeline from 2014 to 2015, and 126 of them can find their corresponding items from our automatically generated timeline. For example, "No serious med infrastructure in the area for response. 'Guinea confirms Ebola as source of epidemic http://aje.me/1gU7kpU via @AjEnglish"' and "#Liberia confirms first #Ebola case in weeks, just as the authorities were beginning the countdown to an Ebola-free nation", were included in both the user-generated Wikipedia timeline and the automatically generated Twitter sub-events timeline.

On the other hand, a large number of sub-events detected by our approach, such as "And now a UK Military Health Worker battling #Ebola in #sierraleone http://edition.cnn.com/2015/03/11/europe/uk-military-ebola/index.html" and "Good news! No confirmed cases of Ebola recorded by the Sierra Leone government in their 20 March daily report. http://reliefweb.int/report/sierra-leone/ebola-outbreak-updates-march-20-2015 ...", were only included in our timeline. This proves that our automatic timeline summarisation approach for the tweet stream can also work as an efficient supplement of the user-generated timeline.

Moreover, since our approach can detect real-time sub-events of the "Ebola outbreak", it can provide some early alarms for potential outbreaks in some countries. The World Health Organization (WHO), an organisation that always releases convincing worldwide epidemic reports and international travel alarms, usually needs more time to gather enough facts than automatic approaches that extract knowledge directly from social media. The real-time timeline generated by our approach, although with much less authority, still can provide some insights for international travellers and local people to avoid some dangerous areas, and also buy some time for them to get

prepared for the potential coming outbreak. For example, WHO released its first report about this outbreak's situation in Guinea on 25th March 2014[6], in Liberia on 30th March 2014[7] and in West Africa on 1st April 2014[8]. It also released an international travel alarm for this outbreak on 28th March 2014[9]. However, starting with the 23rd March 2014, our approach has already detected some sub-events describing new Ebola outbreaks in some West African countries, which could provide valuable information for some people who do not want to take any risk.

4 Related Work

The most related research topic to this work is Event Detection on Twitter. According to [34, 5], event detection algorithms can be broadly classified into two categories: document-pivot methods, which detect events by clustering documents based on their semantic distances, and feature-pivot methods, which study the distributions of words and discover events by grouping words.

There was a burst of researches performing event detection on Twitter utilising feature-pivot methods recently. [19, 17] extracted all the topical terms for some given events first, then clustered the topical terms based on their co-occurrences or temporal frequency patterns. [34, 20] detected events by capturing the bursts in the terms' appearances. Some feature-pivot methods applied modified Latent Dirichlet Allocation (LDA) on tweets, by incorporating some tweet-specific characteristics. For example, [38] proposed a Twitter-LDA model, which assumed a single topic assignment for an entire tweet; [11] applied the LDA model only on hashtag signals that were identified as events indicators through wavelet signal analysis; [12] proposed a TimeUserLDA model, based on the assumption that tweets reporting global events were likely to follow a global topic distribution that was time-dependent, and tweets reporting personal topics were likely to follow a personal topic distribution that was time-independent; [32] enriched the LDA model with the weights of event terms on timeline and the reliabilities of users to extract social events; [26] applied a LinkLDA model to group tweets from the same event category, based on the assumption that an event term's type distribution was shared across its mentions. This kind of feature-pivot methods can achieve good performance on detecting *major events*. However, they cannot be applied to timeline generation, as they did not consider the near-duplicate characteristic of tweets describing the same *sub-event*. Moreover, in some researches, the detected events were groups of terms, with each group representing one event, which made it quite hard to be interpreted and understood. Besides, most feature-pivot methods can only be applied on offline datasets, thus cannot generate a real-time timeline of one ongoing *major event*.

Different clustering algorithms on tweets have been proposed by document-pivot methods. [6] proposed an ensemble clustering approach that combined multiple clustering solutions. Their features included terms, time in minutes and geographic locations. They needed labeled training data to tune the cluster thresholds and weights for different clustering solutions, which can be quite hard in regular occasions. In some research work [25, 7, 15], they measured the cosine similarities between tweets using TF-IDF representation vectors, with weights of some important terms raised. On the basis of textual similarity, [29] also considered the temporal factor and Twitter users' influence scores when calculating the similarities between tweets. Olteanu et al. [23] considered two tweets to be near-duplicates if their longest common subsequences was 75% or more of the length of the shortest tweet. Unlike these methods, our approach aims at tackling the problem of clustering near-duplicate tweets describing the same *sub-event* from the tweet stream. Special measures towards the *low-overlapping* level of near-duplicate tweets, such as extracting key terms and considering key terms with high Jaro-Winkler metric as the same key term, are taken. [2, 21, 27] utilised Locality Sensitive Hashing (LSH) techniques to group tweets into buckets, and tweets in the same bucket were considered as duplicate tweets. LSH techniques could increase the search efficiency. However, it is quite hard to incorporate some specific strategies into the process, because of the limitation of hash functions.

Another track of related researches would be Disaster Surveillance on Twitter. While Event Detection methods are widely used [13, 27, 4] in this research track, there were also some researches, such as [37], which tried to correlate the number of Ebola outbreaks with the number of the symptoms mentions of Ebola on Twitter. Although [37]'s results showed that the correlation was quite low, our results demonstrate that with detailed textual analysis, Twitter can still provide some earlier alarms than traditional media about the outbreaks in some countries.

5 Conclusion

In this paper, we proposed a real-time timeline summarisation approach for pre-known high-impact events, i.e., *major events*. This approach consists of four stages: real-world events reporting tweets extraction, online incremental clustering, post-processing and sub-events summarisation. Using "Ebola outbreak" as the pre-known major event, we applied our approach on a tweet stream, and evaluated the performance of each stage of the approach. Our results showed that our approach was significantly better than several baselines, in terms of *clustering precision* and *compression ratio*, and could generate early alarms for disaster surveillance. Our approach is generic enough, as only event-independent features are used for all the stages, so it could be applied on various major events. Our automatic timeline generation approach is a promising supplement and replacement of user-generated timeline, which could provide people with more insights about the real-time status of the major events they care about.

[6] http://www.afro.who.int/en/clusters-a-programmes/dpc/epidemic-a-pandemic-alert-and-response/outbreak-news/4065-ebola-virus-disease-in-guinea-25-march-2014.html

[7] http://www.afro.who.int/en/clusters-a-programmes/dpc/epidemic-a-pandemic-alert-and-response/outbreak-news/4072-ebola-virus-disease-liberia.html

[8] http://www.afro.who.int/en/clusters-a-programmes/dpc/epidemic-a-pandemic-alert-and-response/outbreak-news/4073-ebola-virus-disease-west-africa-1-april-2014.html

[9] http://www.who.int/ith/updates/20140328/en/

REFERENCES

[1] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao, 'Semantic enrichment of twitter posts for user profile construction on the social web', in *The Semantic Web: Research and Applications*, 375–389, Springer, (2011).

[2] Puneet Agarwal, Rajgopal Vaithiyanathan, Saurabh Sharma, and Gautam Shroff, 'Catching the long-tail: Extracting local news events from twitter', in *Proceedings of the Sixth International Conference on Weblogs and Social Media*, (2012).

[3] Pramod Anantharam, Krishnaprasad Thirunarayan, and Amit Sheth, 'Topical anomaly detection from twitter stream', in *Proceedings of the 4th Annual ACM Web Science Conference*, pp. 11–14. ACM, (2012).

[4] Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta, 'Tweedr: Mining twitter to inform disaster response', in *Proceedings of the 11th International ISCRAM Conference*, (2014).

[5] Farzindar Atefeh and Wael Khreich, 'A survey of techniques for event detection in twitter', *Computational Intelligence*, (2013).

[6] Hila Becker, Mor Naaman, and Luis Gravano, 'Learning similarity metrics for event identification in social media', in *Proceedings of the third ACM international conference on Web search and data mining*, pp. 291–300. ACM, (2010).

[7] Hila Becker, Mor Naaman, and Luis Gravano, 'Beyond trending topics: Real-world event identification on twitter', in *Proceedings of the Fifth International Conference on Weblogs and Social Media*, (2011).

[8] Deepayan Chakrabarti and Kunal Punera, 'Event summarization using tweets', in *Proceedings of the Fifth International Conference on Weblogs and Social Media*, (2011).

[9] Peter Christen, 'A comparison of personal name matching: Techniques and practical issues', in *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pp. 290–294. IEEE, (2006).

[10] William Cohen, Pradeep Ravikumar, and Stephen Fienberg, 'A comparison of string metrics for matching names and records', in *Proceedings of KDD workshop on data cleaning and object consolidation*, volume 3, pp. 73–78, (2003).

[11] MÁRIO Cordeiro, 'Twitter event detection: Combining wavelet analysis and topic inference summarization', in *Doctoral Symposium on Informatics Engineering, DSIE*, volume 8, pp. 11–16, (2012).

[12] Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim, 'Finding bursty topics from microblogs', in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 536–544. Association for Computational Linguistics, (2012).

[13] Wenwen Dou, Xiaoyu Wang, Drew Skau, William Ribarsky, and Michelle X Zhou, 'Leadline: Interactive visual analysis of text data through event identification and exploration', in *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pp. 93–102. IEEE, (2012).

[14] Besnik Fetahu, Abhijit Anand, and Avishek Anand, 'How much is wikipedia lagging behind news', in *Proceedings of the 2015 ACM conference on Web science. ACM*, (2015).

[15] Hansu Gu, Xing Xie, Qin Lv, Yaoping Ruan, and Li Shang, 'Etree: Effective and efficient event modeling for real-time online social media networks', in *Proceedings of Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on*, volume 1, pp. 300–307. IEEE, (2011).

[16] Arpit Khurdiya, Lipika Dey, Diwakar Mahajan, and Ishan Verma, 'Extraction and compilation of events and sub-events from twitter', in *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*, pp. 504–508. IEEE Computer Society, (2012).

[17] Chenliang Li, Aixin Sun, and Anwitaman 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).

[18] Chen Lin, Chun Lin, Jingxuan Li, Dingding Wang, Yang Chen, and Tao Li, 'Generating event storylines from microblogs', in *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 175–184. ACM, (2012).

[19] Rui Long, Haofen Wang, Yuqiang Chen, Ou Jin, and Yong Yu, 'Towards effective event detection, tracking and summarization on microblog data', in *Web-Age Information Management*, 652–663, Springer, (2011).

[20] Adam Marcus, Michael S Bernstein, Osama Badar, David R Karger, Samuel Madden, and Robert C Miller, 'Twitinfo: aggregating and visualizing microblogs for event exploration', in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 227–236. ACM, (2011).

[21] Andrew J McMinn, Yashar Moshfeghi, and Joemon M Jose, 'Building a large-scale corpus for evaluating event detection on twitter', in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pp. 409–418. ACM, (2013).

[22] NK Nagwani, 'Summarizing large text collection using topic modeling and clustering based on mapreduce framework', *Journal of Big Data*, **2**(1), 1–18, (2015).

[23] Alexandra Olteanu, Sarah Vieweg, and Carlos Castillo, 'What to expect when the unexpected happens: Social media communications across crises', in *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pp. 994–1009. ACM, (2015).

[24] Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith, 'Improved part-of-speech tagging for online conversational text with word clusters'. Association for Computational Linguistics, (2013).

[25] Swit Phuvipadawat and Tsuyoshi Murata, 'Breaking news detection and tracking in twitter', in *Proceedings of Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 3, pp. 120–123. IEEE, (2010).

[26] Alan Ritter, Oren Etzioni, Sam Clark, et al., 'Open domain event extraction from twitter', in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1104–1112. ACM, (2012).

[27] Jakob Rogstadius, Maja Vukovic, CA Teixeira, Vassilis Kostakos, Evangelos Karapanos, and JA Laredo, 'Crisistracker: Crowdsourced social media curation for disaster awareness', *IBM Journal of Research and Development*, **57**(5), 1–4, (2013).

[28] Axel Schulz, Benedikt Schmidt, and Thorsten Strufe, 'Small-scale incident detection based on microposts', in *Proceedings of the 26th ACM Conference on Hypertext & Social Media*, pp. 3–12. ACM, (2015).

[29] Lidan Shou, Zhenhua Wang, Ke Chen, and Gang Chen, 'Sumblr: continuous summarization of evolving tweet streams', in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pp. 533–542. ACM, (2013).

[30] Ke Tao, Fabian Abel, Claudia Hauff, Geert-Jan Houben, and Ujwal Gadiraju, 'Groundhog day: near-duplicate detection on twitter', in *Proceedings of the 22nd international conference on World Wide Web*, pp. 1273–1284. International World Wide Web Conferences Steering Committee, (2013).

[31] Tuan A. Tran, Claudia Niederée, Nattiya Kanhabua, Ujwal Gadiraju, and Avishek Anand, 'Balancing novelty and salience: Adaptive learning to rank entities for timeline summarization of high-impact events', in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pp. 1201–1210, (2015).

[32] Bayar Tsolmon and Kyung-Soon Lee, 'An event extraction model based on timeline and user analysis in latent dirichlet allocation', in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pp. 1187–1190. ACM, (2014).

[33] Zhongyu Wei and Wei Gao, 'Gibberish, assistant, or master?: Using tweets linking to news for extractive single-document summarization', in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1003–1006. ACM, (2015).

[34] Jianshu Weng and Bu-Sung Lee, 'Event detection in twitter.', *ICWSM*, **11**, 401–408, (2011).

[35] Donghui Yan, Ling Huang, and Michael I Jordan, 'Fast approximate spectral clustering', in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 907–916. ACM, (2009).

[36] Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power, 'Using social media to enhance emergency situation awareness', *IEEE Intelligent Systems*, (6), 52–59, (2012).

[37] Elad Yom-Tov, 'Ebola data from the internet: An opportunity for syndromic surveillance or a news event?', in *Proceedings of the 5th International Conference on Digital Health 2015*, pp. 115–119. ACM, (2015).

[38] Wayne Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li, 'Topical keyphrase extraction from twitter', in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 379–388. Association for Computational Linguistics, (2011).

[39] Zhe Zhao, Paul Resnick, and Qiaozhu Mei, 'Enquiring minds: Early detection of rumors in social media from enquiry posts', in *Proceedings of the 24th International Conference on World Wide Web*, pp. 1395–1405. International World Wide Web Conferences Steering Committee, (2015).

[40] Max Zimmermann, Irene Ntoutsi, Zaigham Faraz Siddiqui, Myra Spiliopoulou, and Hans-Peter Kriegel, 'Discovering global and local bursts in a stream of news', in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pp. 807–812. ACM, (2012).