Semi-Supervised Event-related Tweet Identification with Dynamic Keyword Generation

Xin Zheng[1,2] Aixin Sun[1] Sibo Wang[3] Jialong Han[4]
[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[2]SAP Research and Innovation Singapore, SAP Asia Pte Ltd, Singapore
[3]School of Information Technology and Electrical Engineering, The University of Queensland, Australia
[4]Tencent AI Lab, Shenzhen, China
xzheng008@e.ntu.edu.sg;axsun@ntu.edu.sg;sibo.wang@uq.edu.au;jialonghan@gmail.com

ABSTRACT

Twitter provides us a convenient channel to get access to the immediate information about major events. However, it is challenging to acquire a clean and complete set of event-related data due to the characteristics of tweets, *e.g.,* short and noisy. In this paper, we propose a semi-supervised method to obtain high quality event-related tweets from Twitter stream, in terms of precision and recall. Specifically, candidate event-related tweets are selected based on a set of keywords. We propose to generate and update these keywords dynamically along the event development. To be included in this keyword set, words are evaluated based on single word properties, property based on co-occurred words, and changes of word importance over time. Our solution is capable of capturing keywords of emerging aspects or aspects with increasing importance along event evolvement. By leveraging keyword importance information and a few labeled tweets, we propose a semi-supervised expectation maximization process to identify event-related tweets. This process significantly reduces human effort in acquiring high quality tweets. Experiments on three real world datasets show that our solution outperforms state-of-the-art approaches by up to 10% in F_1 measure.

KEYWORDS

Dynamic Keyword Generation; Event-related Tweet Identification

1 INTRODUCTION

Social media platforms like Twitter provide us a fast and interactive channel to get access to the latest information. Given a major event happens, users are getting used to search for relevant information through social media. Keyword search is supported by most social media platforms for the convenience of getting desired information [7, 23]. However, it is often challenging for common users to formulate high quality queries. Irrelevant messages would mix with important ones, or some relevant aspects of an event are not retrieved at all [11, 30]. A clean and complete set of event-related data is preferred by both the public to learn about the event development and decision makers to make decisions as soon as possible. Methods to collect high quality event-related tweets are essential to satisfy users' information need.

Given an event of interest, described by a set of seed keywords, the goal of this research is to collect a set of relevant tweets from Twitter stream of high precision and recall. Note that our problem is different from event detection, because the event is already known here from the seed keywords. It is also different from event tracking since we do not have enough knowledge about the event but only a few words. To solve this problem, we define a time window and collect candidate tweets containing keywords computed in each of previous time window. Then an identifier is applied to determine which set of tweets are indeed relevant to the event of interest.

The proposed solution is relatively straightforward, but two challenges make it non-trivial. The first challenge lies in keyword generation. Well-formulated keywords are the key to collect a complete set of event-related tweets. Existing methods suggest dynamic keywords are preferred than static ones [7, 27]. The reasons are two-fold. Events develop over time, and each event has its own characteristics. Even a well-prepared set of static keywords with good domain knowledge could miss some event-specific points [18, 23]. On the other hand, it is challenging to produce a dynamic set of keywords along event evolvement and specifically focusing on one event. The second challenge exists in identification of event-related tweets. A straightforward method comes in mind is classification [18, 28]. However, we rarely obtain enough training data. Even labeled data for some similar events exist, *e.g.,* earthquake, it is hard to make it adapted to the event of interest. Researchers also try to apply topic model [26, 30], but the models hardly well handle all the aspects of the event of interest. Moreover, post classification processing is often required. To construct a high quality identifier, we believe human labels are necessary as input. The challenge is how to minimize human effort without compromising data quality.

To address the first challenge, we observe some desirable properties of event representative words. Firstly, words should be relevant to the event of interest and should have high coverage on event-related tweets. Besides, words always co-occurring with event-related words could also be relevant, which depicts words co-occurrence property. Some representative words are always with high values for the above properties. But these words are not preferred as newly generated keywords because they are already in the keyword set. Newly appearing event-related words and words with increasing importance to the event of interest could help cover more relevant tweets. This point has not been tackled in existing

CIKM'17, November 6–10, 2017, Singapore.

ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: https://doi.org/10.1145/3132847.3132968

works. Based on these observations, we define a word's importance score to measure the single word properties and word co-occurrence property. A configuration similarity is proposed to measure the dynamic changes of word properties. Top-ranked words according to this measure would be used to query new tweets.

To address the second challenge, instead of adopting a pure classification based method [18] or statistical probability based method [26], we take advantage of word importance information to help identify event-related tweets. The rich keyword information help us make the identification process to be semi-supervised through expectation and maximization mechanism, which reduces much human labeling effort to achieve similar results as in supervised manner. Existing methods which leverage word-level information to help decide sentence-level label are mostly based on language model and topic model [30]. However, these methods either require sufficient training data, or only present statistical information of given data making them difficult to produce desirable results. Our method utilizes word importance information and a few labeled tweets to overcome the drawbacks. Our contributions are summarized as follows:

- We propose a novel dynamic keyword generation mechanism based on word importance score and its changes overtime. Several word properties are explored to calculate the importance score. The changes of word importance measured by configuration similarity gives better description of an event's emerging or trending aspects.
- We devise an effective semi-supervised solution based on expectation and maximization mechanism. It leverages word information to infer tweet labels. Thus it can reduce human label efforts, without compromising the quality of the desired result.
- We use real-world data to evaluate our solution. Experiments demonstrate that our solution outperforms state-of-the-art approaches by up to 10% in terms of F_1 measure.

2 RELATED WORK

2.1 Methods for Query Keyword Generation

Keyword generation is relatively a well-studied problem. Many existing studies provide us hints in generating event-related query keywords. As a classic keyword generation method, Mihalcea and Tarau [19] apply PageRank [21] to word graph and rank words by weight. Liu *et al.* [17] build a Topical PageRank to decompose traditional random walks specific to various topics. There are also studies leveraging topic model to extract keywords automatically. Chen *et al.* [8] propose a topic model which exploits lexical relations of words in a given dictionary to extract coherent keywords for topics. Except extracting keywords about all topics, Wang *et al.* [26] propose a targeted topic model, a variation of LDA, to identify keywords on specific aspects. For example, there are many subtopics under camera domain, and users may only concern about the weight or lens. Given a few words which define the a particular aspect, their model returns keywords related to this aspect. Thus, they provide a way to focus on keywords of a targeted subtopic. We also evaluate this method in our experiments.

The aforementioned methods all generate keywords from a static collection. Before we have the relevant data, one may leverage the predefined keyword set to collect desired data from data stream [23]. Then an automatic way of generating keywords that are related to an ongoing event is much desired. Chen *et al.* [7] filter organization-related tweets through a set of fixed keywords and also keywords generated dynamically. The fixed keyword set is generated based on domain knowledge. The dynamic keywords are identified by using chi-square test [14] on word frequencies of the predefined foreground and background tweets. Becker *et al.* [1] leverage term frequency analysis and event-related concept extraction from external sources to generate dynamic keyphrases. Li *et al.* [16] adopt Rocchio query expansion [5] to dynamically change query keyphrases. Wang *et al.* [27] propose a double ranking approach to select a set of keywords which could be used to collect tweets satisfying a given topic or a research problem. This method relies on human input to judge the quality of keywords and assumes the dataset is static. Sadri *et al.* [22] map query phrases generation process to a Multi-armed Bandit problem [29] to maximize query coverage. The existing methods do not consider what kind of words could introduce more event-related tweets. We analyze the problem by considering the changes of keyword graph over time. In this study, we do not consider keyphrase extraction and we refer readers to a recent survey on keyphrase extraction methods [12].

2.2 Event-related Text Identification Method

Event-related text identifier determines the precision of the collected data. Classification and topic model based methods are widely adopted for this task. Becker *et al.* [2] leverage a supervised method to identify tweets as event-related or not. However, tweets related to any event are identified, not necessarily to a specific event. Magdy and Elsayed [18] adopt a classification method. They prepare a static set of well-defined queries about the topic of interest and assume tweets containing words in the query set as topic related. TF-IDF score is used to rank topic related keywords and filter potential topic relevant tweets from irrelevant ones. The trained classifier is used to infer labels for newly incoming data. A drawback of this solution is that the labels of training data is not confidently of good quality compared with human labels. To reduce human annotation efforts, Tong and Koller [24] apply active learning to text classification and analyze the effectiveness of different query strategies. An example of leveraging active learning to query needed data from search engine is [16]. The framework heavily relies on the ranked documents returned by the search engine. Besides classification, Sadri *et al.* [22] identify topic related tweets by measuring phrase-based relevance, textual relevance and user historical tweet relevance.

Topic model has also been applied to identify event-related data. Yang *et al.* [28] collect messages from social media related to adverse drug reaction by leveraging Latent Dirichlet allocation (LDA) and a partially supervised classification method. Wang *et al.* [26] propose a targeted topic model which leverages a latent variable to identify documents that belong to specific aspects. Although [13] studies novel topics detection, they claim that their dictionary learning based method could also identify topic relevant documents with high precision. Yan *et al.* [9] also adopt dictionary learning method to identify topic relevant documents. They build a topic model by combining term correlation matrix with non-negative matrix factorization. Non-negative matrix factorization is a variant of

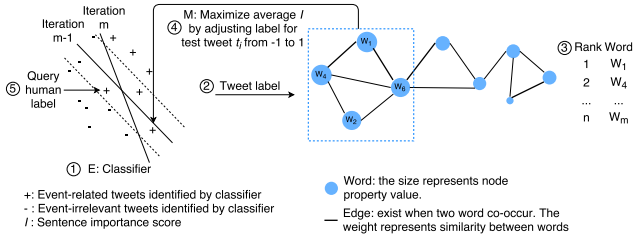


Figure 1: The information flow for one iteration is from Step 1 to 5. After getting new training data from Steps 4 and 5, a new iteration would start to retrain the classifier.

Dictionary Learning [13]. Bommannavar *et al.* [4] estimate the volume of tweets about a specific event or topic at a given precision level. This is achieved by Sequential Probability Ratio Test [25].

3 THE PROPOSED METHOD: SSEM

We define a time window and apply the proposed method on the tweets collected during each time window. Our solution starts with a set of seed keywords. Using these seed keywords, we collect a set of candidate tweets, from the tweet stream. Due to the retweet action on Twitter, many tweets are near duplicate or similar. We cluster similar tweets by incremental clustering [6]. Because our method requires human to label some data for the identifier, clustering similar tweets could reduces human effort. The clusters of tweets are then delivered to the proposed semi-supervised expectation maximization identifier, detailed in Section 3.1. The identifier leverages both human label and keyword information to infer whether a tweet is event-related. A keyword importance score is defined to quantify the keyword information, reported in Section 3.2. To make the expectation maximization identifier more efficient, constrains are proposed to reduce the solution space, discussed in Section 3.3. Next, we take changes of words and their neighbors into consideration and construct a keyword ranking function to select the preferred keywords for more event-related tweets collection. The details are presented in Section 3.4.

3.1 Expectation Maximization Identifier

The Expectation Maximization Identifier is illustrated in Figure 1. The process leverages both the tweet label information and the word importance information to identify event-related tweets.

Recall that we cluster similar tweets by incremental clustering. Afterwards, we randomly select two clusters to label: one is event-related, *i.e.,* positive, and the other is not, *i.e.,* negative. Many methods could ensure only 2 clusters are needed to check. One easy way is based on the keyword set. We select from tweet clusters that contain the most representative keywords in keyword set, and there is a great possibility for them to be positive. For the negative cluster, we select from the clusters that contain the least representative words in keyword set. The rest tweets are the testing data.

Expectation The two clusters of tweets are used as initial training data for SVM classifier. We could obtain current labels for testing data classified by the temporary classifier, and the inference function is:

$$\min_{\boldsymbol{w},b,\xi} \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C \sum_i \xi_i$$
$$\text{s.t.} \forall i : y_i(\langle \boldsymbol{w}, \boldsymbol{x_i} \rangle + b) \geq 1 - \xi_i, \xi_i \geq 0, y_i \in \{-1, 1\} \tag{1}$$

where $\boldsymbol{w}$ is the weight vector and b is a scalar value which would be learned. y_i is tweet label and x_i is tweet feature vector. Here we adopt simple bag-of-words weighted by TF-IDF as tweet representation vector. C and ξ_i are parameters learned during training. All the tweets in positive cluster are considered as event-related tweets Pos, and analogous to not event-related tweets Neg. Note that the labels are temporary and might change after each iteration. The tweets and their labels would be delivered to maximization stage.

Maximization We define a *word importance score* $fs(w)$ and a *sentence importance score* $I(t)$. The word importance score describes the representativeness of the word to the event of interest. The importance score is calculated based on single word properties and property of its co-occurrence words according to candidate tweet labels, to be detailed later. The sentence importance score describes the relatedness of the tweet to the event of interest. It is calculated by the multiplication of importance score for each word in the tweet. The rationale will be detailed in Section 3.2. Intuitively, tweets with high sentence importance score are more related to the event. Thus, we maximize the average sentence importance score for event-related tweets by adjusting the tweets labels which determines the word importance score:

$$\arg_{k_i} \max \frac{1}{|Pos|} \sum_{t_i \in Pos} I(t_i) - \beta \sum_{t_i \in S} (k_{t_i} - 1)^2$$
$$k_{t_i} = \begin{cases} -1, & \text{if } y_i \text{changes;} \\ 1, & \text{otherwise.} \end{cases} \tag{2}$$

where t_i is a tweet and β is a hyperparameter. k indicates whether the tweet label changes or not, and $\beta(k_i - 1)^2$ is a regularizer which penalizes too many tweets adjusting labels in case not event-related tweets dominate the sentence importance score. The details of the objective function will be presented next in Section 3.2. Note that only tweets belonging to a specific set S could adjust their labels, to be detailed in Section 3.3. The expectation and maximization process would iterate for a predefined number of times.

3.2 Word Importance Score

The sentence importance score is computed by the multiplication of the word importance scores. Importance score of a word is determined by single word properties and property by co-occurring words.

3.2.1 Single Word Properties. **Relevance** An event representative word should at least be relevant to the event. We adopt relative entropy to depict the relevance of each word to the event of interest:

$$r(w) = p(w, Pos) \cdot \log \frac{p(w, Pos)}{p(w, Neg)} = \frac{|Pos(w)|}{|Pos|} \cdot \log \frac{|Pos(w)||Neg|}{|Pos||Neg(w)|} \tag{3}$$

where $|Pos(w)|$ means the number of positive tweets that contain word w, and similarly $|Neg(w)|$. Here, the tweet label determines the word relevance score. The relevance value is scaled to $[0, 1]$.

Coverage Besides, we also prefer words with a relatively high coverage ratio, to avoid excluding event-related tweets. Thus the keyword set could maintain high recall for the event of interest. The coverage ratio is denoted by $\frac{|\mathcal{T}(w)|}{|Pos(w)|}$, where $|\mathcal{T}(w)|$ is the number of tweets that contain word w in the whole tweet stream within the current time window. However, it is not the higher coverage ratio the better. The irrelevant word could have a rather high coverage ratio as well. We prefer words with coverage ratio smaller than a threshold α to make sure most of the tweets containing the word w are positive. We assume this kind of words have higher probability to be event-related. Hence, we define the coverage score as follows which could scale the coverage ratio and penalize high coverage ratio:

$$c(w) = \begin{cases} e^{(x-\alpha)}, x = \frac{|\mathcal{T}(w)|}{|Pos(w)|} \leqslant \alpha \\ e^{(\alpha-x)}, x = \frac{|\mathcal{T}(w)|}{|Pos(w)|} > \alpha \end{cases} \tag{4}$$

The hyper-parameter α is empirically set.

Quality Score The quality of a word is determined by both relevance and coverage. We call it quality score and model it as the multiplication of relevance and coverage:

$$rs(w) = r(w) \cdot c(w) \tag{5}$$

3.2.2 Word Co-Occurrence Property. Words do not exist independently in text. Their appearances rely on the contextual information. Thus, the word importance level to the event of interest is also related to the relevance extent of the co-occurring words. We assume that words which always co-occur with high quality words are also of great possibility to be of high quality. On the opposite, words often co-occur with poor quality words would be relatively less important to the event. This is like that nodes in a graph are influenced by its neighbor nodes. Therefore, we model words interplay on a word co-occurrence graph.

Each word in a collected tweet set is a node in the graph and is associated with the word quality score as its node weight. An edge exists between two nodes when they co-occur in a tweet. The edge weight is measured by Pointwise Mutual Information (PMI), which depicts the association between the two words:

$$sim(w, v) = \log \frac{p(w, v)}{p(w)p(v)} = \log \frac{|t(w, v)|/|\mathcal{T}|}{|t(w)|/|\mathcal{T}| \cdot |t(v)|/|\mathcal{T}|} \tag{6}$$

where t is a tweet and $|t(w, v)|$ is the number of tweets that contain both words w and v. Figure 1 is an example of the constructed word co-occurrence graph with weighted nodes and weighted edges.

The word importance score is determined by both single word properties and the word co-occurrence property. And we evaluate the word importance score $fs(w)$ based on its relationship with one-hop neighbors $Nei(w)$ and their quality scores as follows:

$$fs(w) = rs(w) \cdot \frac{1}{|Nei(w)|} \sum_{v \in Nei(w)} sim(w, v) \cdot rs(v) \tag{7}$$

The average relationship with neighbor words is to estimate the average quality of neighbors and also to avoid common words with a large number of neighbors dominating the importance score.

3.2.3 Sentence Importance Score. Recall that all the candidate tweets contain at least one word in the event-related keyword set. However, not all of such tweets are related to the event because of the following two reasons:

- The keyword contained by the tweet may have multiple meanings and it happens to express the meaning that is irrelevant to the event of interest;
- Tweet is informal and short. Sometimes several uncorrelated things are compressed in one tweet simply to attract more attention.

As a result, we simply adopt the multiplication of word importance scores as sentence importance score I to estimate its coherence and relevance to the event of interest.

$$I(t_i) = \prod_{w_{i,j} \in t_i} fs(w_{i,j}) \tag{8}$$

Having all of the definitions of the basic components (*i.e.,* Equations 3, 4, 5, 7, 12), we can detail how to maximize Equation 2 by adjusting tweet labels:

$$\begin{aligned} r(w) &= \frac{\sum_i \frac{1+q_i}{2} 1(w_{i,j})}{\sum_i \frac{1+q_i}{2}} \log \frac{\sum_i \frac{1-q_i}{2} \sum_i (\frac{1+q_i}{2} 1(w_{i,j}))}{\sum_i \frac{1+q_i}{2} \sum_i (\frac{1-q_i}{2} 1(w_{i,j}))} \\ 1(w_{i,j}) &= \begin{cases} 1, \text{ if } w_j \in t_i \\ 0, \text{ if } w_j \notin t_i \end{cases} \\ c(w) &= \begin{cases} e^{(x-\alpha)}, x = \frac{|\mathcal{T}(w)|}{\sum_i \frac{1+q_i}{2} 1(w_{i,j})} \leqslant \alpha; \\ e^{(\alpha-x)}, x = \frac{|\mathcal{T}(w)|}{\sum_i \frac{1+q_i}{2} 1(w_{i,j})} > \alpha. \end{cases} \\ q_i &= y_i \cdot k_i \end{aligned} \tag{9}$$

where y_i is the label for tweet t_i assigned by the classifier and the changing of tweet label is indicated by an indicator variable k_i same as in Equation 2. Both the values of $r(w)$ and $c(w)$ are affected by the overall tweet labels.

3.3 Determination of Changeable Tweets

Making all available tweets labels to be changeable would lead to high computational cost. Therefore, we only allow labels change for tweets that are most possible to increase sentence importance score. Either changing labels from negative to positive or the opposite could make Equation 2 increases. Consider a situation that a tweet is indeed related to the event and is labeled as positive by the temporary classifier, while its sentence importance score is lower than the average value. This would happen when the classifier is of low recall during the iteration process. At this time, by changing the label from positive to negative might lead to higher value for Equation 2. However, by doing so, it would generate the wrong label. Therefore, it is not that confident to change labels from positive to negative. On the contrary, if we change a tweet label from negative to positive and it leads to higher objective value, then the tweet sentence importance score must be above the average value. This means the tweet is coherent and the words are important to the event. Therefore, it is confident to tell that the tweet is related to the event of interest based on the current information. Thus, the negative tweets predicted by the current classifier are changeable. Even so, the changeable tweets are still of a large number. Not

all negative tweets deserve the check and we can further reduce the unnecessary traverse. Among *Neg*, tweets that are closer to the hyperplane of SVM [15] and contain top ranked keywords by quality score rs are more likely to be positive and would maximize the objective function in Equation 2. Recall that before the whole process, we cluster similar tweets together. Therefore, we consider all the tweets in one cluster to be changeable that satisfy the following conditions:

- All the tweets in the cluster are classified as negative by the classifier;
- At least one of the tweets in the cluster contains top ranked keywords according to rs;
- The average distance for the cluster of tweets is among top H closest to the hyperplane of the SVM classifier.

The selected cluster(s) of tweets which could maximize the objective function would change labels from negative to positive, and be added into the training data.

Note that we add positive clusters of tweets into the training set from maximization step. To avoid imbalanced training data and also maintain a high precision, we ask human to label the same number of cluster(s) of tweets. If no cluster changes labels in current iteration, we query one cluster for human label. The top cluster(s) with average distance closest to the hyperplane of SVM is selected as query. This is according to uncertainty sampling [15] which is a commonly adopted query strategy in active learning to reduce human effort.

3.4 Dynamic Keyword Graph

Event develops along with time and new aspects about the event would appear. Therefore, we would like the query keywords to capture the newly appearing aspects. We hope the top ranked keywords could have some differences from the previous ones, so that the keyword set could be updated effectively. Specifically, we prefer keywords over time with the following properties:

- Words are newly appearing and with high importance scores;
- Words already appeared in the past time window but with increasing importance scores and better quality of neighbors.

Definitely, the newly appearing words having high importance scores with respect to the event of interest, would represent new aspects. Such words should be added into the keyword set. There is another kind of words which appeared before but now has higher importance score. This indicates the aspect depicted by these words attract more attention. However, we do not prefer words which are with high importance scores but have similar neighborhood relationships as those in the past time window. Such words appear in similar sentence environment along with time and would not help to cover more event-related tweets.

Therefore, we need to measure a word's neighborhood structure similarity along with time and we call it word *configuration similarity*. Eger and Mehler [10] provide us a method to evaluate the semantic changes. At each time window, the similarities between each word and its neighbor words construct a similarity vector. The similarity between the similarity vectors at different time windows reflects the word semantic changes over time.

In our problem setting, the word graph changes over time as illustrated in Figure 2:

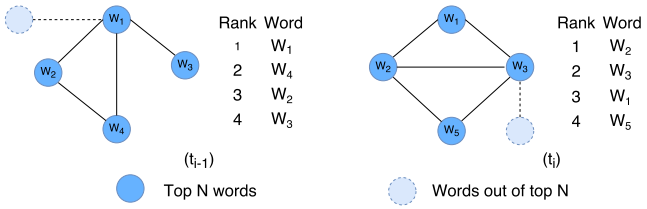


Figure 2: An example of word graph changes over time.

Table 1: Example of word's configuration similarity. $sim(w,v)$ and $sim(w,k)$ are calculated according to Equation 6. If there is no edge between word w and k, the edge weight $sim(w,k)=0$. m is the number of words in keyword set.

Feature	k_1	k_2	$\cdots$	k_m	Ratio
$s(w)_{t-1}$	$sim(w,k_1)_{t-1}$	$sim(w,k_2)_{t-1}$	$\cdots$	$sim(w,k_m)_{t-1}$	r_{t-1}
$s(w)_t$	$sim(w,k_1)_t$	$sim(w,k_2)_t$	$\cdots$	$sim(w,k_m)_t$	r_t

- Words appear and disappear;
- Edges appear and disappear;
- Edge weight changes between neighbor words;
- Quality score changes for single word.

Thus, we can not directly apply Eger and Mehler's [10] method because they assume the words' neighbors are static. In fact, only top ranked words by word importance score fs are in consideration of the dynamic change, *i.e.,* top N, which are the core candidate representative keywords for the event of interest. To make the similarity vector comparable over two contiguous time windows, we fix a set of neighbors for each word, *e.g.,* the keyword set. Besides the set of fixed neighbors, there are other neighbors for each word as well. To measure the co-occurred extent of the fixed set of neighbors over all the neighbors, we add the ratio $\frac{\sum_{k\in KeywordSet} sim(w,k)}{\sum_{v\in Nei(w)} sim(w,v)}$ in the similarity vector, illustrated in the last column of Table 1.

According to Table 1, the configuration similarity could be calculated by cosine similarity of the two similarity vectors:

$$sim(w_{t-1}, w_t) = cosine(s(w)_{t-1}, s(w)_t). \tag{10}$$

For newly appearing words, the configuration similarity value is 0.

Based on the preference of keywords selection, we propose a re-rank function for top ranked words by word importance score fs, to take word changes over time into consideration:

$$rank_d(w) = (1 - sim(w_{t-1}, w_t)) \cdot \frac{N - rank_s(w)_t}{N - rank_s(w)_{t-1}} \tag{11}$$

where $rank_s$ is the ranking position by fs and $(1 - sim(w_{t-1}, w_t))$ indicates the preference of words with dissimilar neighbors. However, this could also be the word that becomes less important than before. Thus the ranking position comparison $\frac{N - rank_s(w)_t}{N - rank_s(w)_{t-1}}$ is adopted to avoid such situation.

At the first time window, words that are added into keyword set are selected by word importance score fs. From the second time window onwards, top R words ranked by $rank_d(w)$ are added in keyword set.

Overall, our method is based on the expectation and maximization framework. In expectation step, we leverage SVM classifier to predict the tweets labels. We adopt the simple linear kernel SVM which has time complexity $O(n^2)$, where n is the number of training samples. In maximization step, we first calculate single word properties and words co-occurrence property separately. The time complexity is $O(w)$ and $O(w^2)$ respectively, where w is the vocabulary size. Then, we need to maximize the objective function, *i.e.,* Equation 2, by changing tweets labels. The single word properties and words co-occurrence property are needed to re-calculate at each round of tweet label changing. As the result, the time complexity is $2^H * O(w^2)$, where H is the number of changeable tweets. Since we have constraints to set the changeable tweets H to be a small number, the number 2^H can be considered as a constant a. Eventually, the time complexity of maximization step is $a * O(w^2)$. The expectation and maximization process iterates for several rounds, denoted by b, until relatively stable results are obtained. The time complexity for the whole process becomes $b * a * O(w^2)$. Experiments suggest that the number of iterations b is smaller than 25. The re-ranking process only takes linear time complexity. In short, the proposed method is efficient.

4 EXPERIMENT

4.1 Data

In our experiments, we simulate Twitter stream with a large static dataset collected by the Internet Archive.[1] The tweets were collected by Spritzer[2], which is a Twitter Stream sampling API and returns around 1% of the full Twitter stream. We conduct experiments on three real world events.

- *Chile Earthquake*: A magnitude 8.2 earthquake struck offshore of Iquique, Chile on 1 April 2014 at a depth of 20.1 km (12.5 mi);
- *Indian Flood*: The 2015 South Indian flood resulted from heavy rainfall generated by the annual northeast monsoon from November to December, 2015.
- *NBA All-Star Game*: The 2016 NBA All-Star Game was an exhibition basketball game played on February 14, 2016.

We use *EQ*, *Flood* and *NBA* to denote the three events. Tweets are collected from the first two days of *EQ*. For the *Flood* event, we take the data of the three most severe days because the rainfall takes some time to become a flood. For the *NBA* event, we collect data from Feb 14th to 16th.

We split all tweets in the Twitter stream into different time windows because of the different characteristics of the three events. Statistics are reported in Table 2. Specifically, we set the time window size for *EQ* to be 8 hours, and 12 hours for *NBA*. For *Flood* event, the event-related tweets on the collected third day becomes fewer than 100, which may be caused by the fact that only around 1% of real data are sampled by Spritzer. Due to the small number of tweets on this event, we take one day as the time window size for *Flood* data. Observe from Table 2 that the number of positive tweets in each time window is extremely small compared with that in Twitter stream.

[1] https://archive.org/details/twitterstream
[2] https://dev.twitter.com/streaming/overview

Table 2: Number of tweets $\mathcal{T}$ fall within the time window in the simulated Twitter stream (*i.e.,* 1% sample of the entire Twitter), and number of positive tweets $\mathcal{T}_\mathcal{P}$ in each time window on the three datasets.

EQ	$\mathcal{T}$	$\mathcal{T}_\mathcal{P}$	Flood $\mathcal{T}$	$\mathcal{T}_\mathcal{P}$	NBA $\mathcal{T}$	$\mathcal{T}_\mathcal{P}$
t_1	79,726	280	day_1 1,613,579	193	t_1 162,599	345
t_2	244,529	513			t_2 719,089	1,894
t_3	364,964	373	day_2 1,618,825	236	t_3 538,070	2,088
t_4	499,713	1,153			t_4 720,170	2,517
t_5	246,540	280	day_3 1,783,940	116	t_5 678,860	966
t_6	342,098	168			t_6 709,626	756

During each time window, we collect tweets by keyword set K. As long as a tweet contains one of the keywords in K either as a word or as a substring, it would be collected. For example, given a keyword "Chile", tweets containing the hashtag #chileearthquake will be collected as well.

For evaluation purpose, we need to construct the ground truth. We define event-related tweets for *EQ* and *Flood* according to [20], which provides several most popular aspects of a disaster queried by users. For *NBA*, we determine the event relevant tweets as long as the content is related to NBA 2016 but not focus on specific aspects. Since the entire stream dataset is too large for manual annotation, we choose to label two subsets of tweets within each time window for evaluation purpose as follows:

- Tweets that are found to be event-related by the proposed method and any of the baseline methods in each time window;
- Randomly collect 1,000 tweets without any words in keyword set from the Twitter Stream in each time window.

The above two sets of tweets together form the ground truth collection. In this sense, the recall scores reported in our experiments are limited by these two sets. Nevertheless, it is infeasible to get the true recall due to the large number of tweets.

The tweets collected through keyword search in each time window will be cleaned as follows. First, non-English tweets are removed by a language detection tool.[3] Afterwards, URLs, punctuation marks (except @ and # which denote accounts and hashtags), special symbols, and stop words are removed. The proposed method and baseline methods are applied on the cleaned tweets.

4.2 Initial Seed Keyword Generation

A comprehensive set of seed keywords is essential for the high-recall requirement in tweets collection. Here, we leverage knowledge from Wikipedia to construct a set of seed keywords. Note that the seed keywords could be generated from any domain-specific documents or even given by domain experts. We adopt Wikipedia to avoid the bias of personal knowledge.

Consider Chile earthquake as the event of interest. We consider all Wikipedia entries with the word *earthquake* appearing in their titles as earthquake-related. After collecting all such entries, we perform Part-of-Speech (POS) tagging on the content of these documents and select nouns as candidate event-related keywords,

[3] https://github.com/optimaize/language-detector

assuming that keywords are nouns. To find out which words are most related to the target event *earthquake*, we adopt Mutual Information (MI) as a measure of the association strength between the word *earthquake* and each of the nouns in the candidate set, shown in Equation 12.

$$I(e; w) = p(e, w) \log \frac{p(e, w)}{p(e)p(w)} \tag{12}$$

In this equation, e denotes the word *earthquake*, $w \in C$ denotes a word w (or a noun) from the candidate set C. The top-ranked M words are selected and added into the K (in our experiments, $M = 10$). Besides, we also add the event name and location into K, *e.g.,* "earthquake" and "Chile". With these initial seed keywords, we use Twitter Stream API to collect all tweets that contain at least one word in K. We denote the set of collected tweets as candidate tweet set TC, $TC = \bigcup_{w \in K} T(w)$, where $T(w)$ is the set of tweets containing keyword w. The same process is used to obtain the initial seed keywords for events *Flood* and *NBA*.

4.3 Baselines

As reviewed in Section 2.2, the most commonly adopted methods for event-related text identification are classification and topic model. Meanwhile, topic model can also be applied for keyword extraction. Thus, we compare the proposed method *SSEM* with five state-of-the-art methods: ADP [18], Latent Dirichlet Allocation (LDA) [3], Dictionary Learning (DL) [13], Targeted Topic Modeling (TTM) [26], and Active Learning (AL). Except *ADP*, all the other baseline methods are applied on the cleaned tweets in the keyword filtered candidate tweet set. We will detail the processing for *ADP* shortly.

The *LDA*, *DL*, and *TTM* methods all need a topic number to be assigned, and would return several clusters of tweets and corresponding keywords under different topics. We take the top-ranked keywords of the most relevant topic(s) from *LDA* and *DL*, and top-ranked keywords from every topic of *TTM* are all considered relevant to the specified events. Then we add them into the keyword set K to collect tweets in the next time window. The number of selected keywords from the chosen topic in each time window is the same as the number of keywords extracted by our method. If more than one cluster is selected as event-related in these baseline methods, *e.g.,* 2, then top $R/2$ keywords from each cluster will be selected as keywords.

LDA We consider each tweet as an input document for *LDA*, and the same for *DL* and *TTM*. We empirically evaluate various topic numbers and select the one with the best topic grouping results. For each topic number, among the resulting topics, we select the candidate relevant topic(s) according to top-ranked keywords by manual checking. The manual checking process is the same for *DL*.

DL The objective function in dictionary learning is as follows [13]:

$$l(X, D_T) = \min_{\Psi \in \mathbb{R}^{k \times m}} \frac{1}{2} \parallel X - D_T \Psi \parallel_2^2 + \lambda \parallel \Psi \parallel_1$$

where X is a word × document matrix, D_T is considered as a word × topic matrix, and Ψ is regarded as a topic × document matrix. We assign each word to the topic where the value of the row in D_T is the largest. The assignment of documents to topics is the same. When selecting top-ranked words, we rank all the words in one topic according to their corresponding values in D_T.

TTM This method requires documents of one domain but with different aspects as input [26]. Because we can not assign all the tweets into specific domains, we take the tweets selected by keywords as one domain. The aspect-specified keywords also need to be predefined. Note that, the model does not require comprehensive keywords to describe the target aspect. We select the most representative words as aspect-specified keywords.[4] Experiments in the paper [26] show the quality of keywords, but no result of the aspect-related documents is reported. In the model, there is a status variable $r \in \{0, 1\}$ which indicates the document's relevance to the targeted aspect, where $r = 1$ means the document is relevant to the aspect and $r = 0$ otherwise. Thus we identify event-related documents by variable r.

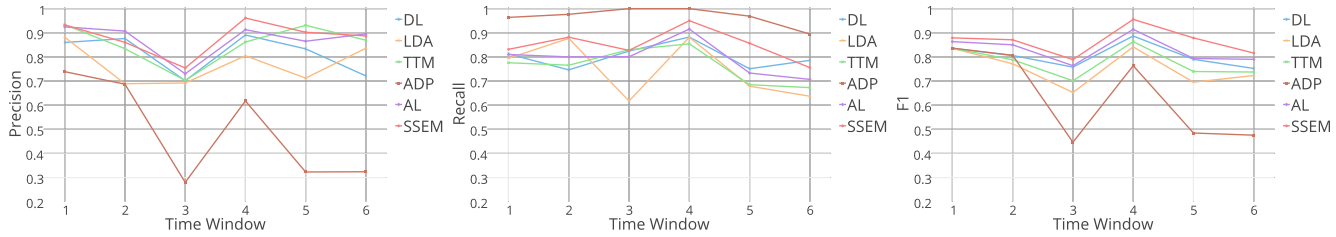
ADP We follow the setting in [18]. A well-defined set of queries Q on the given event is used to capture the static part of the event. Tweets containing words from the set of queries are considered as positive training data P_t. Negative training tweets N_t are randomly selected from the rest of tweets. A TF-IDF score is adopted to rank words in P and top x are selected as event-representative keywords E. Tweets in N_t containing E are removed. A classifier trained by P_t and N_t is used to classify tweets in the next time window, and the classifier is periodically retrained by new tweets. Note that the query keyword set Q is not updated along time.[5] We consider tweets extracted in the first 6 hours of each time window for the *EQ* event as training data candidates, and tweets in the remaining 2 hours for testing. For the *NBA* event, tweets from the first 9 hours of each time window are used as the training data, and those from the remaining 3 hours are used for testing. Since the event-related tweets for the *Flood* event are extremely unevenly distributed over time, we divide each time window to extract training data from the time period covering the first 80% positive tweets. Tweets from the remaining time period in a time window are used as test data. This is simply to obtain a relatively good results.

Note that we do not set the query set of *TTM* and *ADP* to be the same as the initial seed keyword set, because poorer results are observed from these two methods using seed keywords. The reason could be noises introduced by the general words like *people, area, time*. The presented query sets for these two baseline methods are those leading to the best performance after an extensive tuning the query sets for these two methods.

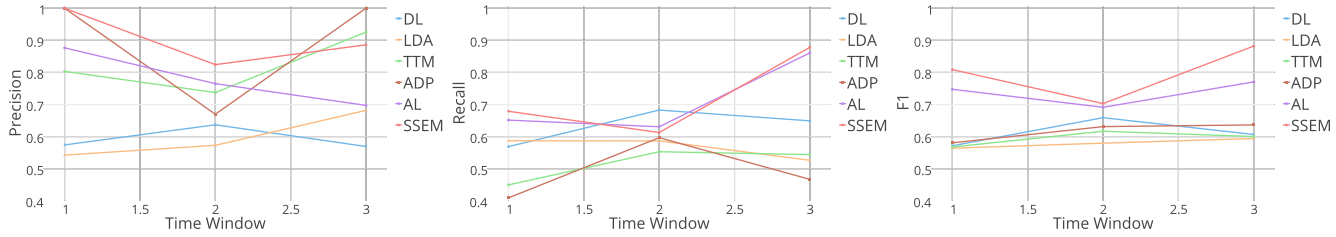
AL This is a variation of our proposed method, which does not include the label inference step from keywords. Thus this is a supervised setting. We adopt SVM as the classifier and uncertainty sampling as the querying strategy. The keyword generation method is the same as in *SSEM*. To make a fair comparison, the input tweets are also clustered by incremental clustering and each time

[4] We set *earthquake, quake, tsunami, magnitude, Chile* for the *EQ* event, and *flood, flooding, floods, Indian* for the *Flood* event. Keywords for the *NBA* event is changed for each time window, and the whole set of best-performance keywords are *NBA, star, dunk, dunks, #nbaallstarto, @nbaallstar, kobe, steph, contest.* Each time a subset of these words would be used to train the model.

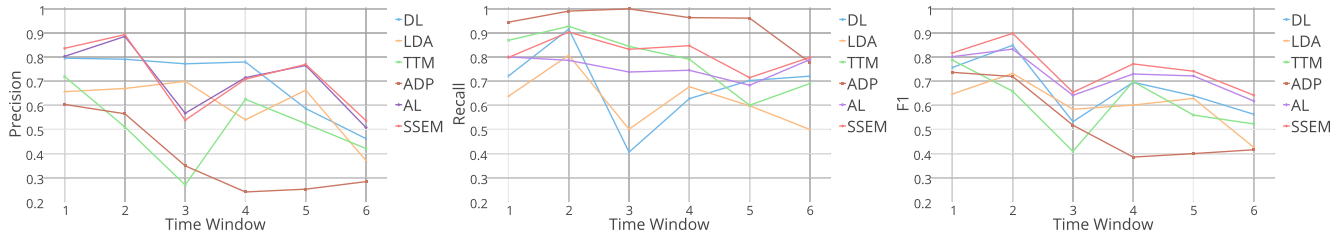
[5] We set the query set of *Chile, earthquake, quake, tsuman, magnitude* for the *EQ* event, and *flood, flooding, floods, Indian* for the *Flood* event. For the *NBA* event, the keyword set is *dunk, dunks, all-star, aaron, gordon, kobe, steph.* All of them give best experimental results.



(a) Precision, Recall and F_1 of 6 methods on EQ event



(b) Precision, Recall and F_1 of 6 methods on Flood event



(c) Precision, Recall and F_1 of 6 methods on NBA event

Figure 3: Performance comparison of the proposed method SSEM and baseline methods DL, LDA, TTM, ADP, and AL, on three events EQ, Flood and NBA.

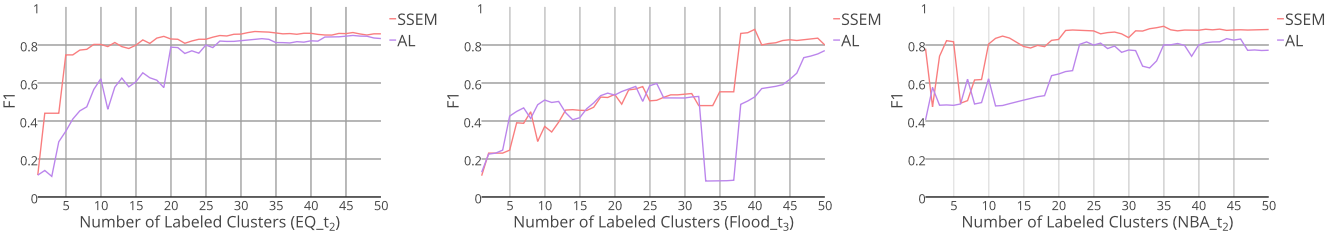


Figure 4: Changing of F_1 measures of AL and SSEM, with different number of human labeled clusters on the three events EQ, Flood and NBA.

the cluster with minimum average distance to the hyperplane is selected to ask for human labeling. The process stops after a predefined number of iterations.

4.4 Evaluation on Event-related Tweets

Figure 3 is the comparison results of *SSEM* and baseline models. Based on F_1 measure, our method outperforms all baselines on all the three datasets. With a deeper look at precision and recall, we observe that *SSEM* also achieves better results than other methods most of the time. The *AL* model achieves the second best F_1 result, and *SSEM* performs better than *AL* all the time on the three datasets. This result suggests that the tweet label inference from keywords, *i.e.,* the maximization step, does help to improve the model performance. Looking at precision and recall, we observe that *SSEM* outperforms *AL* most of time as well. The third best method is *DL* by F_1 measure, and it is better than *TTM* and *LDA*.

Table 3: Example keywords extracted on EQ in time t_3. Keywords in bold are NOT related to the event of interest.

Seed		earthquake, chile, damage, magnitude, people, area, epicenter, building, time, fault, region, utc
DL		magnitude(67), panama(41), earthquake(200), hits(28), 6.2(20), chile(258), 8.2(57), tsunami(38), coast(46), quake(56), **download**(11), prison(15), female(17), #prayforchile(62), @matthunter123 (15)
LDA		chile(163), earthquake(142), tsunami(76), quake(66), magnitude(67), hits(28), panama(41), coast(44), death(24), massive(22), hit(19),japan(20), female(18), **ice**(11), damage(20)
TTM		earthquake(214), chile(282), #prayforchile(37), quake(91), panama(37), hits(28), **focus**(16), tsunami(46), **today**(23), **following**(30), 6.2(20), coast(40), death(19), magnitude(64)
ADP		earthquake(125), chile(116), magnitude(60), 8.2(42), #prayforchile(19), tsunami(44), panama(32), coast(31), **following**(227), hits(116), 6.2(19), chilean(10), usgs(10), #earthquake(13), aftershocks(13)
AL		magnitude(71), #chileearthquake(23), quake(86), #chile(19), #tsunami(19), #earthquake(27), 8.2(65), earthquake(202), @reuters(13), @matthunter123(14), preliminary(6), panama(39), inmates(10), terremoto(6), #prayforchile(78)
SSEM	fs	@matthunter123(14), @reuters(13), **declined**(14), #prayforchile(78), #chileearthquake(23), preliminary(6), toll(15), panama(39), earthquake(202), chile(266), terremoto(6), inmates(10), **lindsay**(14), quake(86), 6.2(19)
	ReRank	m8.2(7), #chileearthquake(23), magnitude(71), quake(86), #earthquake(27), terremoto(6), #tsunami(19), @reuters(13), @matthunter123(14), preliminary(6), panama(39), inmates(10), 6.2(19), iquique(10), earthquake(202)

Table 4: Example keywords extracted on Flood in time day_2. Keywords in bold are NOT related to the event of interest.

Seed		rainfall, floods, people, damage, water, part, area, displace, land, river
DL		**#stopscientology**(50), november(57), december(25), #weather(23), flooding(52), **superb**(42), **craze**(42), flood(3), **great**(1), **partly**(1)
LDA		river(13), blocked(14), flood(23), **sam**(3), **shows**(3), **owblfn**(13), ahmedabad(13), **stopscientology**(23), **fatehbaad**(29)
TTM		**@karunainsan12**(18), flood(42), flooding(53), indian(30), flooded(26), haryana(13), **explanation**(14), **superb**(42), **near**(15), blocked(14)
ADP		**indianapolis**(24), **#othercityofficials**(33), **area**(408), **ikea**(83), **indiana**(26), flooded(24), **plans**(536), **store**(759), bloodflood(17), **ii**(301)
AL		#msgonthetopinharyana, **#stopscientology**, zeal(29), **#banscientology**(28), #floodthefeed(28), disaster(14), #infloodwetrust(11), flooding(53), flood(42), rain(13)
SSEM	fs	**#stopscientology**(28), boundaries(29), zeal(29), disaster(14), flooding(53), **marineland**(14), **#banscientology**(28), #infloodwetrust(11), **@voice_ot_orcas**(14), #msgonthetopinharyana(29)
	ReRank	#msgonthetopinharyana(29), **#stopscientology**(28), zeal(29), **#banscientology**(28), #floodthefeed(28), disaster(14), #infloodwetrust(11), rain(13), flooding(53), flood(42)

Table 5: Example keywords extracted on NBA in time t_4. Keywords in bold are NOT related to the event of interest.

Seed		NBA, all-star, basketball, game, MVP, Toronto, Raptors, celebrity, three-point, dunk
DL		back(390), aaron(109), @nba(349), kobe(824), nba(991), dunks(80), **jeans**(56), curry(109)
LDA		dunk(414), nba(656), double(6), lebron(19), curry(105), jordan(28), weekend(94), contest(248), points(77), aaron(108)
TTM		star(1344), dunk(554), game(733), kobe(496), nba(862), wars(206), contest(266), #nbaallstarto(511), best(184)
ADP		#contestentry(2), **@bigjuicyjack**(1), **@1035kissfm**(1), **#kissfmchicagolovestroye**(1), #contest(103), win(1019), lavine(5), best(1381), back(1264), slam(11)
AL		@ballislife(6), sabres(6), @cbssportsnba(6), tracy(37), dragonball(6), spurs(51), @double0ag(19), evander(15), retirement(25), @fullcourtprez(7)
SSEM	fs	kobe(590), @nba(349), star(650), nba(823), @bleacherreport(157), bryant(255), #nbaallstarto(511), #legend(15), tracy(37), sabres(6)
	ReRank	#legend(15), @ballislife(6), honors(35), #nbaallstarto(511), #dunkcontest(7), bryant(255), sabres(6), tracy(37), #zachlavine(6), @bleacherreport(157)

The performance of *ADP* is rather sensitive to the workload. This is because they consider all tweets containing the query keywords as positive, while this is not always true on the noisy data from Twitter. When the keyword set covers most of event-related tweets, recall would be very high while precision is low as shown on the *EQ* and *NBA* events in Figure 3. If the keyword set cannot cover all aspects of an event, the recall would be rather low but the precision is fairly high, as in the case of *Flood* event in Figure 3. Thus, we cannot directly consider tweets containing keywords as event-related. Human labels and other label inference processes (*i.e.,* label inference from keyword) are needed to acquire high quality data.

Figure 4 shows one example of human label comparison over the number of clusters on the three events.[6] Observe that, at almost all times, *SSEM* achieves stable results earlier than the *AL* method. In other words, our *SSEM* method requires fewer human labels than *AL* to achieve the same performance. Therefore, *SSEM*

[6]For space limitation, we cannot show the comparison results on all time windows.

demonstrates the capability of reducing human labels by leveraging keyword information.

4.5 Evaluation on Keywords Generation

Tables 3, 4, and 5 list example keywords of one time window over the three events. Note that in the first time window for all three events, the top-ranked keywords for *AL* and *SSEM* are ranked by the importance score fs (see Section 3.2.2 for the definition of fs), which is the single time window ranking. From the second time window on, the top ranked keywords are generated by considering keyword graph changes. Results from all three tables are generated by considering keywords' dynamic changes over time. Some topic-model based methods are with fewer keywords because they generate the same keywords in different topics.

From the results, some important keywords appear all the time along the development of the event, such as *earthquake, chile* for the *EQ* event, *flood, flooding* for the *Flood* event, and *NBA, dunk* for the *NBA* event. This kind of keywords could be produced by all baseline methods and ranked higher than other words. However, generating such kind of keywords all the time does not help introduce new aspects of events. To make the newly added keywords effective on collecting event-related data, our keyword generation method tries to rank such kind of words lower. Emerging words are given higher ranks. Results from the three tables demonstrate the desired results, *e.g., terremoto, @reuters* for *EQ*, *disaster, #infloodwetrust* for *Flood*, and *#dunkcontest, #legend* for *NBA* are extracted and rank higher. That is the proposed keyword generation method could produce keywords representing the emerging aspects of an event and also the important user accounts, *e.g., @reuters, @bleacherreport*, which is also a desirable feature for depicting a developing event. However, the other methods could not produce such useful keywords and would generate some irrelevant words.

The fs and *ReRank* in Tables 3, 4 and 5 give a comparison example of word ranking by the importance score fs and re-ranking results by considering keywords' dynamic changes. Observe that, before considering word changes over time, top ranked words by importance score fs include those core keywords of each event or words that have already been added before, *e.g., #prayforchile, NBA*. After considering the keyword graph changes over time, more relevant keywords are introduced, *e.g., #tsunami, iquique, #dunkcontest*.

5 CONCLUSION

We propose a semi-supervised expectation and maximization mechanism to collect high-quality tweets related to an event of interest from the Twitter stream. We make the first attempt to leverage word properties to help identify event-related tweets, which could reduce human annotation effort and maintain high performance in terms of F_1 measure. By considering the word graph changing over time, we can generate keywords with few overlap with historical ones. Existing works do not consider words' dynamic changes. Since keywords that could depict the emerging aspects of events are preferred, one can also refer to the top-ranked keywords to learn about the event development. Our proposed method outperforms state-of-the-art methods on both event-related tweets identification and keywords generation.

ACKNOWLEDGMENTS

The first author is in the SAP Industrial Ph.D Program, partially funded by the Economic Development Board and the National Research Foundation of Singapore.

REFERENCES

[1] Hila Becker, Dan Iter, Mor Naaman, and Luis Gravano. 2012. Identifying content for planned events across social media sites. In *WSDM*. 533–542.
[2] Hila Becker, Mor Naaman, and Luis Gravano. 2011. Beyond Trending Topics: Real-World Event Identification on Twitter. In *ICWSM*.
[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3 (2003), 993–1022.
[4] Praveen Bommannavar, Jimmy J. Lin, and Anand Rajaraman. 2016. Estimating topical volume in social media streams. In *SAC*. 1096–1101.
[5] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. 1995. Automatic query expansion using SMART: TREC 3. *NIST special publication sp* (1995), 69–69.
[6] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. 2004. Incremental Clustering and Dynamic Information Retrieval. *SIAM J. Comput.* 33, 6 (2004), 1417–1440.
[7] Yan Chen, Hadi Amiri, Zhoujun Li, and Tat-Seng Chua. 2013. Emerging topic detection for organizations from microblogs. In *SIGIR*. 43–52.
[8] Zhiyuan Chen, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malú Castellanos, and Riddhiman Ghosh. 2013. Discovering coherent topics using general knowledge. In *CIKM*. 209–218.
[9] Xueqi Cheng, Jiafeng Guo, Shenghua Liu, Yanfeng Wang, and Xiaohui Yan. 2013. Learning Topics in Short Texts by Non-negative Matrix Factorization on Term Correlation Matrix. In *SDM*. 749–757.
[10] Steffen Eger and Alexander Mehler. 2016. On the Linearity of Semantic Change: Investigating Meaning Variation via Dynamic Graph Models. In *ACL(2)*.
[11] Feifan Fan, Yansong Feng, Lili Yao, and Dongyan Zhao. 2016. Adaptive Evolutionary Filtering in Real-Time Twitter Stream. In *CIKM*. 1079–1088.
[12] Kazi Saidul Hasan and Vincent Ng. 2014. Automatic Keyphrase Extraction: A Survey of the State of the Art.. In *ACL (1)*. 1262–1273.
[13] Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. 2011. Emerging topic detection using dictionary learning. In *CIKM*. 745–754.
[14] Henry Oliver Lancaster and Eugene Seneta. 2005. *Chi-Square Distribution*. Wiley Online Library.
[15] David D. Lewis and Jason Catlett. 1994. Heterogeneous Uncertainty Sampling for Supervised Learning. In *ICML*. Morgan Kaufmann, 148–156.
[16] Cheng Li, Yue Wang, Paul Resnick, and Qiaozhu Mei. 2014. ReQ-ReC: high recall retrieval with query pooling and interactive classification. In *SIGIR*. 163–172.
[17] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic Keyphrase Extraction via Topic Decomposition. In *EMNLP*. 366–376.
[18] Walid Magdy and Tamer Elsayed. 2014. Adaptive Method for Following Dynamic Topics on Twitter. In *ICWSM*.
[19] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into texts. In *EMNLP*, Vol. 4. 404–411.
[20] Alexandra Olteanu, Sarah Vieweg, and Carlos Castillo. 2015. What to Expect When the Unexpected Happens: Social Media Communications Across Crises. In *CSCW*. 994–1009.
[21] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: bringing order to the web. (1999).
[22] Mehdi Sadri, Sharad Mehrotra, and Yaming Yu. 2016. Online Adaptive Topic Focused Tweet Acquisition. In *CIKM*. 2353–2358.
[23] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *WWW*. 851–860.
[24] Simon Tong and Daphne Koller. 2002. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research* 2 (2002), 45–66.
[25] Abraham Wald. 1945. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics* 16, 2 (1945), 117–186.
[26] Shuai Wang, Zhiyuan Chen, Geli Fei, Bing Liu, and Sherry Emery. 2016. Targeted Topic Modeling for Focused Analysis. In *SIGKDD*. 1235–1244.
[27] Shuai Wang, Zhiyuan Chen, Bing Liu, and Sherry Emery. 2016. Identifying Search Keywords for Finding Relevant Social Media Posts. In *AAAI*. 3052–3058.
[28] Ming Yang, Melody Y. Kiang, and Wei Shang. 2015. Filtering big data from social media - Building an early warning system for adverse drug reactions. *Journal of Biomedical Informatics* 54 (2015), 230–240.
[29] Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*. 1201–1208.
[30] Deyu Zhou, Liangyu Chen, and Yulan He. 2015. An Unsupervised Framework of Exploring Events on Twitter: Filtering, Extraction and Categorization. In *AAAI*. 2468–2475.