Migrating Social Event Recommendation Over Microblogs

Xiangmin Zhou
School of Computing Technologies, RMIT University
Melbourne, Australia
xiangmin.zhou@rmit.edu.au

Lei Chen
Hong Kong University of Science and Technology
Hong Kong, China
leichen@cse.ust.hk

ABSTRACT

Real applications like crisis management require the real time awareness of critical situations. However, the services using traditional methods like phone calls can be easily delayed due to busy lines, transfer delays or limited communication ability in disaster areas. Existing social event analysis solutions enhanced the situation awareness of systems. Unfortunately, they cannot recognize the complex migrating social events that are first observed in social media at a specific time, place and state, but have further moved in space and time, which may affect the system comprehension. While the discussion on events appears in microblogs, their movement over different contexts is unavoidable. So far, the problem of migrating social event analysis from big media is not well investigated yet. To address this issue, we propose a novel framework to monitor and deliver the migrating events in big social media data, which fully exploits the social media information over multiple attributes and their inherent interactions among events. Specifically, we first propose a Concept TF/IDF model to capture the content that is constrained by the time and location of media without costly learning process. Then, we construct a novel Maximal User Influence Graph (MUIG) to extract the social interactions. With MUIG, the event migrations over space and time are well identified. Finally, we design efficient query strategies over Apache Spark for recommending events in real time. Extensive tests over big media are conducted to prove the high effectiveness and efficiency of our approach.

PVLDB Reference Format:
Xiangmin Zhou and Lei Chen. Migrating Social Event Recommendation Over Microblogs. PVLDB, 15(11): 3213-3225, 2022.
doi:10.14778/3551793.3551864

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/xiangminemilyzhouAu/MEIR.git.

1 INTRODUCTION

The popularity of microblogging services has provided a vital source for online reporting real-world events. These social events may hold materials on the critical situations during disasters, but invisible to crisis coordinators or users. Being aware of these situations helps watcher officers to make right decision rapidly, reducing the injuries, life and economic loss . In real applications, social events may involve complex migrations, especially when natural disasters such as COVID-19 and Nepal earthquake 2015 happened. For example, during Nepal earthquake 2015, the east of Gorkha District at Barpak, Gorkha saw epicenter on 25 April, while continued aftershocks occurred throughout Nepal at the intervals of 15–20 minutes. The Chinese border between the capital of Kathmandu and Mount Everest became the epicenter of a major aftershock on 12 May. The donation for this disaster was made in countries over the world including India and US etc. During Texas flood 2015, flood warnings were issued for the counties in South East Texas on 14 May. On May 23, Oklahoma saw heavy flash flooding. On May 26, a flood emergency was issued for southwest Harris County and northeast Fort Bend County. In COVID-19, infected cases appeared in Wuhan Hubei province of China first, then spread to other provinces in China and further to other countries with the travelling of infected people. Accurately finding the event migration and recommending it to relevant users on time greatly helps people in crisis.

Making recommendations on events to users is an effective way of increasing their engagement in critical situations. In security, terrorist attacks from the same criminal groups may happen at various locations over a time period [5]. Recommending these events to proper users helps security offers to protect people and identify criminals collaboratively. In e-business, sales promotion for a new product may be carried out in different time and locations [2]. Recommending these events to interested users helps online merchants accelerate the purchase activities. For transportation, traffic issues can happen at multiple locations in a time period [6]. Awareness and recommendation of them helps people make better travel route on time. We study effective solutions for real-time migrating event recommendation over microblogs. For social event recommendation, there are still several challenges in critical situation awareness and natural disaster scenarios due to the special characteristics of social media and crisis events in contrast to general planed events.

- Uncertainty: social media consists of uncertain content, posting time, user location. As media are generated by worldwide users without supervision, the ambiguous texts like word variations, abbreviations or synonyms can be easily introduced. Due to the posting delay and user movement, the contexts appear to be uncertain as well.
- Dynamic: migrating events are not static but evolve over space and time. Due to the evolution of events or event subject movement, migrating events may involve consecutive and non-consecutive space and time changes.
- Big data: social media flow in huge volume and velocity.

To conduct effective and efficient migrating event recommendation, we need to well address three key issues. First, we need to construct a robust model that well handles the uncertainty in social content and contexts, and captures the migration of social events over different contextual attributes. Social messages contain

Proceedings of the VLDB Endowment, Vol. 15, No. 11 ISSN 2150-8097.
doi:10.14778/3551793.3551864

uncertain textual and contextual information, while the movements exist over media contexts. Failing to capture these characteristics may lead to a low event identification quality and further generate a low quality event delivery, which affects the user engagement in crisis directly. For instance, during COVID-19, when we ignore the location contexts of positive cases, the information on all US COVID-19 cases will be delivered to Australian Government Department of Health, which generally mix up the US COVID-19 event with Australia COVID-19 event. On the other hand, when we ignore the event movement over location in event recommendation, the movement of people infected by COVID-19 from US to Australia could not be recognised, thus either Australian government or the relatives of these US patients could not take actions effectively. Second, we need to design novel solutions for identifying the interested users of an incoming social event. As coupling event behaviours exist among event parts in migrating event, a good recommendation solution will recognize these event interactions, and greatly ascertain the system quality. Finally, we need to design efficient techniques to accelerate the user-event relevance identification. Due to the big volume of social media, sequential event-user relevance matching over a single processor is clearly infeasible for the real-time event detection and recommendation over microbogs.

The previous studies have various definitions of social events catered for individual applications and domains, like burst [34], topics [21, 33], composition of multiple event elements [38], real-world occurrence with evolution over consecutive time periods [9, 11, 26] etc. However, they did not consider the event evolution [33, 34, 38], or studied the event evolution over consecutive time periods only [9, 11, 21, 26]. While migrating events involve non-consecutive space and time changes, all existing solutions assume the temporal event evolution is consecutive, which cannot be extended for migrating event identification by simply adding a location dimension into their models. Though location-based recommendations [32] incorporate location information, such as that from a mobile device, into algorithms to attempt to provide more relevant recommendations to users, the item location they handled (e.g. restaurant, or traffic jam in Sydney City Centre) is fixed and reliable, not movable. The same type of different items (restaurant in Melbourne, restaurant in Sydney) are treated as same for recommendation depending on a target user's location change just as the events investigated in traditional event detection. However, migrating events involve the movement of a single item over space and time, where the location information of social media cannot reliably reflect the event happening due to the consecutive and non-consecutive location shift. Thus location-based recommendation cannot be applied for migrating events. Geotagged tweet pattern analysis approaches explore the semantics of movements such as transportation methods, frequent visiting sequences and keyword descriptions [10], or takes geotagged locations as fixed location ranges [12, 15]. However, these methods only handle the simple predictable user behaviour movement within a short time period for transportation field, or treat geotagged locations as a particular attribute of tweets. Thus, none of existing methods can recognize the complex event migration over both time and space. While interactions exist among users and sub-events, existing models lack mechanisms to capture this information. To overcome the challenges of social event recommendation, we propose a framework for Migrating Event Identification and Recommendation (MEIR) over media streams. Specifically, we first propose a Concept TF/IDF model (ConTF/IDF) to handle the uncertainty of media content. Then, we design a novel user influence graph over social network, which incorporates users' interaction to identify the event migration over space and time and generate recommendation in dynamic environment. We design a novel hash-based scheme over Apache Spark to speed up the event recommendation in big media data. Our main contributions are summarized as follows:

- We propose a novel ConTF/IDF model over the concepts of a social post constrained by its time and location, with a cosine-based similarity over the model. While ConTF/IDF handles the media uncertainty, it does not need the training process for inferring the related concepts.
- We propose a novel MUIG model to estimate the influence distribution over a use's social network. MUIG well captures the user interactions over events, enabling the detection of event migrations and event coupling for recommendation.
- We design an efficient event similarity join over Apache Spark to improve the recommendation efficiency, reducing the time cost by a novel hash-based hyper-cone data partition with a novel upper-bound-based candidate pruning.
- Last but not the least, we propose a novel algorithm which well maintains the social updates in microblogs. The test results prove the effectiveness and efficiency of MEIR.

2 RELATED WORK

This section reviews the existing research related to this work, including social event detection and social media recommendation.

2.1 Social Event Detection

Typical event detection methods can be categorized into two types: non-location constrained [9, 26, 33] and location constrained [11, 17, 27, 28, 35, 36, 38]. In [33], Xing et al. explored the event-related hashtags containing contexts like locations and dates, and the concise sub-event related descriptions for enhancing the quality of sub-event discovery over twitter. In [26], Peng et al. proposed Pairwise Popularity Graph Convolutional Network model for event classification and evaluation classification. In [9], Cao et al. proposed a Knowledge-Preserving Incremental Heterogeneous Graph Neural Network (KPGNN) for incremental social event detection. These methods are inapplicable to the space-sensitive social events.

Approaches have been proposed to incorporate the location as an attribute of social media for space-sensitive applications. In [27], a probabilistic spatiotemporal model was produced for the target event to find the center and the trajectory of the earthquake event location. In [28], Singh et al. aggregated the user interest levels at different geo-locations as social pixels, which were further used for the situation detection and showcased using a Swine flu monitoring application. In [17], Kim et al. developed a visual tool that allows users to intuitively understand the differences between topic movements over space and time and demonstrate the topic movement on the Great East Japan Earthquake. In [36], Yin et al. detected the situational topics using time and location for crisis events such as London riots or Earthquake in Virginia. In [35], Yin et al. detects the stable temporal topics such as flu and michael jackson death

by exploring the spatial and temporal prior information. In [38], Zhou et al. proposed a location and time constrained topic model to detect the composite social events for disasters such as flood and Cyclone. In [11]. Chen et al. explored the user retweeting behaviour to capture the event evolution over time during crisis. In [8], Cai et. al exploited physical user interaction to model the influence propagation of targeted campaigns or advertisements. However, with these techniques, the topic movement and the spatio-temporal situation detection fix time or location in a spatio-temporal range [11, 27, 28], if not infeasible for the event migration scenarios that allow the space and time to be non-consecutive [8, 35, 36, 38].

Studies have been done to detect patterns in geo-tagged tweets for social events [10, 12, 15]. In [10], Chen et al. analyzed the user movement patterns from geo-tagged tweets to explore the semantics of movements including transportation methods, frequent visiting sequences and keyword descriptions. However, they only handled the simple user behaviour movements in transportation within short time intervals. In [12], Choi et al. proposed a topic model-based visual analytics system TopicOnTiles for detecting anomalous events in an area from geo-tagged tweets. However, they defined event as what happened in a particular time and region. In [15], Huang et al. detected the spatial-temporal patterns of events from geo-tagged tweets. However, they focused on small-scale spatial-temporal events and their textual content. All these tweet pattern detections treated geo-tags as a particular location attribute of tweets, and cannot handle the complex events with non-consecutive migrations over time and space. None of them can be adjusted to incorporate location for migrating event detection as they all assume that the movements over time and space are limited to a small range, which follows certain distributions and is predictable. However, in migrating events, location movements are unpredictable and locations cannot reliably reflect event happening, which conflicts with the assumptions of existing models. This work fully exploits the social user trust relationship and proposes a maximal user influence graph to identify the event migrations.

2.2 Social Media Recommendation

Social media recommendation can be put into two categories: non-personalized [19, 23–25, 39] and personalized [16, 18, 20, 22, 31, 41, 42]. Methods have been proposed for non-personalized media or event recommendation in event-based social networks. In [39], Zhou et al. proposed to fully exploit the user contexts hidden in shared communities for the cold-start video recommendation. In [19], Liao et al. proposed an event recommendation model that considers participant influence, and exploits the influence of existing participants on the decisions of new participants. In [23], Macedo et al exploited contextual signals from EBSNs for the future planed event recommendation. In [25], Mo et al. explored the graph theory for event scoring in event recommendation. All these methods focus on the planned events on an EBSN which are non-personalized and simple, while infeasible for the complex migrating crisis events happening in an unpredictable manner and lasting for long period. In [24], Madisetty recommend users the events based on their popularity. However, they ignore the personalized user interests.

Personalized social recommendation identifies the relevant media to target users' interests reflected by their user history. In [20], Liao et al. modeled the deep, non-linear influence of contexts on users, groups, and events through multi-layer neural networks for recommending planned events to groups in EBSNs. In [22], Ma et al. developed a social network and self-attension-based method for event recommendation in EBSNs. In [16], Jhamb et al. proposed group-aware event recommendation in EBSNs using the user-oriented and event-oriented latent factors. These methods were customized for the planned events in EBSNs, but unsuitable for the unplanned streaming events in microblogs. Zhou et al. recommend streaming items by capturing the diversity of items and the effect of influential users [42]. However, this method cannot capture sufficient information on the complex contents, contexts and interactions of events. Zhou et al. proposed extendible CCIG that captures the contents, contexts and the interactions between features to recommend social items in big media sets [41]. However, the location in a CCIG is a coordinate pair of the position attached to a media. which cannot capture the location movement in migrating events. Thus CCIG can only handle the traditional social events without location movement, but not work over migrating events. Location-based recommendation help people discover attractive points of interest (POI). For example, Xie et al. [32] developed a graph-based embedding to capture the sequential, geographical, temporal cyclic and semantic effects in a unified way. However, location-based recommendation only handled the coordinates of POIs that are fixed and reliable. It cannot work for migration events involving complex location shift. This work handles migrating social events involving consecutive and non-consecutive location movements, focuses on designing a maximal user influence graph model that captures location movements and a set of query optimization techniques over Apache Spark for real-time recommendation.

3 PROBLEM FORMULATION

In MEIR, there are two vital elements: event and event migration. The event describes what happens at a location and a time, while event migration indicates the event movement over space and time.

Definition 1. *An event is defined as a real world occurrence over a space and time range. Formally, an event is described as a triple* (lr, tr, v)*, where* lr *is a location region,* tr *is a time range, and* v *is a vector in the concept space, which describes the real world occurrence, such as what is happening, who are involved, the effect of the occurrence etc. A social message is an instance of an event, describing one or more aspects of the event. A number of messages sharing the common hashtags together with their retweets in a time period form a sub-event on an event. A message together with its retweets form a sub-event candidate.*

Definition 2. *Given two sub-events* $E_1(lr_1, tr_1, v_1)$ *and* $E_2(lr_2, tr_2, v_2)$*, if* E_1 *and* E_2 *describe two stages of a real world occurrence happening within different time ranges and space regions, they are relevant ones involving event migration.*

To detect an event or event migration, we can extract four types of features, textual, time, location and user connection, from each tweet. Here, textual features are concepts extracted from the keywords of each tweet. Time is derived from the timestamp of each tweet. Locations are extracted from both tweets and users' profile, while social connections are from the users who reply, retweet,

mention a tweet. The textual, location and time features decide the aspects of an event, while user connection plays a vital role in monitoring event migration, reflecting the users' interest on relevant events with migration over space and time. We perform personalized event recommendation over twitter by considering the user interests and the event migration over space and time.

Definition 3. *Given an incoming sub-event E_i over tweet stream, social event relevance function f, our personalized event recommendation constructs a user profile $p(u)$ for each user, detects a list of users, U, with the best relevance to E_i, i.e., for any $u_1 \in U$ and $u_2 \notin U$, $Sim(E_i, u_1) \geq Sim(E_i, u_2)$.*

We address the problem of effective and efficient event recommendation over twitter by exploiting four types of features. MEIR shown in Figure 1 mainly includes two parts: event detection and recommendation. In event detection, each tweet is first modelled over different features, and the event-related tweets in a time slot are obtained. The event migrations are then detected by social user behavior. In event recommendation, user profiles are constructed based on their browsing history and interest changes over time and space. The recommendation is generated by the similarity join over user profiles and the incoming sub-events in each time window.

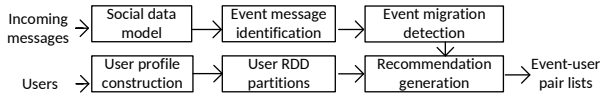


Figure 1: MEIR Framework

4 SOCIAL EVENT MIGRATION

This section presents our new model for social sub-event detection, and event migration detection by the users' network connection.

4.1 Social Sub-event Detection

We aim to find the social sub-events based on event contents, time and location. We incorporate database-oriented fusion for combining the similarities over these attributes, and focus on building a model robust to the attribute uncertainty without costly learning.

4.1.1 Data Modeling. To capture the textual information of messages, there are mainly two types of representations, topic models and TF/IDF model. Topic models such as LDA [7] well handles the uncertainty in documents by Bayesian inference. However, when topic models are applied in high dynamic social networks, they have to be updated frequently by training to keep its high effectiveness. Meanwhile, our recommendation needs to find the social events and deliver those involving migration to interest users, which is extremely time-critical. Thus, more efficient data model is required to meet the time requirement. TF/IDF model counts the number of each keyword in a document and that in the whole document set statistically, which does not rely on any training. Thus, its effectiveness does not degrade with the dynamic update of media streams, which is time efficient. However, the traditional approach directly applies TF/IDF model on the tweet tokens, which cannot handle the variations or incompleteness in textual information, i.e, various expressions on the same entities. Thus the correlation of tokens is not well captured. Consequently, some semantically relevant messages with various token representations cannot be identified.

We propose a new Concept TF/IDF (ConTF/IDF) to keep the superiority of traditional TF/IDF model, and overcome its weakness. Note that a *concept* in this paper refers to a word that can be found from the knowledge graph ConceptNet [1]. Unlike the TF/IDF that constructs a vector over the tokens of each document, we build the ConTF/IDF over a set of concepts for each keyword with the support of the ConceptNet. Similar to WordNet, ConceptNet supports the operations like query expansion and determining semantic similarity. Meanwhile, ConceptNet has advantages of making practical context-oriented inference over real-world texts due to its focus on concepts-rather-than-words. For a message M, we extract its keywords $\{w_i\}$. For each keyword w_i, we exploit ConceptNet to find a number of its concepts, $\{c_i^j\}$, including its analogous and relevant concepts. A ConTF/IDF vector $V_i = (v_1, ..., v_i, ..., v_d)$ is constructed over the concept set of each keyword, $\{c_i^j\}$. Here, d is the size of the *concept vocabulary* and v_i is the TF-IDF weight of i^{th} dimension of $\{c_i^j\}$ in the concept space. All the ConTF/IDF vectors of the keywords $\{V_i\}$ in a post are averaged over each concept dimension to form its textual feature. We denote this textual feature as topic vector. Here, a *topic* describes the primary subject of the related concepts in a message. In practice, the concept vocabulary is very big, leading to extremely high dimensionality of topic vectors that need to be mapped into a lower space. It is proved that the statistics over 4-grams of corpus can well keep the high quality of clustering over the original social media data [41]. Thus, following [41], we construct the ConTF/IDF vectors over the 4-grams of concepts, and apply SVD to reduce their dimensionality to 50.

Once the ConTF/IDF vector of each post is constructed, we can use different measures, such as cosine similarity, L_p-norm and Jaccard similarity, to decide the matched messages. It has been proved that cosine similarity is a better measure since the direction of a document vector is more important than the magnitude [37]. Considering this superiority, we choose cosine measure for ConTF/IDF vectors. Given two vectors, V_1 and V_2, the cosine similarity measures the cosine of the angle between them, as computed by:

$$S_{cos}(V_1, V_2) = (V_1.V_2)/(\|V_1\|\|V_2\|) \quad (1)$$

where $V_1.V_2$ is the dot product of V_1 and V_2, $\|V_1\|$ and $\|V_2\|$ the magnitudes of the two compared vectors respectively. In national security applications like crisis management, social events are constrained within certain time and space regions. Considering the time delay of social post and continuity of the event, the event of a message usually happens within a time range covering its post time point. Given a social post, we describe its temporal information as a time range centered at its timestamp, i.e. $tr :< t - \tau, t + \tau >$. Given the time ranges of two posts, $tr1 :< t_1 - \tau, t_1 + \tau >$ and $tr2 :< t_2 - \tau, t_2 + \tau >$, we define their temporal similarity as the ratio of their range intersection to their range union as equation 2:

$$\gamma(tr_1, tr_2) = (tr_1 \cap tr_2)/(tr_1 \cup tr_2) \quad (2)$$

Given two locations, $l_1 :< lt_1, lg_1 >$ and $l_2 :< lt_2, lg_2 >$, we measure the location similarity based on their great-circle distance [3], which is computed by $GD = R.arccos(sinla_1.sinla_2 + cosla_1.cosla_2.cos(lo_1 - lo_2))$, where R is the Earth radius (6371km). Then their space similarity is derived and normalized as:

$$\lambda(l_1, l_2) = 1 - GD/M_T \quad (3)$$

where M_T is the maximal distance between two possible similar locations used to normalize the location similarity.

4.1.2 Detecting Sub-events. Intuitively, the hashtagged tweets are usually about certain social events, and the unhashtagged tweets similar to a hashtagged one are usually a component of the same event while not noisy posts. Thus, we apply a two-step online sub-event detection that first generates a number of sub-event seeds by grouping the hashtagged posts with their retweets within a time window, and then finds the sub-events that match any hashtagged seeds from the candidates extracted from non-seed-hashtagged messages and non-hashtagged messages in this time slot. As such, the noise posts unrelated to the current events are excluded directly.

Given a sub-event E_1 and a sub-event candidate E_2, their relevance is measured by their global similarity $\widetilde{Sim}$ at sub-event level. As a sub-event or sub-event candidate is formed over certain topic in a time slot over stream, tweets in each of them are relevant over content and time. Thus, turning message matching to sub-event matching, we can simply use the centre points of the sub-event clusters for the matching with respect to the textual content and time attributes. However, as the location shift and event migration, the centre location of a sub-event cannot reflect the event happening in real applications. To overcome this problem, we embed Hausdorff metric over their location sets into the location similarity. Hausdorff measures the greatest of all distances from a point in one set to the closest point in the other set, which allows the location relevance of sub-events is captured in a flexible manner. Given two location sets L_1 and L_2, their Hausdorff distance $d_H(L_1, L_2)$ is defined as:

$$d_H(L_1, L_2) = max\left\{\sup_{l_1 \in L_1}\inf_{l_2 \in L_2} GD(l_1, l_2), \sup_{l_2 \in L_2}\inf_{l_1 \in L_1} GD(l_1, l_2)\right\} \tag{4}$$

Let l_{1c} and l_{2c} be the centre locations of L_1 and L_2 respectively. We define the overall location distance of L_1 and L_2 as:

$$GHD(L_1, L_2) = (GD(l_{1c}, l_{2c}) + d_H(L_1, L_2))/2 \tag{5}$$

The space similarity of E_1 and E_2 is derived and normalized as:

$$\lambda(L_1, L_2) = 1 - GHD/M_T \tag{6}$$

Having the models for the textual information, location and time of social media, we can fuse them using a good integration function to obtain the similarity of sub-events over these attributes. Existing relevance fusion has been done for search fusion [29] and social recommendation [39] by taking the average of the relevance over different attributes, the highest relevance score among them or the weighted sum of them. It has been proved that the weighted sum-based integration is effective for social media applications [39]. Thus we borrow the idea of relevance fusion in [39] to integrate the new measures that decide the similarities of sub-events over texts, time and space. Let V_{1c} and V_{2c} be the centre topic vector of E_1 and E_1, and tr_{1c} and tr_{2c} be the centre time of E_1 and E_1 respectively, the global similarity of E_1 and E_2 is computed by:

$$\widetilde{Sim}(E_1, E_2) = \omega_1 S_{cos}(V_{1c}, V_{2c}) + \omega_2 \gamma(tr_{1c}, tr_{2c}) + (1-\omega_1-\omega_2)\lambda(L_1, L_2) \tag{7}$$

Using this sub-event similarity, we can determine if a candidate is a true sub-event by its relevance score from a true sub-event seed. For each candidate sub-event, we identify its top relevant hashtaged subevents in its current time window and previous one. If any of the identified relevant subevents are a seed of the investigated event, This candidate is identified as a subevent of the investigated crisis.

4.2 Event Migration Detection

Social users are usually influenced by their trusted friends, and care about certain posts delivered by them. This trust reflects an inherent long-term user relationship although the social influences may evolve over time, which affects the post propagation over both consecutive and non-consecutive time periods. Based on this intuition, we propose a Maximal User Influence Graph (MUIG) model from which the influence distribution over a user's social network is computed, and the matching between the user influence distributions of two sub-events is conducted to detect the event migration. We construct MUIG over a training dataset, and update it with social stream dynamically. A MUIG, $G_{ui} = (U, E)$, is a directed graph consisting a user node set U and a set E of directed edges linking users. An edge from user u_1 to user u_2 denotes the probability that u_2 accepts u_1's information, which is computed by:

$$I(u_1, u_2) = \frac{the\ number\ of\ u_2's\ response\ on\ u_1's\ posts}{the\ number\ of\ u_1's\ posts} \tag{8}$$

Suppose we have a collection of 6 users, $u_1, ..., u_6$, where u_1 responded to u_2 and u_3, u_2 to u_3 and u_5, u_3 to u_4, and u_6 to u_1. Figure 2 shows an example of its user influence graph. Suppose that u_2 posted 10 messages, u_1 retweeted/commented 6 of them. Then the social influence probability of u_2 on u_1 is 0.6.

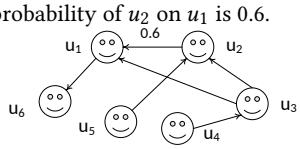


Figure 2: Example of user influence graph

We then construct the user influence distribution of each user by finding the maximal probability from it to all other users. Usually, there may be several paths from one user to another over the user influence graph. Each path may consists of one or several edges. The probability of a path from one user to the other is the product of all the edge weights. Let $(u_1, u_i, ..., u_j, u_2)$ be a path p from u_1 to u_2, then the probability of this path is:

$$I_p(u_1, u_2) = I(u_1, u_i) * ... * I(u_j, u_2) \tag{9}$$

Suppose that there are k paths, $(p_1, ..., p_k)$, from u_1 to u_2. The influence distribution of u_2 over u_1 dimension is obtained by finding the path with the maximum influence among k paths, as computed by: $I_m(u_1, u_2) = \max_{i=1}^{k} I_{pi}$. By computing the maximal influence probability between two users, we obtain an influence distribution for each social user, which can be further used for the detection of event migration. Given two sub-events E_1 and E_2 (E_1 is before E_2 temporally), let U_1 and U_2 be two sets of users in E_1 and E_2 respectively. The relevance probability of E_1 and E_2 is defined as:

$$Prob_r(E_1, E_2) = \frac{1}{\|U_1\|\|U_2\|}\sum_{i=1}^{\|U_1\|}\sum_{j=1}^{\|U_2\|} I_m(u_{1i}, u_{2j}) \tag{10}$$

5 SOCIAL EVENT RECOMMENDATION

This section presents our social recommendation solution, its optimization for real-time processing and dynamic update maintenance.

5.1 Event Recommendation

We will first present how to construct a user profile and then discuss the matching between an incoming sub-event and a user profile.

5.1.1 User Profile Construction. In real applications, users may have different interests or requests on the crisis situations. To deliver relevant events to different users, we need to construct a profile for individual users that reflects their interests. In traditional document recommendation, user profiles are usually constructed based on the keywords or concepts of the documents accessed by them. In our social event recommendation, we aim to deliver to right users the relevant events including the information on the same events and those involving event migration. Thus, the item relevance is not only decided by their content, but highly depends on location, time and social factors etc. In addition, one user may be interested in multiple social events. Considering these requirements in crisis, we build users' profiles by exploiting their interested sub-events $U = \{E_i\}$, each of which is described as a five-attribute tuple $E_i =< L, tr, V, I_m, un >$, where

- L denotes a set of locations to a sub-event, which is the union of the location regions of messages on this event.
- tr denotes the time range of a sub-event, which is the centre of the time ranges of messages on this sub-event.
- V is the topic vector of a sub-event, which is computed based on ConTF/IDF model over concept space.
- I_m represents a set of user influence vectors, each of which indicates the maximal influence of a certain user in sub-event E_i to the users in social community.
- un is the number of users in the community influenced by the users attached to E_i.

These five attributes interact with each other, and contribute to the relevance of social events.

5.1.2 User Profile Matching. We perform recommendation to users based on the content and contexts, and the social interaction between events and users. We propose a novel relevance matching between a user profile and an incoming event. Given a number of incoming events, our recommendation is performed by matching each incoming event with the sub-event sets of user profiles. Given an incoming sub-event $E_n =< L_n, tr_n, V_n, I_n, un_n >$ and a sub-event in a user profile $E_u =< L_u, tr_u, V_u, I_u, un_u >$, the similarity between them is derived by equations 7 and 10.

$$ESim(E_n, E_u) = (1-\alpha)\widetilde{Sim}(E_n, E_u) + \alpha Prob_r(E_n, E_u) \quad (11)$$

Then the similarity between E_n and the user profile U is defined as the maximal similarity value between E_n and the event in U.

$$ESim(E_n, U) = \max_{E_u \in U} ESim(E_n, E_u) \quad (12)$$

A set of incoming events are recommended to their top relevant users by checking the prediction probability of each to different user profiles. Given a set of incoming events in a time period, a naive recommendation is to match each incoming event with a large number of user profiles, each consists of multiple interested social events. This incurs high time cost. To fit our system for time critical online environment, we optimize the recommendation by efficient event similarity join over Apache Spark.

5.2 Event Recommendation Optimization

We will discuss how to optimize the event recommendation under Apache spark to efficiently identify the interested users for incoming events in the current timeslot. The Spark environment [4] is built on two components: a resilient distributed dataset (RDD) and the task scheduler. A RDD is a collection of elements partitioned across the nodes of the cluster that can be operated on in parallel. RDD supports two types of operations: transformations, which create a new dataset from an existing one, and actions, which return a value to the driver program after running a computation on the dataset. To minimize the shuffling and computational cost in the recommendation over Spark, we need to address two challenges: (1) how to find a good partition of the user profiles; (2) how to find the minimal set of the incoming events for each Spark processor.

5.2.1 User Profile Data Partition. Given a set of social events in user profiles, the first MapReduce job is launched to divide the event set into several partitions, each of which is described using its statistics information. Here, to remove the unnecessary computation over duplicate sub-events in multiple user profiles, we describe a social sub-event as a pair $< E_i, \{u_j\} >$, where E_i is the event itself, and $\{u_j\}$ is a set of users whose profiles include E_i. We perform the data partition based on locality sensitive hashing over the topic vectors. Since the user profiles are dynamically updated over microblogs, we ignore the time and location constraints, which reduces the chances of free processors. So more processors participate in the incoming tasks. We use the family of LSH functions base on p-stable distributions [13] to convert each topic vector into hash keys. Given a topic vector V, we use k independent hash functions of the form as below to obtain its k sets of hash values.

$$h_{a,B}(V) = \lfloor (a.V + B)/W \rfloor \quad (13)$$

Here, a is a d-dimensional vector whose elements are chosen independently from a p-stable distribution, W a constant, B a real number chosen uniformly from the range $[0, W]$. Given an Event E with topic vector V, with k hash functions over V, we can map it to a vector of k hash values. We use the sourcecode provided by Tao et al. and follow their parameter setting in [30], and give k a value 15 for Nepal earthquake data and 16 for Texas flood data.

Given a set of sub-events $\{E_i\}$ in user profiles, we perform the data partition based on the conflicts of their hash keys. First, the sub-events with conflict over all corresponding elements of their hash vectors are put into the same bucket. Then, we group the buckets based on the hash conflicts to ensure that the sub-events in a group is of high possibility to be (dis)similar to an incoming sub-event to similar extent, and the number of sub-events in each group is balanced. Figure 3 shows the algorithm for partitioning user profile data into a number of sub-event groups for multiple processors. It first convert each sub-event into a hash vector over the topic vector (line 1). A number of buckets are generated by putting the sub-events with conflicts over all dimensions of the vectors into the same group (line 2). The minimal content similarities of sub-events in a group to the group center are calculated, the buckets with small minimal similarities are split and reunioned based on their content similarities (lines 3). Then we allocate buckets to different groups for different processors (lines 4-13). The first group for the processor one is selected by finding the bucket that contains vectors

Procedure UserProfileDataPartition($\{E_i\}$, N).
$\{E_i\}$ - a user profile sub-event set, N- Number of groups
1. $\{V_i\}$ ←MapEvent2Vec($\{E_i\}$)
2. $\{B_i\}$ ← ConflictEvents2Buckets($\{E_i, V_i\}$)
3. CalculateSim2CentreVector($\{B_i\}$) BucketReUnion($\{B_i\}$)
4. $G \leftarrow \emptyset$ $B_j \leftarrow FindMaxConflictBucket(\{B_i\})$
5. $G_1 \leftarrow B_j$; remove B_j from $\{B_i\}$
6. **for** $i \in [2 : N]$
7. $B_j \leftarrow FindMaxConflictBucket2G(\{B_i\}, G)$
8. $G_i \leftarrow B_j$; remove B_j from $\{B_i\}$
9. **while** $\{B_i\} \neq \emptyset$
10. $G_i \leftarrow FindSmallestGroup(G)$
11. $B_j \leftarrow FindMaxConflictBucket2G_i(\{B_i\}, G_i)$
12. $G_i \leftarrow B_j$; remove B_j from $\{B_i\}$
13. return G. /*a set of user profile sub-event groups */

Figure 3: Partitioning profile sub-event data.

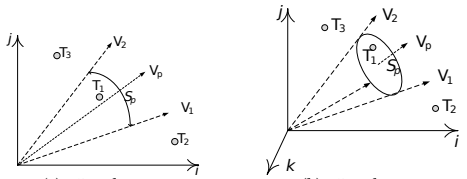
having the maximal conflicts with all other buckets (line 4). The bucket is removed from the bucket set once it is selected for the first processor (line 5). Following the first bucket allocation for the first processor, we select the first bucket for each of the rest processors (lines 6-8). By finding a bucket that has the maximal conflicts with the allocated buckets from the unallocated bucket set, the rest of processors are assigned to their first buckets, which are then removed from the unallocated set. We recursively allocate unallocated buckets to different processors (lines 9-11). To balance the workload of each processor, we give the priority to the one with the smallest data size, and select a bucket with the maximal conflict from the unallocated ones (lines 10-11). The selected bucket is removed from the bucket set after allocation (line 12). Finally, N sub-event groups for different processors are returned (line 13).

Once the profile data are allocated to processors, we produce a summary for each data partition, which records the ranges of topic vectors and influence vectors of its sub-events, the time and space range boundaries, the minimal and maximal numbers of its sub-events. Given a partition of n sub-events, we describe it with a pilot topic vector $Vp : (v_1, ..., v_d)$ with the cosine value of its biggest angle to topic vectors enclosed by the partition denoted by S_{cos}^{min}, an influence range vector $Ir : (< I_{1min}, I_{1max} >, ..., < I_{\|u\|min}, I_{\|u\|max} >)$, the influenced user number range $UNr :< un_{min}, un_{max} >$ where $\|u\|$ is the number of users in this partition, and the time range boundary $Tr :< tr_{min}, tr_{max} >$. The components of this summary are obtained by simply computing the average value of each dimension for the pilot topic vector and finding the minimal and maximal values over other attributes of messages in a partition. In addition, we generate a summary for each bucket under the partitions with the same structure as the data partition summary. The data partition summaries and bucket summaries are organized as a compact two-level tree structure that can be kept into memory. Using this in-memory two-level summaries described as their bound values, we can filter the incoming sub-event candidates and reduce the recommendation time cost.

5.2.2 Event Similarity Join over Apache Spark. Distributing user profiles onto different processors, with the created RDDs, data operations can be performed in parallel. Each processor is responsible for one partition of the user profiles. The similarity join from all history events in user profiles of the partition to the incoming sub-events in each time window is done in parallel for all processors. For sub-event similarity join over Apache Spark, a naive method is to sent the entire set of incoming sub-event stream in the current window to each processor to be joined with user profiles on it. However, this method may incur high cost due to the unnecessary operations during similarity join. Another way is to divide the incoming sub-event stream into disjoint partitions, and match the user profiles and sub-event stream over their partitions. However, one user profile partition may have matches over multiple stream data partitions, which requires additional stages for the final matching results and introduces extra shuffling cost of the data transfer between stages. Thus, we need to effectively reduce the shuffling cost and the sub-event similarity join cost. We identify a subset S_n of incoming sub-events for each partition S_p, and conduct the kNN join over S_n and S_p. The final recommendations are obtained by integrating all the results from different partitions.

Given a user profile partition S_p, we identify its corresponding subset S_n that contains all the sub-events interested by any users in it. However, we cannot get the exact S_n without performing the similarity join on S_p and the whole incoming sub-event set in the current time window. Thus, we derive a similarity bound based on S_p which decides S_n in an approximate way. Given a user profile partition or bucket S_p described as a triplet of pilot topic vector with the minimal cosine value and influence range vector $< V_p, S_{cos}^{min}, I_r >$, and an incoming sub-event E_i described using a pair of topic vector and influence vector $< V_i, I_i >$ together with the contexts, we define an upper bound similarity function, UP_{max}, to measure the similarity between a user partition and a sub-event. Before introducing UP_{max}, we first compute the UP_{max}^t that is the similarity between the user partition and a sub-event over topic attribute. To do this, we need to consider the position of the incoming topic vector, and decide what is the smallest angle between the topic vector of an incoming sub-event and any topic vectors of the partition. We called this angle *bound angle*. Applying this upper bound to incoming events, we can identify a much smaller subset of incoming events for each processor, which reduces the shuffling time cost in join operation over Spark and the similarity calculation operations for each RDD during joining. As the upper bound is applied to a small number of bucket summaries, the extra cost for calculating the bound values for each incoming event is neglectable comparing with the similarity matching over a big number of history events in user profiles. With the upper bound, the incoming sub-events unmatched with a partition do not need to be delivered to that partition for the similarity join.

When we decide the *bound angle* of a user partition with respect to a sub-event over topic vector space, we need to minimize its angle from this sub-event according to their relative locations. Figure 4 (a)-(b) show a user partition area and a sub-event topic vector with all possible relative locations over 2-dimensional and 3-dimensional subspaces respectively. The topic vector of any sub-event in the user partition only appear in the area enclosed by V_1 and V_2 in Figure 4 (a). It is natural to decide the *bound angle* based on the relative position of the incoming sub-event. A sub-event located in this area as T_1 indicates the possibility of highest similarity value 1, which should not be used for candidate filtering. A sub-event below V_1 as shown by T_2 indicates a biggest similarity from it, then the



(a) 2D subspace (b) 3D subspace

Figure 4: Positions of sub-event topic vector

angle between V_1 and this sub-event will be the minimal angle of this sub-event and this user partition. The one above V_2 shown as T_3 indicates the biggest similarity with V_2, and further the bound angle is enclosed by V_2 and this sub-event. Moving from 2-dimensional to 3-dimensional subspace as in Figure 4 (b), all the sub-event topic vectors are enclosed by a 3-dimensional cone, where the boundary cannot be decided by a number of vectors. Likewise, extending to high-dimensional space, the topic vectors of a user partition will be a hyper-cone. Thus, the boundary topic vector can be numerous, and the one enclosing the bound angle cannot be easily decided. We decide the maximal cosine value of the bound angle with the help of the pilot topic vector that is the axis of the hyper-cone to the partition. Let β be the biggest angle between the axis of the hyper-cone V_p and a topic vector of user partition, θ be the angle enclosed by V_p and the topic vector V_i of an incoming sub-event E_i. The minimal angle between the user partition and V_i is $\theta - \beta$ if the V_i is outside of the hyper-cone. If it is enclosed by the hyper-cone, the smallest angle is 0. Denote $S_{cos}(V_i, V_p)$ as $S_{cos}(V_{ip})$, The topic similarity upper bound between the sub-event and user partition over the topic vector, $UP^t_{max}(E_i, S_p)$, is as below:

$$\begin{cases} 1 & \theta \leq \beta \\ S_{cos}(V_{ip})S_{cos}(V_{1p}) + \sqrt{(1 - S^2_{cos}(V_{ip}))(1 - S^2_{cos}(V_{1p}))} & otherwise \end{cases} \tag{14}$$

Next, we prove UP^t_{max} is an upper bound of S_{cos} of the sub-events by Theorems 1. With this bound, we can filter the sub-event candidates based on their topic vectors, which reduces the workload of each Spark processor in recommendation generation.

Theorem 1. *UP^t_{max} is an upper bound of $\widetilde{Sim}$ of the sub-events.*

Proof. When $\theta \leq \beta$, the maximal cosine value can be 1 as the incoming event is enclosed in the partition region. For the case $\theta > \beta$, we prove that UP^t_{max} upper bounds $\widetilde{Sim}$ in the whole space via a two-step deduction. We normalize the whole dataset, map all the data to the surface of a unit hyper-sphere in a 50-dimensional space. Given two vectors, V_i and V_j, the angle between them is the arc length between them on their geodesic of this unit hyper-sphere, denoted as $A(V_i, V_j)$. Given three vectors, the pilot vector V_c, a vector V_p on the boundary of the hyper-cone, and the incoming event topic vector V_i, there are two cases for their positions.

- V_c, V_p and V_i are on the same geodesic of this unit hyper-sphere. Under this condition, we have $A(V_i, V_p)$=$A(V_c, V_p)$-$A(V_c, V_i)$. i.e. $A(V_i, V_p)$=$\theta - \beta$
- V_c, V_p and V_i are not on the same geodesic of this unit hyper-sphere. Under this condition, V_c, V_p and V_i form a spherical triangle. Based on the property of spherical triangle that the sum of two edges is bigger than the third edge, we have: $A(V_i, V_p)$>$\theta - \beta$. i.e. $A(V_i, V_p)$>$\theta - \beta$

Thus, we have $A(V_i, V_p) \geq \theta - \beta$. As V_c and V_i are given vectors and $A(V_c, V_p)$ is fixed, we have fixed θ and β values. Thus the bound angle equals to $\theta - \beta$. We deduce the cosine value of $\theta - \beta$. Since $cos(\theta - \beta) = cos\theta cos\beta + sin\theta sin\beta$ and $sin\theta = \sqrt{(1 - cos\theta)^2}$, we have:

$$UP^t_{max} = cos(\theta - \beta) = cos\theta cos\beta + \sqrt{(1 - cos\theta)^2(1 - cos\beta)^2}. \tag{15}$$

Applying the values of $cos\theta$ and $cos\beta$ to equation 15, equation 14 holds. Thus, we conclude that UP^t_{max} in the bound value subspace upper bounds $\widetilde{Sim}$ in the whole space. □

For time context, it is easy to derive the upper bound of the time relevance between E_i and S_p, $UP^{tr}_{max}(E_i, S_p)$, based on the time boundary of S_p and the location of E_i. Let tr_{min} and tr_{max} be the minimal and maximal time ranges of S_p, τ be the radius of a time range, tr_i be the time range of E_i, $UP^{tr}_{max}(E_i, S_p)$ is computed by:

$$UP^{tr}_{max}(E_i, S_p) = \begin{cases} 1 & tr_{min} \leq tr_i \leq tr_{max} \\ \gamma(tr_i, tr_{min}) & tr_i < tr_{min} \\ \gamma(tr_i, tr_{max})\} & tr_i > tr_{max} \quad otherwise \end{cases} \tag{16}$$

$UP^{tr}_{max}(E_i, S_p)$ is formulated based on the following consideration: when tr_i is inside the time range of S_p, there may be a sub-event in S_p whose time range is totally same as tr_i. When tr_i is smaller than tr_{min}, the time relevance between E_i and any sub-event in S_p is no bigger than that between tr_i and tr_{min}. Likewise, when tr_i is bigger than tr_{max}, the time relevance between them is no greater than that between tr_i and tr_{max}. Thus, for any sub-events $E_j \in S_p$, $UP^{tr}_{max}(E_i, S_p)$ upper bounds $\gamma(E_i, E_j)$. For the space context, since we consider the migration events which move over a large space and show low location-based clustering quality, the upper bound filtering techniques over space are infeasible in this application. Thus, we directly apply the maximal similarity value 1 in the upper bound measure for the location similarity.

Incorporating the $d^{min}_H(E_i, E_v)$ and $GD(E_i, E_v)$ into equations 5-6, we have $\lambda(E_i, E_v)$ upper bounding $\lambda(E_i, E_j)$. Next we compute the UP^I_{max} that is the influence probability between sub-event E_i and user partition S_p. To do this, we estimate the user influential between E_i and S_p by searching the MUIG over the whole user set. All the users in S_p and influencing users in E_i are selected, and then ranked by the average influential values they can generate to E_i. Then, we select the top un_{min} users from S_p as the dominant ones for the calculation of $Prob_r$ from S_p to E_i. We define the relevance of S_p and E_i as the relevance probability of a virtual sub-event E_v and E_i, where E_v includes all the dominant users in S_p.

$$UP^I_{max}(E_i, S_p) = Prob_r(E_i, E_v) \tag{17}$$

Next, we prove the $Prob_r$ between E_v and E_i upper bounds the $Prob_r$ between any sub-event E_l in S_p and E_i.

Theorem 2. *Given any sub-event E_l in a user profile S_p and a sub-event E_i, let E_v be a virtual sub-event including the top un_{min} dominant users in S_p. $Prob_r(E_v, E_i)$ upper bounds $Prob(E_l, E_i)$.*

Proof. We first prove that the $Prob_r(E_l, E_i)$ in projected dominant space S_d with top un_{min} dominant users in E_l upper bounds the $Prob_r(E_l, E_i)$ in its whole space. Since for any users $u_j \notin S_d$ and $u_k \in S_d$, the maximal influence probability from u_j to a user in E_i is no bigger than that from u_k to this user. Thus the average of the

maximal influence probabilities over the top un_{min} dominant users is no smaller than that over all the users in a sub-event E_l. Thus, based on equation 10, we have $Prob_r(E_l, E_i)$ in S_d upper bounds the $Prob_r(E_l, E_i)$ in its whole space.

Then, we prove that $Prob_r(E_l, E_i)$ for E_v upper bounds that for S_d. Since the dominant users are top un_{min} users selected from all the sub-events in S_p, the un_{min} probability values of E_v is no smaller than their corresponding probability values to S_d. Thus based on equation 10, we have $Prob_r(E_l, E_i)$ to E_v upper bounds that in S_d. Based on the transitivity of inequality, we conclude that the following condition holds: $Prob_r(E_v, E_i) \geq Prob(E_l, E_i)$. Based on these upper boundaries of similarity, we can easily calculate the overall bound of the global event similarity in equation 11. □

5.3 Cost Analysis

We estimate the filtering power of the upper bound UP^t_{max} over the data partitions using two methods: our LSH-based partition and the existing competitor UP [41]. Denote the partition number as N, the divided sub-event groups as G=$\{G_1, G_2...G_N\}$, the sub-event group axis as $\{V_1, V_2...V_N\}$ and the corresponding half bound angles as $\{\theta_1, \theta_2...\theta_N\}$. Suppose that T is a preset relevance threshold. For an incoming sub-event E_i and a sub-event group G_i, if E_i is within a range with axis V_i and half angle $\theta_i + arccos(T)$, then G_i cannot be filtered out. Let V_G be the axis of the whole user profile set and θ_G be its half bound angle. Then the probability of group G_i being filtered out is: $p_f = 1 - (\theta_i + arccos(T))/(\theta_G + arccos(T))$. Thus for a set of N_e incoming sub-events, the filtering power of G_i is: $FPower = p_f * N_e/N_e = p_f = (\theta_G - \theta_i)/(\theta_G + arccos(T)$. The ratio of the filtering power of LSH-based partition and that of UP is: $RatioFP = FPower_{lsh}/FPower_{UP} = (\theta_G - \theta_i^{lsh})/(\theta_G - \theta_i^{UP})$. As the UP partition keeps all the sub-events uniformly distributed into all groups, and the LSH-based partition groups the most similar sub-events together, the θ_i^{lsh} is much smaller than θ_i^{UP}. Thus UP^t_{max} has higher filtering power over our LSH-based partition.

5.4 Social Updates Maintenance

In practice, due to the user interactions via social messages over streams, the social influence from a user to her followers may change over time. Furthermore, new users may come to the microblogs, while some existing users may not interact with each other after a time period. Thus, dynamic updates are necessary for the MUIG to reflect the recent social user influences and the new users to be allocated to processors. We will discuss the details on how to maintain the social updates periodically. The details on our social update maintenance algorithm is shown as Figure 5. First, we update the nodes and edges corresponding to the influenced users (lines 2-7). If the user is in the MUIG already, we update its connected edges (lines 4-5). Otherwise, the new user node is created, and the corresponding edges are constructed and inserted into the MUIG (lines 6-7). After that, we find the suitable partitions for the updated users, update the MUIG parts on the corresponding spark processors and update the summaries of the updated user partitions (lines 9-12). The final updated MUIG is returned (line 13).

Procedure UserUpdateMaintenance(G_{ui},$\mathcal{U}$).
G_{ui} - a MUIG, $\mathcal{U}$- a user set with interactions in recent timeslot
1. Let $\tilde{\mathcal{U}} = \emptyset$ be a set keeping new users
2. **for** each $u_i \in \mathcal{U}$
3. $\tilde{U} \leftarrow$ FindInfluencedUsers(u_i,U)
4. For each pair $< u_i, \tilde{u_i} >$
5. if $u_i \in G_{ui}$ and $\tilde{u_i} \in G_{ui}$, UpdateEdge($u_i, \tilde{u_i}$)
6. else if $u_i \notin G_{ui} || \tilde{u_i} > \notin G_{ui}$
7. $\tilde{\mathcal{U}} \leftarrow (u_i, \tilde{u_i})$; $e_i \leftarrow$ ConstructEdge($u_i, \tilde{u_i}$);
8. InsertMUIG(e_i)
9. **for** $u_i \in \tilde{\mathcal{U}}$, let $\{E_i\}$ be the user profile sub-event set of u_i
10. FindConflictProcessors($\{E_i\}$);
11. AllocateToProcessors($\{E_i\}$, u_i)
12. UpdateUserPartitionSummaries
13. return updated G_{ui} /*updated MUIG*/

Figure 5: Maintaining social user updates.

6 EXPERIMENTAL EVALUATION

This section evaluates the effectiveness and efficiency of our MEIR.

6.1 Experimental Setup

We conduct the experiments on data collected from Twitter during two natural disasters, Nepal earthquake 2015 and Texas Flood 2015. The Nepal earthquake 2015 data were collected from 15 April and 24 May 2015, and include 42.1G tweets within 1000km of earthquake centre. The raw data was filtered based on the keywords extracted from Wikipedia on April 2015 Napel earthquake [1]. The Texas Flood 2015 data consist of 16.3G tweets from Texas US from 12 May to 5 June 2015. We use a subset of Napel earthquake data set including the messages posted within 168 hours from 25 April to 1 May 2015, and that of Texas flood data from 168 hours posts between 22 May to 28 May 2015 for the effectiveness test. We capture the location information from user profiles and text messages, and filtered out the messages without location information. We use the subset of Napal earthquake messages posted in the period 15-24 April and those for Texas Flood in 12-21 May 2015 as the training set for the initial user interaction construction and concept vector construction. We use the Napal earthquake subset after 1 May 2015 and the Texas Flood subset after 28 May for the precision verification of recommendation. For each dataset, we conducted preprocessing that stems the texts and removes the stop words. The final filtered Nepal earthquake data set contains 3,141,036 tweets in total with 41,825 tweets in the subset for the period 25 April-1 May 2015. The final filtered Texas Flood data set contains 1,392,208 tweets in total with 39,855 tweets in 22 May - 28 May 2015. We manually built the ground truth of these two events. The ground truth of each migrating sub-event is labeled by three assessors based on the relevance judgements with the instruction of our event migration definition as in [40]. The reliability of this user study has been proved in [40]. All three assessors are PhDs majored in computer science and have background on social media media analysis to ensure they have good understanding on events. Each individual is given all the messages in the datasets in a random order. After viewing these messages, they were asked to give a rating score from 1 to 5 indicating if a message is relevant to an investigated event. Here, higher score indicates more relevance. A message with the rating no smaller than 4 is considered as semantically relevant. Finally, 15,613 tweets are labelled as NepalEquake and 2,921 tweets

[1] https://en.wikipedia.org/wiki/April_2015_Nepal_earthquake

are labelled as TexasFlood. Among them, 8229 NepalEquake ground truth tweets are in 25 April-1 May 2015, and 982 TexasFlood ground truth tweets are in 22 May - 28 May 2015.

For the event migration detection, the state-of-the-art event detection methods MGe-LDA [33] and RL-LDA [11], and our proposed two alternatives ConTF/IDF-TL and ConTF/IDF-TL-U (Concept TF/IDF with time, location and MUIG) are used for the effectiveness evaluation. MGe-LDA uses Hashtag topic model, and RL-LDA uses user retweeting behaviour topic model. For the event migration recommendation, three alternatives of our proposed approach, ConTF/IDF-TL-U, ConTF/IDF-TL and ConTF/IDF, are used in the comparison. Also, we derived the context-aware media recommendation, CCIG, [41] to migrating event recommendation for effectiveness comparison. CCIG is a state-of-art context-aware recommendation that exploits the content, time, location and users. For MGe-LDA [33] and RL-LDA [11] for event detection, and CCIG-based recommendation [41], we use their optimal parameters that have been tuned in their original papers.

6.2 Evaluation Methodology

We evaluate the effectiveness of migration detection in terms of two metrics, probability of missed detection (P_{Miss}) and probability of false alarm (P_{Fa}) over two datasets. These two metrics are widely used to evaluate the effectiveness of event detection and topic tracking tasks [14, 38], and fits the application of emergency applications [2]. A target is defined as a ground truth tweet that should be assigned to an event, while a non-target is the opposite. These metrics are defined as: $P_{Miss} = \frac{number\ of\ missed\ detections}{number\ of\ targets}$ and $P_{Fa} = \frac{number\ of\ false\ alarms}{number\ of\ nontargets}$. The effectiveness of event recommendation is evaluated by a metric precision@k (P@k), which is the number of relevant users recommended in the top-k recommendation, and computed by: $P@k = \frac{\#\ of\ recommended\ users\ @k\ that\ are\ relevant}{\#\ of\ recommended\ users\ @k}$

We evaluate the system efficiency in terms of the overall time cost of event detection and recommendation over tweet streams. The combination of two subsets, one week for each, is used for the efficiency test. We test the overall time cost of recommendation over the 2 weeks tweets under Apache Spark. All tests are conducted on a server using an Intel(R) Xeon(R) CPU E3-1230 v5@3.40GHz 3.41 GHz with 32.0GB RAM running 64-bit operating system.

6.3 Effectiveness Evaluation

We test the effectiveness of event migration and recommendation.

6.3.1 Effectiveness of event migration. We first test the effect of parameters, τ, ω_1 and ω_2, and α to obtain their optimal values. We then compare our solution with the state-of-art competitors.

Effect of τ. We evaluate the impact of uncertain time range τ to find the optimal τ value by conducting the event detection over time attributes. We test the P_{Miss} and P_{Fa} of the non-migrated event detection by varying τ from 0 to 10m. At each τ value, we measure the P_{Miss} and P_{Fa} of the detection by finding the top 100 sub-events relevant to the true seed sub-events obtained by groudtruth hashtags, and report the P_{Miss} and P_{Fa} at each τ. As reported in Figure 6 (a)-(b), the P_{Miss} of the event detection drops

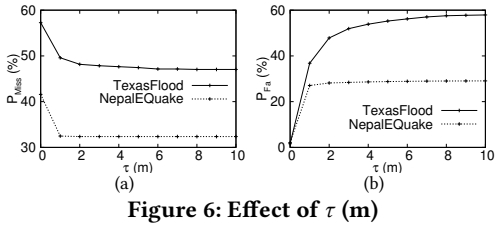


Figure 6: Effect of τ (m)

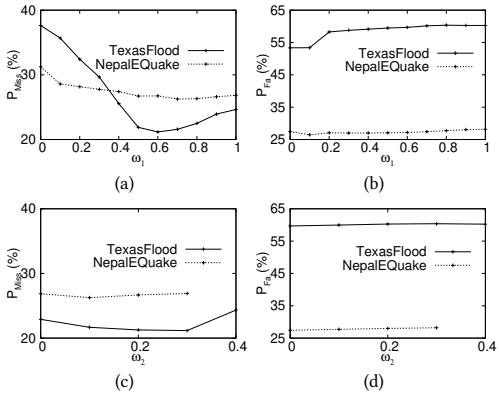


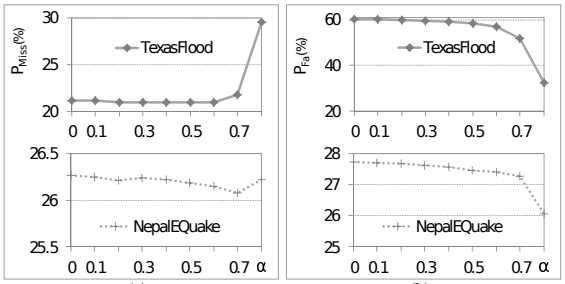
Figure 7: Effect of ω_i

first, and becomes steady after an optimal τ value. Here the optimal τ value is 2 for Nepal earthquake 2015 data, 8 for Texas Flood 2015 data. Meanwhile, the P_{Fa} of the detection increases although the speed of increasing become slow after the optimal τ. Thus, we set the default τ to 2 for Nepal earthquake 2015 data and to 8 for Texas Flood 2015 data for a good trade-off of P_{Miss} and P_{Fa}.

Effect of ω_i. We test the effect of ω_1 and ω_2 on the effectiveness of our event detection over two datasets, where ω_1 is the weight for topic vector, ω_2 is for time context and (1-ω_1-ω_2) is the weight for location context. We fix τ to its default value. For each combination of ω_1 and ω_2, where the sum of ω_1 and ω_1 is 1, we test the effectiveness of event detection using our ConTF/IDF-TL. The optimal ω_1 is first decided by varying it from 0 to 1, and the optimal P_{Miss} and P_{Fa} of the system are reported at each ω_1 value. Figure 7 (a)-(b) show the optimal P_{Miss} and P_{Fa} of the detection at each ω_1 value for two datasets. Clearly the P_{Miss} is dropped first, and reaches to an optimal value, ω_1=0.7 for NepalEQuake 2015 and ω_1=0.6 for TexasFlood 2015, and then increases after the optimal value. Meanwhile, P_{Fa} increases steadily with the increase of ω_1. Thus, we set default ω_1 to 0.7 for NepalEQuake 2015 and 0.6 for TexasFlood 2015 to get a good balance between P_{Miss} and P_{Fa}. Then we fix ω_1 to the default value and test the effectiveness of the detection by varying ω_2 from 0 to 0.3 for NepalEQuake 2015 and from 0 to 0.4 for TexasFlood 2015. Figure 7 (c)-(d) show the test results over two datasets. Clearly, the best performance can be achieved when we set ω_2 to 0.1 for NepalEQuake and 0.3 for TexasFlood.

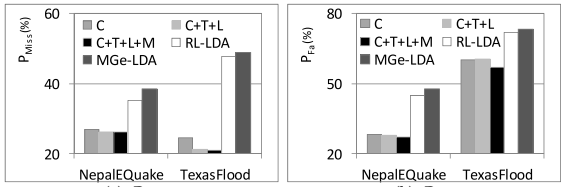
Effect of α. We evaluate the effect of user influence probability α over two datasets of the P_{Miss} and P_{Fa} of the migration detection results. We find the optimal α for the system by setting the parameters, τ, ω_1 and ω_2, to their default values, and computing the P_{Miss} and P_{Fa} of the migrating event detection at different α. As shown in Figure 8 (a)-(b), for the Texas flood data, when α is increased, the P_{Miss} of the detection keeps steady, while its P_{Fa} drops slowly. With the further increasing of the α value after 0.6,

[2]https://en.wikipedia.org/wiki/False_alarm



(a) (b)

Figure 8: Effect of α



(a) P_{Miss} (b) P_{Fa}

Figure 9: Effectiveness of event migration detection

the the P_{Miss} increases clearly and the P_{Fa} drops quickly. For Nepal earth quake data, the P_{Miss} of the detection drops first, reaches to the minimal value at $\alpha = 0.7$ and then increases again after this α value. Meanwhile, the P_{Fa} drops quickly with the increasing of α. This is because the user influence can be better used in finding the sub-events with migration over time and space with the increasing of α at the beginning. With the further increase of α, the capability of sub-event identification for the sub-events with little movement is reduced significantly, leading to the degradation of the detection performance. On the other hand, the effects of α on the event detection over two datasets present to be slightly different due to their different data characteristics. As Nepal earthquake 2015 is a more serious natural disaster, it involved more frequent discussions and user engagement over twitter than Texas flood 2015. Thus, the user influence took a more significant role in migration detection of Nepal earthquake. Thus, to achieve the optimal trade-off between P_{Miss} and P_{Fa}, we choose 0.7 for Nepal earth quake and 0.6 for Texas flood to be their default α values.

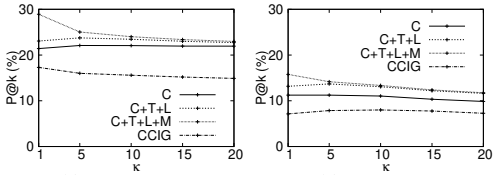
Discussion on parameter tuning. We follow existing work [41] to use exhaustive experimental evaluations to obtain their optimal values. We evaluate ω_1 and ω_2 jointly, which is an end-to-end joint parameter tuning. In practice, we can start the parameter tuning from the median parameter values, and test the next step points in bidirectional way until the optimal parameter values are obtained for a dataset. Alternatively, we can use binary search technique [3] to try the tunning points for the optimal parameter values. These can significantly reduce the test steps for obtaining the optimal parameters for a dataset.

Sub-event detection effectiveness comparison. We compare the effectiveness of migrating event detection using four approaches, MGe-LDA, RL-LDA, ConTF/IDF, ConTF/IDF-TL, and ConTF/IDF-TL-U by performing the migrating event detection over two streams. We set all mothods to their optimal values. Figure 9 (a)-(b) show the comparison of four approaches over two datasets in terms of P_{Miss} and P_{Fa}. Clearly, our ConTF/IDF-TL-MUIG performs much better

[3]https://en.wikipedia.org/wiki/Binary_search_algorithm

Table 1: Effect of dynamic updates on event detection

Datasets	NepalEQuake		TexasFlood	
	$P_{Miss}(\%)$	$P_{Fa}(\%)$	$P_{Miss}(\%)$	$P_{Fa}(\%)$
Update	26.0785	20.9611	20.9611	57.0451
NoUpdate	26.0785	27.2607	20.9611	57.0593



(a) NepalEQuake (b) TexasFlood

Figure 10: Effectiveness of recommendation

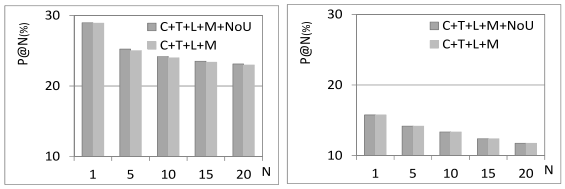
than other competitors, MGe-LDA and RL-LDA, on both metrics. This is because ConTF/IDF-TL-U well captures the content and context information, time and space, and the maximal user influence over microblogs, which enables the detection of event migration over these attributes. Though RL-LDA handles the temporal event evolution, it can only handle the direct and limited location change over social media because it only consider the retweeting behaviour of hashtagged social messages in each fixed time window. Thus, it cannot identify the non-consecutive movement over time and space in migrated events or non-hashtagged sub-events. This has proved that ConTF/IDF-TL-MUIG is superior to state-of-art techniques.

Effect of social updates on sub-event detection. We evaluate the effect of social updates on the effectiveness of sub-event detection, and compare our ConTF/IDF-TL-U which applies dynamic MUIG update and our ConTF/IDF-TL-U-NoUpdate which applies static MUIG over training dataset. Table 1 reports the P_{Miss} and P_{Fa} of two different approaches over two datasets. With the dynamic update maintenance, the detection is significantly improved for NepalEQuake dataset while slightly improved for TexasFlood dataset. This is because nepal earthquake is a much bigger event involving more user activities comparing with Texas flood. Our dynamic maintenance can adjust the model to reflect the recent user interactions, which leads to the improvement of the detection.

6.3.2 Effectiveness of event recommendation. This section first compares the event recommendation of different methods, and then evaluate the effect of social updates on event recommendation.

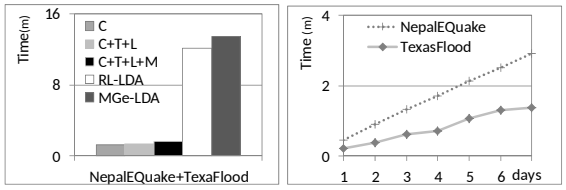
Event recommendation effectiveness comparison. We evaluate our final recommendation effectiveness by comparing four different approaches, ConTF/IDF, ConTF/IDF-TL, ConTF/IDF-TL-M and CCIG-based event recommendation over two datasets. We apply the optimal settings in the test, and report the $P@k$ of different event recommendation approaches at each k value from 1 to 20. Figure 10 (a)-(b) show the comparison results over two datasets in terms of the recommendation precision at each k. Clearly, ConTF/IDF-TL-M-based recommendation performs best for both datasets, followed by our alternative approach ConTF/IDF-TL. Meanwhile, all proposed alternatives perform much better than the CCIG-based approach. This is because ConTF/IDF well infers the semantic relevant concepts without training process. Also, ConTF/IDF-TL-M can recognize the event migration over both time and location.

Effect of social updates on event recommendation. We test the effect of social updates on the recommendation effectiveness



(a) NepalEQuake (b) TexasFlood

Figure 11: Effect of dynamic updates on recommendation



(a) Comparing event migration (b) Cost of update maintenance

Figure 12: Efficiency of event detection

by comparing ConTF/IDF-TL-U and ConTF/IDF-TL-U-NoUpdate methods over two datasets. Figure 11 (a)-(b) show the $P@k$ values of two approaches over two datasets. As we can see that with dynamic updates, the performance of recommendation keeps steady. This reflects a fact that the prediction on future user interests is mainly effected by the inherent long term interactions among users.

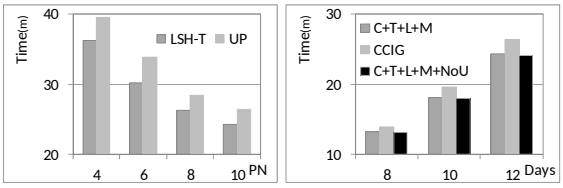
6.4 Efficiency Evaluation

We evaluate the efficiency of our MEIR by first reporting the results of event migration, followed by those for event recommendation.

6.4.1 Efficiency of event migration. We evaluate the efficiency of our event migration ConTF/IDF-TL-U by comparing with existing competitors, MGe-LDA and RL-LDA, in terms of time cost for the whole stream. In this test, we vary the tweet streams from 1 to 8 weeks, and report the time cost of detection with each method. As shown in Figure 12(a), our ConTF/IDF-TL-U (noted as C+T+L+M) achieves much higher efficiency than MGe-LDA and RL-LDA. This is because our ConTF/IDF-TL-U is constructed on the top of ConceptNet, which enables the effective inference without costly model training. For MGe-LDA and RL-LDA, due to the topic model training in the stream processing, these graphical models incur high time cost of model learning. RL-LDA costs more time than MGe-LDA, due to the additional graph construction over retweeting behaviours, which compromise its effectiveness improvement.

6.4.2 Efficiency of social updates. We test the cost of social updates over two datasets using different sizes of updates. For each of two datasets, the training set is treated as a source set and the following one week social media is considered as a test set. We varying the test sets from 1 to 7 days updates. As shown in Figure 12(b), the cost increases steadily with the increase of social media update size. This is because we adopt incremental social update strategy, which well controls the update maintenance cost.

6.4.3 Effect of event recommendation optimization. We examine the effect of our proposed LSH-based data partition over topic vectors (denoted as LSH-T) and our proposed upper bounds jointly by comparing with the Uniform data partition (UP) [41]. We test the time cost changes of recommendation with the support of LSH-T



(a) Effect of optimization (b) Comparing recommendations

Figure 13: Efficiency of recommendation

and four upper-bound-based filtering. Figure 13(a) shows the time cost comparison of different data partition strategies under 4 to 10 partitions. Clearly, our LSH-T method performs better than HP, because it can group similar events together that are further processed over a Spark processor. With this partition, a large number of irrelevant events can be filtered out without further spark processing after mapping operations. This reduces the workload of processors and the shuffling cost of the join operation.

6.4.4 Comparing Different Recommendations. We compare our recommendation approach with the state-of- the-art method CCIG in terms of the average response time of each system over the data stream by varying the incoming data streaming size from 8 to 12 days. The time cost of each system is shown as Figure 13(b). Obviously, our approach is much faster than CCIG due to the efficient candidate filtering using the proposed upper bounds. Though CCIG takes advantages of parallel processing, all the incoming events have to be passed to processors to identify the interested users due to its uniform data distribution. Our LSH-T partition strategy only requires to process less event candidates due to the efficient candidate filtering based on upper bounds proposed, thus high efficiency is achieved.

7 CONCLUSIONS

This paper studies the problem of migrating social event detection and recommendation. First, we propose a new ConTF/IDF model to overcome the uncertainty in social media. Then, we propose a novel MUIG model that fully recognizes the user influence in microblogs, and infers the migrations between users. Finally, we propose an efficient recommendation generation under Apache Spark with advanced algorithms and upper bound filtering, well maintain the updates in microblogs periodically and incrementally. The test results have proved the high efficacy of MEIR.

ACKNOWLEDGMENTS

Xiangmin Zhou's work is partially supported by the ARC Discovery Project (DP200101175). Lei Chen's work is partially supported by National Key Research and Development Program of China Grant No. 2018AAA0101100, the Hong Kong RGC GRF Project 16202218, CRF Project C6030-18G, C1031-18G, C5026-18G, CRF C2004-21GF, AOE Project AoE/E603/18, RIF Project R6020-19, Theme-based project TRS T41-603/20R, China NSFC No. 61729201, Guangdong Basic and Applied Basic Research Foundation 2019B151530001, Hong Kong ITC ITF grants ITS/044/18FX and ITS/470/18FX, Microsoft Research Asia Collaborative Research Grant, HKUST-NAVER/LINE AI Lab, Didi-HKUST joint research lab, HKUST-Webank joint research lab grants and HKUST Global Strategic Partnership Fund (2021 SJTU-HKUST).

REFERENCES

[1] https://conceptnet.io/.
[2] https://www.dominos.com.au/coupon-voucher/vic-ringwood-98593.
[3] https://en.wikipedia.org/wiki/Great-circle_distance. Accessed on 23 Sept 2020.
[4] https://spark.apache.org/.
[5] http://www.thereligionofpeace.com/attacks/attacks.aspx?Yr=2015.
[6] http://www.melbourne.vic.gov.au/parking-and-transport/roads/Pages/road-closures.aspx.
[7] David M. Blei, Andrew Y. Ng, Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*,3:993–1022.
[8] Taotao Cai, Jianxin Li, Ajmal Mian, Rong-Hua Li, Timos Sellis, Jeffrey Xu Yu. 2022. Target-Aware Holistic Influence Maximization in Spatial Social Networks. *IEEE Transactions on Knowledge Data Engineering*, 34(4): 1993-2007.
[9] Yuwei Cao, Hao Peng, Jia Wu, Yingtong Dou, Jianxin Li, and Philip S. Yu. 2021. Knowledge-Preserving Incremental Social Event Detection via Heterogeneous GNNs. *WWW*, pages 3383-3395.
[10] Siming Chen, Xiaoru Yuan, Zhenhuang Wang, Cong Guo, Jie Liang, Zuchao Wang, Xiaolong Zhang, and Jiawan Zhang. 2016. Interactive Visual Discovering of Movement Patterns from Sparsely Sampled Geo-tagged Social Media Data. *IEEE Transactions on Visualization and Computer Graphics* , 22(1):270-279.
[11] Xi Chen, Xiangmin Zhou, Timos Sellis, and Xue Li. 2018. Social event detection with retweeting behavior correlation. *Expert Systems with Applications*, pages 516–523.
[12] Minsuk Choi, Sungbok Shin, Jinho Choi, Scott Langevin, Christopher Bethune, Philippe Horne, Nathan Kronenfeld, Ramakrishnan Kannan, Barry Drake, Haesun Park, Jaegul Choo. 2018. TopicOnTiles: Tile-Based Spatio-Temporal Event Analytics via Exclusive Topic Modeling on Social Media. In *CHI*, pages 1-11.
[13] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Symposium on Computational Geometry*, pages 253-262.
[14] Jonathan G. Fiscus and George R. Doddington. 2002. Topic Detection and Tracking Evaluation Overview. *Topic detection and tracking*, pages 17 - 31.
[15] Yuqian Huang, Yue Li and Jie Shan. 2018. Spatial-Temporal Event Detection from Geo-Tagged Tweets. *ISPRS International Journal of Geo-Information*, 7(4): 150.
[16] Yogesh Jhamb and Yi Fang. 2017. A dual-perspective latent factor model for group-aware social event recommendation. *Information Processing and Management*, 53(3): 559-576.
[17] Kyoung-Sook Kim, Ryong Lee, and Koji Zettsu. 2011. *m*Trend: discovery of topic movements on geo-microblogging messages. In *GIS* , pages 529–532.
[18] Lei Li, Dingding Wang, Tao Li, Daniel Knox, and Balaji Padmanabhan. 2011. SCENE: a scalable two-stage personalized news recommendation system. In *SIGIR* , pages125–134.
[19] Yi Liao, Wai Lam, Lidong Bing, and Xin Shen. 2018. Joint Modeling of Participant Influence and Latent Topics for Recommendation in Event-based Social Networks. *ACM Transactions on Information Systems* , pages 36:29:1-29:31.
[20] Guoqiong Liao, Xiaomei Huang, Neal N. Xiong, Changxuan Wan. 2020. An Intelligent Group Event Recommendation System in Social networks. *CoRR*, abs/2006.08893.
[21] Jimmy Lin, Rion Snow, and William Morgan. 2011. Smoothing techniques for adaptive online language models: topic tracking in tweet streams. In *SIGKDD* , pages 422–429.
[22] Tiemin Ma, Rui Chen, Fucai Zhou, Shuang Wang, and Xue Wang. 2021. Social Event Recommendation Based on Social Relationship and Attention Mechanism. In *ICMAI*, pages 72–77.
[23] Augusto Q. Macedo, Leandro B. Marinho, and Rodrygo L.T. Santos. 2015. Context-Aware Event Recommendation in Event-based Social Networks. In *RecSys* , pages 123–130.
[24] Sreekanth Madisetty. 2019. Event Recommendation using Social Media In *ICDE*, pages 2106-2110.
[25] Yijun Mo, Bixi Li, Bang Wang, Laurence T. Yang, and Minghua Xu. 2018. Event Recommendation in Social Networks Based on Reverse Random Walk and Participant Scale Control *Future Generation Computer Systems* , 79: 383–395.
[26] Hao Peng, Jianxin Li, Yangqiu Song, Renyu Yang, Rajiv Ranjan, Philip S. Yu, and Lifang He. 2021. Streaming Social Event Detection and Evolution Discovery in Heterogeneous Information Networks. *ACM Transactions on Knowledge Discovery from Data*, 15(5): 89:1-89:33.
[27] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *WWW* , pages 851–860.
[28] Vivek K. Singh, Mingyan Gao, and Ramesh Jain. 2010. Situation detection and control using spatio-temporal analysis of microblogs. In *WWW*, pages 1181–1182.
[29] John R. Smith, Alejandro Jaimes, Ching-Yung Lin, Milind R. Naphade, Apostol Natsev, and Belle L. Tseng. 2003. Interactive search fusion methods for video database retrieval In *ICIP(1)*, pages 741–744.
[30] Yufei Tao, Ke Yi, Cheng Sheng, Panos Kalnis. 2009. Quality and efficiency in high dimensional nearest neighbor search. In *SIGMOD*, pages 563–576.
[31] Can Wang, Longbing Cao, Mingchun Wang, Jinjiu Li, Wei Wei, and Yuming Ou. 2011. Coupled nominal similarity in unsupervised learning. In *CIKM*, pages 973–978.
[32] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, Sen Wang. 2016. Learning Graph-based POI Embedding for Location-based Recommendation. In *CIKM*, pages 15–24.
[33] Chen Xing, Yuan Wang, Jie Liu, Yalou Huang, and Wei-Ying Ma. 2016. Hashtag-Based Sub-Event Discovery Using Mutually Generative LDA in Twitter. In *AAAI*, pages 2666–2672.
[34] Junjie Yao, Bin Cui, Yuxin Huang, and Xin Jin. 2010. Temporal and Social Context Based Burst Detection from Folksonomies. In *AAAI*, pages 1474–1479.
[35] Hongzhi Yin, Bin Cui, Hua Lu, Yuxin Huang, and Junjie Yao. 2013. A unified model for stable and temporal topic detection from social media data. In *ICDE*, pages 661–672.
[36] Jie Yin, Andrew Lampert, Mark Cameron, Bella Robinson, Robert Power. 2012. Using Social Media to Enhance Emergency Situation Awareness. *IEEE Intelligent Systems*, 27(6): 52–59.
[37] Shi Zhong. 2005. Efficient streaming text clustering. *Neural Networks*, 18(5-6):790–798.
[38] Xiangmin Zhou, and Lei Chen. Event detection over twitter social media streams. *VLDB Journal* 23(3):381–400, 2014.
[39] Xiangmin Zhou, Lei Chen, Yanchun Zhang, Longbin Cao, Guangyan Huang, and Chen Wang. 2015. Online Video Recommendation in Sharing Community. In *SIGMOD*, pages 1645–1656.
[40] Xiangmin Zhou, Lei Chen, Yanchun Zhang, Longbin Cao, Guangyan Huang, and Chen Wang. 2017. Enhancing online video recommendation using social user interactions. *VLDB Journal* 26(5): 637-656.
[41] Xiangmin Zhou, Dong Qin, Lei Chen, and Yanchun Zhang. 2019. Real-time context-aware social media recommendation. *VLDB Journal* 28(2):197–219.
[42] Xiangmin Zhou, Dong Qin, Xiaolu Lu, Lei Chen, and Yanchun Zhang. Online Social Media Recommendation Over Streams. In *ICDE*, pages 938–949, 2019.