Streaming Social Event Detection and Evolution Discovery in Heterogeneous Information Networks

HAO PENG, Beihang University, China
JIANXIN LI*, Beihang University, China
YANGQIU SONG, Hong Kong University of Science and Technology, China
RENYU YANG, University of Leeds, UK
RAJIV RANJAN, Newcastle University, UK
PHILIP S. YU, University of Illinois at Chicago, USA
LIFANG HE, Lehigh University, USA

Events are happening in real-world and real-time, which can be planned and organized for occasions, such as social gatherings, festival celebrations, influential meetings or sports activities. Social media platforms generate a lot of real-time text information regarding public events with different topics. However, mining social events is challenging because events typically exhibit heterogeneous texture and metadata are often ambiguous. In this paper, we first design a novel event-based meta-schema to characterize the semantic relatedness of social events and then build an event-based heterogeneous information network (HIN) integrating information from external knowledge base. Second, we propose a novel Pairwise Popularity Graph Convolutional Network, named as PP-GCN, based on weighted meta-path instance similarity and textual semantic representation as inputs, to perform fine-grained social event categorization and learn the optimal weights of meta-paths in different tasks. Third, we propose a streaming social event detection and evolution discovery framework for HINs based on meta-path similarity search, historical information about meta-paths, and heterogeneous DBSCAN clustering method. Comprehensive experiments on real-world streaming social text data are conducted to compare various social event detection and evolution discovery algorithms. Experimental results demonstrate that our proposed framework outperforms other alternative social event detection and evolution discovery techniques.

Additional Key Words and Phrases: Social event detection; Event evolution; Streaming data; Heterogeneous information network; Graph convolutional network; Fine-grained categorization; DBSCAN; Pairwise learning

*This is the corresponding author

A preliminary version [66] of this article appeared in the Proceedings of the 28th International Joint Conference on Artificial Intelligence, Pages 3238-3245 (IJCAI'19). Authors' addresses: Hao Peng, Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, China, and with School of Cyber Science and Technology, Beihang University, Beijing, China, penghao@act.buaa.edu.cn; Jianxin Li, Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, China, and with State Key Laboratory of Software Development Environment, Beihang University, Beijing, China, lijx@act.buaa.edu.cn; Yangqiu Song, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, HongKong, China, yqsong@cse.ust.hk; Renyu Yang, School of Computing, University of Leeds, Leeds, UK, r.yang1@leeds.ac.uk; Rajiv Ranjan, Computing Science and Internet of Things, Newcastle University, Newcastle, UK, raj.ranjan@newcastle.ac.uk; Philip S. Yu, Department of Computer Science, University of Illinois at Chicago, Chicago, USA, psyu@uic.edu; Lifang He, Department of Computer Science and Engineering, Lehigh University, Bethlehem, USA. lih319@lehigh.edu.

1556-4681/2021/2-ART32 $15.00
https://doi.org/0000001.0000001

ACM Reference Format:
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 Trans. Knowl. Discov. Data.* 1, 1, Article 32 (February 2021), 32 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

Nowadays, various social media platforms, such as Weibo, Twitter, Facebook, Instagram, Forum site and Blog, etc., have become major sources for publicizing social events. With a large amount of events being announced on social media, a large number of comments, reposts and discussions with opinions and emotions from social network users have been generated, and such content can reflect public opinion about many political, economic, security, employment and social welfare, and education issues, etc. Mining of social media posts, such as online social event detection and evolution discovery, will benefit a lot of real applications, such as predictive analysis [7, 46, 68, 73], disaster risk management [60], public opinion analysis [45, 49], information organization [4], recommended systems [50] and others [14]. In general, real-time social event detection and evolution focus on learning high-precision models for identifying event-related clusters from large-scale social messages, as well as fast streaming processing from online social data.

The tasks of real-time social event detection and evolution discovery are more challenging than traditional text mining or social network mining, since a social event is a combination of social user network and text streams in terms of short messages over it. More specifically, on the one hand, social events are described in short messages and usually contain keywords and different types of entities, such as person, location, organization, score, date and time, etc [2, 8, 31]. This leads to social events being very complicated and heterogeneous. Besides, events are posted, commented or retweeted by social media users at a specific time. Thus, modeling social events needs to consider heterogeneous elements as well as social users within social posts. On the other hand, with the spread of social practices including the participation of new social users and external interventions, events often change over time and streaming nature makes capturing useful semantic information difficult [10, 26, 28]. In addition, the variety of events is very diverse and covers a wide range, but the number of events per category is small. This feature is very important in social event analysis but has not yet been studied in much detail. Therefore, due to the above mentioned facts, it is desirable to develop streaming social event detection and evolution discovery methods that consider heterogeneous and dynamic features of events, as well as small sample size problems in each category as a whole.

In the literature, several studies have been aimed at streaming social event detection and evolution discovery. However, to our knowledge, none of them consider these two problems in a systematic manner. The majority of current studies focus on social event detection [11, 16, 17, 69], leveraging either homogeneous or heterogeneous network. The first line of thought [2, 6, 17] is to model streaming social messages as homogeneous words/elements co-occurrence network, and then extract abnormal subgraphs as the events. Despite the promising results from these studies, their detection accuracies remain unsatisfactory for building reliable and open domain event detection systems in practice. The second line of thought [31, 33, 40, 69] is to model social events as heterogeneous network, which use manually defined frames-based event definitions for extracting social event frames from corpus. The frame-based methods can extract entities and their relationships, but can only be used in a limited number of event types, such as earthquake disaster, stock market, venues, politics, etc. Moreover, most of them are pipeline models consisting of multiple machine learning models to incorporate different levels of annotation and features, and thus their capacities are limited by the components. For social event evolution discovery, traditional methods mainly focus on exploring evolutionary relationships between events based on various similarity measures, such as TF/IDF [5], KeyGraph [57, 61, 71], event element based schemes [48, 62] and bipartite graph matching methods [47], which allow to infer the strong association relationships between events. However,

similar to the existing detection methods, the above evolution discovery methods are unable to capture the different influences of heterogeneous event elements.

In this paper, we first represent social messages as hyper-edge in an HIN, where all the keywords, entities, topics, social users and time can be connected by this hyper-edge, and define an event meta-schema to characterize the semantic relatedness of social events and build event-based streaming HIN. In order to enrich the HIN, we extract some information as a complement of the relationships based on the external knowledge base and algorithms. For accurate social event similarity calculation, we define a weighted Knowledgeable meta-paths Instances based Event Similarity measure, namely *KIES*, from semantically meaningful meta-paths. Second, we then design a novel Pairwise Popularity Graph Convolutional Network model, namely PP-GCN, to learn not only the representation of each event instance, but also optimal weights of meta-paths in event classification and evolution classification tasks, respectively. Third, in streaming scenario, we use meta-path similarity strategy to extract historical social messages to build a provisional event-based HIN, and then we make use of both the optimal weights of meta-paths, similarity measure *KIES* and a heterogeneous DBSCAN clustering method to detect the social events. We also estimate the best time delay which is compatible with both the capacity of data acquisition and time consumption of event detection in batch manner, according to experimenting different scales of social messages. In addition, we also employ both the heterogeneous DBSCAN, namely H-DBSCAN, the optimal weights of meta-paths and event similarity to gather social events with evolutionary relationship. Finally, in order to speed up the streaming processing of methods, we employ multi-threaded parallel processing technology in the construction of HINs, message searching and clustering to achieve real-time event detection and evolution discovery. The source code of this work is publicly available at *https://github.com/RingBDStack/social-events*. The streaming social event detection and event evolution system is online at *http://ring.act.buaa.edu.cn*.

A preliminary version of this work appeared in the proceedings of IJCAI 2019 [66]. This journal version has extended the static fine-grained social event categorization model to streaming social event detection and evolution discovery models, and this full version involves several improvements in upgrading methodology and the frame structure of the proposed approach. We added time element and temporal nearest neighbor relationship when modeling event-based HIN. We also supplemented more explicit explanations of the meta-path in the two scenarios of event detection and event evolution. We presented novel streaming social event detection and evolution framework, and adopted the DBSCAN method - differing from the K-means method that requires to specify the number of clusters - to realize open domain event detection and evolutionary discovery based on density clustering. More in-depth experimental results and further analysis are discussed to demonstrate the effectiveness and efficiency of the proposed framework. We supplied the variances of the results of social event mining. For streaming social event mining scenarios, the results on all of the online time consumption and accuracy are displayed. Thorough and deeper analyses are presented, such as effectiveness, efficiency and case study. In particular, the contributions of this paper are summarized as follows:

- By modeling social events based on the streaming event-based HIN, the proposed framework integrates event elements including keywords, topic, entities, time, social users and their relations, in a semantically meaningful way, and can also calculate the similarity between any two events.
- By modeling pairwise popularity graph convolutional network, the model achieves state-of-the-art results in static fine-grained event classification and evolution classification, and learns the optimal weights of meta-paths in different tasks.
- By learning the optimal weights of meta-paths on a small sample with PP-GCN model, the proposed social event similarity measure *KIES* based DBSCAN, referred to as H-DBSCAN, achieves new state-of-the-art results for static event detection and evolution discovery tasks, respectively.

- The combination of multiple meta-paths based social event searching, event similarity and the H-DBSCAN clustering in batch manner achieves state-of-the-art results in streaming social event detection and evolution discovery tasks, respectively.
- We conduct extensive experiments to demonstrate that the proposed models considerably outperform the state-of-the-art social event detection and evolution discovery methods. We further illustrate the effectiveness and efficiency of the proposed methods.

2 HETEROGENEOUS EVENT MODELING

In this section, we define the problem of modeling social events in heterogeneous information network (HIN) and introduce several related concepts, necessary notations and formulas.

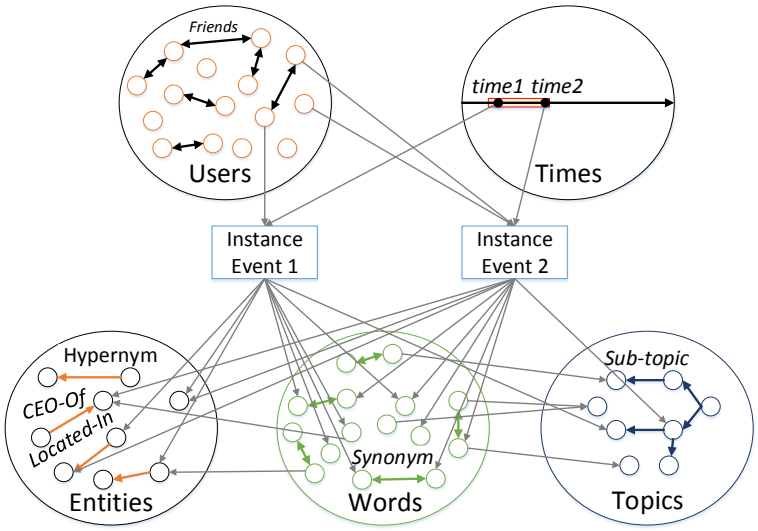

Fig. 1. Example of two instance events connected by different types of nodes and edges, where instance events are represented as hyper-edges.

2.1 Event Modeling in HIN

The definition and characterization of "social event" have received a great deal of attention in both academic and industrial fields, such as linguistics [34], cognitive psychology [87], tourism [21], social networks [88], etc. The simplest way to monitor social media events is to represent events as bags-of-words, but it will be more semantically meaningful if we can annotate words and multi-word-expressions as entities with types and different relationships. Intuitively, a social event generally refers to influential facts that appear on social networks and occur in the real world at some specific time, including creators (posters), named entities such as participants, organizations, festival, specific times, places, currency, address, etc., and other elements such as keywords and topics. For example, in the tweet "*Xinhua News Agency, 2019-04-05 19:46: Lightning strike has been confirmed as the cause of a forest fire that killed 27 firefighters and 3 locals in southwest China's Sichuan province, local forest public security bureau said at afternoon of April 5. The provincial forestry and grassland administration cited local police as saying that the lightning igniting the fire was first witnessed and reported by locals in Muli County, Liangshan Yi autonomous prefecture.*", we can extract the above time, keywords, typed entities and topics from

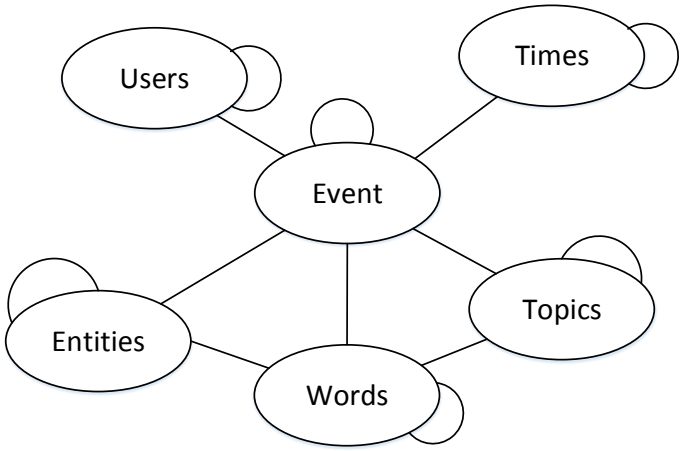

Fig. 2. Meta-schema of event-based HIN.

the original social message with NLP tools, and extract social user who post, comment or retweet message from social network. There are multiple event elements: **Date**: *April 5, 2019*; **County**: *Muli*; **Nation**: *China*; **Entity 1**: *Lightning Strike* **Entity 2**: *Local Forest Public Security Bureau*; **Entity 3**: *Provincial Forestry and Grassland Administration*; **Entity 4**: *Liangshan Yi Autonomous Prefecture*; **Entity 5**: *Sichuan Province*; **Entity 6**: *Southwest China*; **Topic 1**: *Natural disaster* **Topic 2**: *Forest fire*; **Poster**: *Xinhua News Agency*, **Posting Time**: *2019-04-05 19:46*, etc. Obviously, the above event's elements are of different types. We name the above elements and topics as *event-oriented elements*. Moreover, in addition to intuitive co-occurrence relationship, after extracting entities, we can make use of external knowledge base [9, 39, 83] to complement more relationships between entities, such as "**located-in**" relationships among the above locations, "**managed-by**" relationships between the above administrations, etc. Even within most of the events, there are some relationships between *event-oriented elements*, such as relationships between entities, relationships between keywords, relationships between topics, social relationships between social media users, time relationships, and so on. We name the above relationships as *event-elements relationships*. Thus we can model social media events as HIN [75, 76]. Next, we introduce how to extract the above rich *event-elements relationships*.

We use the manually organized synonyms to add simple synonym relationships among words in the HIN. For hierarchical topic structures and the affiliation relationship between keywords and topics in the HIN, we employ the hierarchical latent dirichlet allocation technologies [12, 22] (with about 30 most probable words for each topic). In order to extract the relationship between entities in the HIN, we consider both accuracy and efficiency, and tackle the problem by following three-steps. First, we retrieve the entity candidate set with the same name from an external knowledge base, such as the Chinese CN-DBpedia [83] or English Freebase [39]. Second, we use word embeddings [54, 55] based Word Mover's Distance (WMD) technology [36] to measure the text similarity between the context of entity in the social message and the description of entity in the candidate of the external knowledge base, and then choose the entity with the highest similarity score. Third, we query the relationship between the above aligned entities from the external knowledge base as the final relationship between entities in the HIN. In order to establish the relationship between entities and words in the HIN, we extract the keywords in the relevant description of each entity in the external knowledge base and use this affiliation as the relationship

between the entity and the keywords. For the relationship between social users, we consider users with a large number of friends or same interests, and extract friend relationship and shared community relationship [64] between social users in advance.

After extracting the above *event-oriented elements* and *event-elements relationships* from social messages, external knowledge base, resources and tools, we build an event-based HIN, as shown in Figure 1, where we name each social message as a instance event. On the one hand, one social event can usually consist of one unique event instance or multiple strongly associated event instances. On the other hand, one social event can also be regarded as a co-occurrence of *event-oriented elements*, and the *event-elements relationships* are conducive to explaining the relationship between various elements. Thus, one social event can be treated as a subgraph of the whole HIN. To characterize events on social messages, we give an example of the event-based HIN meta-schema, as shown in Figure 2. One particular advantage of the HIN is that meta-paths defined over types (e.g., a typical meta-path "event-keyword-event" represents the event similarity based on overlapped keywords between two event instances) can reflect semantically meaningful information about similarities, and thus can naturally provide explainable results for event modeling.

2.2 Preliminaries

We introduce some basic definitions based on previous works [75, 77, 80], and give some event-HIN related concepts, necessary notations and examples.

Definition 2.1. *A heterogeneous information network (HIN) is a graph $G = (V, E)$ with an entity type mapping $\phi : V \rightarrow A$ and a relation type mapping $\psi : E \rightarrow R$, where V denotes the entity set, E denotes the link set, R denotes the relation type set and A denotes the entity type set. The number of entity types $|A| > 1$ or the number of relation types $|R| > 1$.*

For example, Figure 1 shows an example of two instance events connected with different types of entities, keywords, topics, times, social users and relationships. After giving a complex HIN for event modeling, it is necessary to provide its meta level (i.e., schema-level) description for better understanding and modeling.

Definition 2.2. *Given a HIN $G = (V, E)$ with the entity mapping $\phi : V \rightarrow A$ and the relation type mapping $\psi : E \rightarrow R$, the meta-schema (or network schema) for network G, denoted as $T_G = (A, R)$, is a graph with nodes as entity types from A and edges as relation types from R.*

For example, Figure 2 shows an example of the HIN meta-schema characterizing events on social messages. Another important concept is the meta-path which systematically defines relationships between entities at the schema level.

Definition 2.3. *A meta-path P is a path defined on the graph of network schema $T_G = (A, R)$ of the form $A_I \xrightarrow{R_1} A_2 \xrightarrow{R_2} A_3 \cdots A_L \xrightarrow{R_L} A_{L+1}$ which defines a composite relation $R = R_1 \dot{R}_2 \cdot \vdots \cdot \dot{R}_L$ between types A_1 and A_{L+1}, where $\cdot$ denotes relation composition operator, and L is the length of P.*

For simplicity, we use object types connected by $\rightarrow$ to denote the meta-path when there are no multiple relations between a pair of types: $P = (A_1 - A_2 - \cdots - A_{L+1})$. We say that a meta-path instance $p = (v_1 - v_2 - \cdots - v_{L+1})$ between v_1 and v_{L+1} in network G follows the meta-path P, if $\forall l$, $\phi(v_l) = A_l$ and each edge $e_l =< v_l, v_{l+1} >$ belongs to each relation type $R_l \in P$. We call these paths as path instances of P, denoted as $p \in P$. R_l^{-1} represents the reverse order of relation R_l. We will introduce more semantically meaningful meta-paths that describe events in relation to the explanation of meta-paths instances based similarity.

Definition 2.4. *Given a meta-path $P = (A_1 - A_2 \cdots A_{L+1})$, Cou is a function of the count of meta-path instances such that $Cou_P(v_i, v_j) = M_P(v_i, v_j)$ where $M_P = W_{A_1A_2} \cdot W_{A_2A_3} \cdots W_{A_LA_{L+1}}$ and $W_{A_kA_{k+1}}$ is the adjacency matrix between types A_k and A_{k+1} in the meta-path P.*

For example, the composite relation of two instance events containing the same event element and co-occurrence relationship can be described as "Instance event-Element-Instance event (IEI)" for simplicity. This meta-path simply gives us $M_{IEI} = W_{IE}W_{EI}^T$, which is the dot product between event instances, where W_{EI} is the event Instance-Element co-occurrence matrix. We can give more event related meta-paths over different lengths, e.g., P_1: *Event instance-(containing)-Forest Fire-(contained by)-Event instance*, P_2: *Event instance-(containing)-Custom Line-(produced by)-Ferretti Group-(contained by)-Event instance*, P_3: *Event instance-(containing)-Stephen Curry-(belonging to)-Golden State Warriors-(located at)- Chase Center-(contained by)-Event instance*, etc. P_1 means two event instances are similar if they are containing the same named entity 'Forest Fire'. P_2 means two event instances are similar if they mention named entities 'Custom Line' and 'Ferretti Group', respectively, where Custom Line is a very famous luxury yacht brand produced by the Ferretti Group. P_3 means two event instances are similar if they can be associated by a chain of three event elements with meaningful relationships. For instance, two event instances contain 'Stephen Curry' and 'Chase Center', respectively, where the home of the Golden State Warriors, to which Stephen Curry belongs, is located at the Chase Center. Note that the meta-path does not need to satisfy symmetry. Finally, we can enumerate 22 symmetric meta-paths in the meta-schema of event-based HIN shown in Figure 2. Here, because a large number of social messages can be generated in a short period of time, we ignore the similarities that occur at the same time. But, we will implement event detection within the scope of the event instances that occur within a given time period.

Essentially, one social event consists of one or more instance events. However, in addition to the *event-elements relationships* within the event, there may be evolutionary relationships between events, which can help to reveal more complex and interpretable relationships between events [3, 51]. Therefore, the co-occurrence relationship of event evolution can be described as "Event-Instance event-eleMent-Instance event-Event (EIMIE)", where the relation between event and instance event is a composition relationship. Similar to various meta-paths between instance event, we also present three types of meta-paths that describe the evolution of events over different lengths, e.g., P_1: *Event-(consisting of)-Instance event-(containing)-Break Prisoner-(contained by)-Instance event-(consisted of)-Event*, P_2: *Event-(consisting of)-Instance event-(containing)-Break Prisoner-(synonym of)-Manhunt-(contained by)-Instance event-(consisted of)-Event*, P_3: *Event-(consisting of)-Instance event-(containing)-Break Prisoner-(synonym of)-Manhunt-(managed by)- Department of Justice -Instance event-(consisted of)-Event*, etc. Finally, we also can enumerate 22 symmetric meta-paths in the meta-schema of event-based HIN shown in Figure 2. Next, we introduce how to normalize the different impacts and the counts of different meta-paths on event similarity.

Similar to the weighted meta-path instances based text similarity [80], we also define our Knowledgeable meta-paths Instances based social (instance) Event Similarity measure, namely *KIES*. Intuitively, if two instance events or events are more strongly connected by some important (i.e., highly weighted and meaningful) meta-paths, they tend to be more similar. Formally, we have

Definition 2.5. *KIES: a knowledgeable meta-paths instances based social event instance or event similarity. Given a collection of meaningful meta-paths, denoted as $P = \{P_m\}_{m=1}^{M'}$, the KIES between two instance events or events e_i and e_j is defined as:*

$$KIES(e_i, e_j) = \sum_{m=1}^{M'} \omega_m \frac{2 \times Cou_{P_m}(e_i, e_j)}{Cou_{P_m}(e_i, e_i) + Cou_{P_m}(e_j, e_j)}, \tag{1}$$

where $Cou_{P_m}(e_i, e_j)$ is a count of meta-path P_m between two instance events or events e_i and e_j, $Cou_{P_m}(e_i, e_i)$ is that between e_i and e_i, and $Cou_{P_m}(e_j, e_j)$ is that between e_j and e_j. We use a parameter vector $\vec{\omega} = [\omega_1, \omega_2, \ldots, \omega_{M'}]$ to denote weights of meta-paths, where ω_m refers to the weight of meta-path P_m. $KIES(e_i, e_j)$ is defined in two parts including the semantic overlap and the semantic broadness in the numerators. Here, the semantic overlap is defined by the number of meta-path instances between two instance events or events e_i and e_j, and the semantic

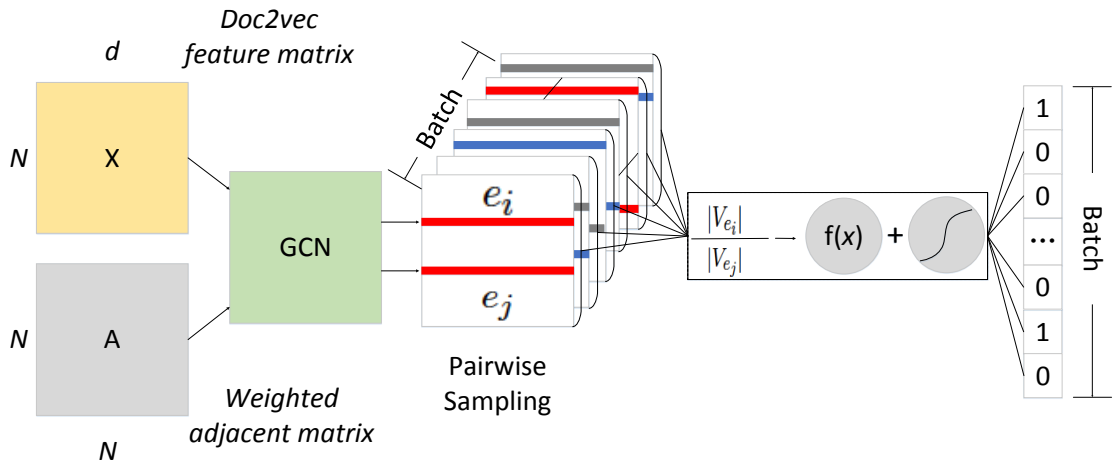

Fig. 3. An overview of the proposed Pairwise Popularity Graph Convolutional Network (PP-GCN).

broadness is defined by the number of total meta-path instances between themselves. Therefore, we can calculate a weighted *KIES* distance for any two instance events or events. Next, we introduce how to learn the different weights $\vec{\omega}$ of meta-paths through limited labeled datasets in event classification and event evolution classification tasks, respectively, by a novel pairwise popularity graph convolution network model.

3 PAIRWISE POPULARITY GRAPH CONVOLUTION NETWORK

After building the event-based HIN, we can calculate a weighted similarity for any two instance events or two events by the *KIES* for manually annotated social event data, and then construct an $N \times N$ weighted adjacency matrix A, where N is the number of instance events or events and $A_{ij} = A_{ji} = KIES(e_i, e_j)$. Then we use the unsupervised document representation technology [37] to learn a generalized feature for each event instance or event. So, we can also construct an $N \times d$ feature matrix X, where d is the dimension of the generalized feature. It is quite clear that, so far, we can make use of the popular GCN architectures [19, 35] to learn discriminating event representation and weights of meta-paths based on both the interactions among events and generalized event features in node classification task. Here, the input of the GCN model includes both A and X matrices, and one class represents one social event class or one social event evolution class.

In order to construct a preliminary GCN model, we utilize the popular multi-layer GCN architecture [35] with the following layer-wise propagation rule:

$$H^{(l+1)} = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}), \tag{2}$$

where $\widetilde{A} = A + I_N$, $\widetilde{D}$ is diagonal matrix such that $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$, I_N is the identity matrix, W is the parameter matrix, and l is the number of layers. Next, let Z be an output $N \times F$ feature matrix, where F to be the dimension of output representation per event instance. The input layer to the GCN is $H^{(0)} = X, X \in R^{N \times d}$, which contains original event instance feature, $H^{(l)} = Z$, and Z is graph-level output. And σ denotes an activation function such as Sigmoid or ReLU. Note that we also divide the N event instances or events into two parts: the training set and the testing set.

However, the real-world and real-time social events naturally have two problems of sparsity: the small number of instance events for each event and a large number of categories. So, we sample instance events pair, and judge

whether the pair belongs to one event to train instance event representation by a pairwise GCN model. For event evolution, we also sample the event pair, and judge whether the pair belongs to one event evolution to train event representation by the pairwise GCN model. As shown in Figure 3, we present the proposed pairwise GCN model. Before explaining the pairwise GCN model, we show how to implement a pairwise sampling to generate training samples. We assume that if a pair of instance events e_i and e_j from a training set belong to the same class, we name the pair e_i and e_j as a *positive-pair* sample. If a pair of instance events e_i and e_j from training set belong to two different classes, we name the pair e_i and e_j as a *negative-pair* sample. As shown in Figure 3, if the pair is a positive-pair sample, we represent its by two red lines; if the pair is negative-pair sample, we use both gray line and blue line to represent it.

After explaining how to generate training samples, we take instance event classification as an example to introduce the pairwise GCN model. Firstly, we randomly select R (i.e., 1024) instance events from training set of the original N samples as a preliminary set, then randomly select two instance events for each instance event in the set to construct one positive-pair sample and one negative-pair sample, and finally we construct a $2R$ instance event pairs set to train the pairwise GCN model. Here, both the positive-pair and negative-pair samples are equal to R. Second, we randomly sample B (i.e., 64) samples from the $2R$ (i.e., 2000) instance event pair set to construct one batch to forward propagation of our proposed model, as shown in Figure 3. Third, the second step is cycled E (i.e., 64) times to form an epoch. For the next epoch, we loop through the above three steps to train the GCN model and learn discriminating instance event representation and weights of meta-paths with instance event classification tasks. However, the above pairwise sampling based GCN model can not guarantee that the model has a smooth convergence and avoids overfitting during training. In order to avoid the uncertainty fluctuation of learning weights, we introduce a novel pairwise popularity GCN model to train it.

Taking instance event classification as an example, we assume that any event class has an average of r event instances. The probability that any event instance selected as a positive-pair sample is about $\frac{1}{r}$, and the probability of being selected as a negative-pair sample is about $\frac{1}{N-r}$. We note that $\frac{1}{r} \gg \frac{1}{N-r}$ in general. It is obvious that the negative-pair samples have more diversity than the positive-pair samples. Most previous works [15, 63, 82] have observed the phenomenon that the connected probability of a sample determines the popularity of it. The greater the probability of the connection, the greater the popularity of the features of the sample. Inspired by this observation, we assume that in feature representation learning, the modulus of the learned feature vector is also larger if the popularity is greater. So, the two modulus of learned event instance feature vectors of positive-pair will be closer. Similarly, the angle between the features of the two samples represents the semantic distance, and the smaller the angle, the closer the semantics. However, the range of angle between any two event instance features is $[0, \frac{\pi}{2}]$, but the range of the ratio of any two modulus can be $[1, +\infty)$. We believe that the ratio of modulus based pairwise popularity GCN model can generate more discriminating instance event representation than the angle based pairwise GCN, namely PA-GCN, as validated in the experiment section. Next, we introduce how to design a suitable neural network layer of the output, and calculate the loss function for the GCN model.

For more discriminating feature representation learning of our GCN model, we make use of the popularity of the output instance event feature vectors in Z to distinguish different classes. As shown in the Figure 3, for any two learned instance event vectors V_{e_i} and V_{e_j} that satisfy $|V_{e_i}| \geq |V_{e_j}|$, we employ a ratio of modulus $x = \frac{|V_{e_i}|}{|V_{e_j}|}$ as the input of a nonlinear mapping function $f(x) = -log(x - 1 + c)$, where the coefficient c is 0.01 to limit the upper bound of output of the function. Note that we assume that the ratio of modulus of *positive-pair* belongs to $[1, 2)$, and the ratio of modulus of *negative-pair* belongs to $[2, +\infty)$. So, the nonlinear mapping function $f(x)$ can map the above ratio x from [1, 2) to (0, 2], and [2, +∞) to (-∞, 0). Next, we add a Sigmoid function to map the output of the nonlinear mapping layer to 0 or 1 by a threshold 0.5. As shown in the Figure 3, one *positive-pair* or *negative-pair* input sample can only be paired with an output of 0 or 1. For one batch (64 pairs) samples, our model can generate one batch size (1×64) of a one-zero output vector. It stands to reason that we can use a cross

entropy function as our GCN model's loss function, and employ the popular stochastic gradient descent (SGD) method to iterate all parameters to train the model. The learned weights also will be used to measure similarity for any two social events or instance events, respectively, by different classification tasks of event evolution classification or event classification. To verify the smooth convergence and avoidance of overfitting ability of our model, we can perform over 7000 epochs, and observe that the evaluation criteria of the model changes over time in Section 5.

To evaluate the effectiveness of the PP-GCN model, we design the following steps to test any instance event or event sample t from the test set of the original N samples. We first assume that there is a total of C event or event evolution classes in the original N instance events. Secondly, we calculate the ratio of the modulus of the representation vectors for sample t and the remaining $N-1$ samples, respectively. If the ratio of the modulus falls within the interval of $[1, 2)$, we consider the pair belongs to the same class. Otherwise, if the ratio of the modulus is greater than 2, we consider the pair belongs to different classes. Then, for each event or event evolution class, we can get a probability that the sample t most likely belongs to it. Finally, we select the event class with the highest probability as the test output for the sample t. Note the fact that the event category of the test set may not be included in the event category of the training set.

After the previous analysis, we can calculate the similarity for any two instance events under the event-HIN framework and the weights $\vec{\omega}$. We also calculate the average similarity threshold θ_e between the instance events that make up the event, and the average similarity threshold θ_s between the events that make up the event evolution, from the manually annotated event dataset. Since our proposed meta-paths and similarity measure *KIES* have better interpretability, we also implement event clustering based on the learned weights of meta-paths and the *KIES* for event detection and event evolution tasks, respectively. Next, we introduce how to implement streaming event detection and event evolution based on the technologies of meta-paths based social messages searching, event-HIN, the weights $\vec{\omega}$, similarity measure *KIES* and clustering algorithm.

4 STREAMING SOCIAL EVENT DETECTION AND EVOLUTION

In this section, we introduce how to implement streaming social event detection and event evolution based on the proposed technologies and multi-core parallel heterogeneous DBSCAN algorithm [20, 65], respectively. For real-world original streaming social messages, we divide the daily streaming social messages into time slices evenly according to the acquisition time. However, the occurrence of a social event is likely to span multiple neighbouring time slices, in other words, a social event may cover social messages in multiple time slices. Similarly, a social event evolution set often covers more events over longer neighboring time slices. Thus, we introduce a meta-paths based social messages and event searching method to build provisional event-based HIN. Third, we propose the multi-core parallel heterogeneous DBSCAN (H-DBSCAN) method based on two distances including the weights learned by PP-GCN's static event classification and event evolution classification tasks, respectively, to achieve event detection and evolution discovery.

4.1 Meta-paths guided searching on HIN

As shown in Figure 4, we present two types of meta-paths that describe the relationship between social instance events, and the relationship between social events. As discussed in the example of the Definition 2.4, if any two social messages or social events can be connected by any instances of above meta-paths, we consider them to have interpretable relationships. In order to reduce the number of historically relevant social messages and consider only the real situation of social events, we limit the time span for searching social data. In the case of event detection, given any instance event E_i in the current t time slice, we collect the historically relevant social event instance E_j meeting:

$$\Delta T = E_i.time - E_j.time < T_1 \quad \text{and} \quad Score = E_i.elements \cap E_j.elements \neq \{\varnothing\}, \tag{3}$$

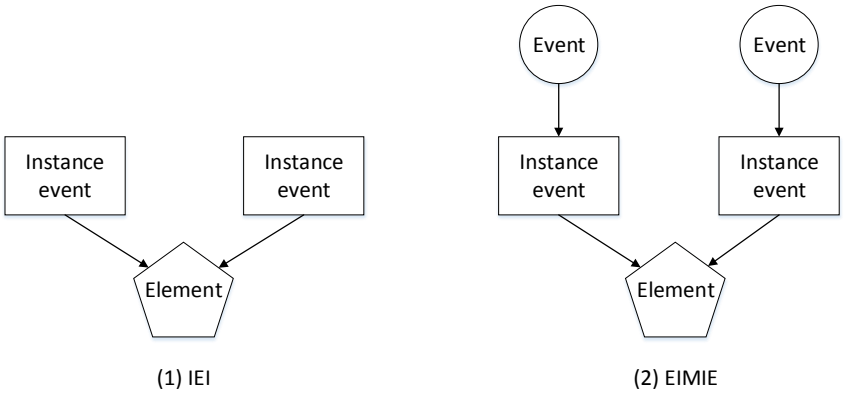

Fig. 4. Examples of two meta-paths in event-based HIN.

where T_1 refers to the longest time span for event detection. We note the current streaming social messages collection that combines latest social messages in the t time slices and historical relevant social messages, as $CM(t)$. Similar to event evolution, given any detected event E_m in current time slice, when we collect the historically relevant social event E_n meeting:

$$\Delta T = E_m.time - E_n.time < T_2, \quad T_1 < T_2 \quad \text{and} \quad Score = E_m.elements \cap E_n.elements \neq \{\varnothing\}, \tag{4}$$

where T_2 refers to the longest time span for event evolution. We denote the current streaming social events collection that combines latest social events in the t time slice and historical relevant social events, as $CE(t)$. Since the more relevant the event, the greater the correlation $Score$, we also limit the number of instance events or events retrieved based on the ranking of correlation $Score$. We have noticed that the computational complexity of the query in Eq. 3 is $O(n)$ in the HIN. And, the computational complexity of the query in Eq. 4 is $O(kn)$, where k refers to the average number of instance events contained in an event. Next, we present how to implement streaming event detection and evolution discovery based on $CM(t)$ and $CE(t)$, respectively.

4.2 Parallel H-DBSCAN based streaming event detection and evolution discovery and system

Social events are regarded as a co-occurrence of event elements including themes, dates, locations, people, organizations, keywords, social behavior participants and time, etc. And social events are the unique aggregation of a variety of semantics. We believe that the same event or the evolution tends to be cohesive, and there is a discernible distance between the semantics of different events or evolutions. So, we propose heterogeneous DBSCAN (H-DBSCAN), which is the *KIES* distance based DBSCAN. The main advantages of H-DBSCAN include that 1), it does not require one to specify the number of clusters in the data a priori; 2), it can find arbitrarily shaped clusters, and can even find a cluster completely surrounded by a different cluster; and 3), it has a notion of noise, and is robust to outliers. Intuitively, the H-DBSCAN method is very suitable for event clustering.

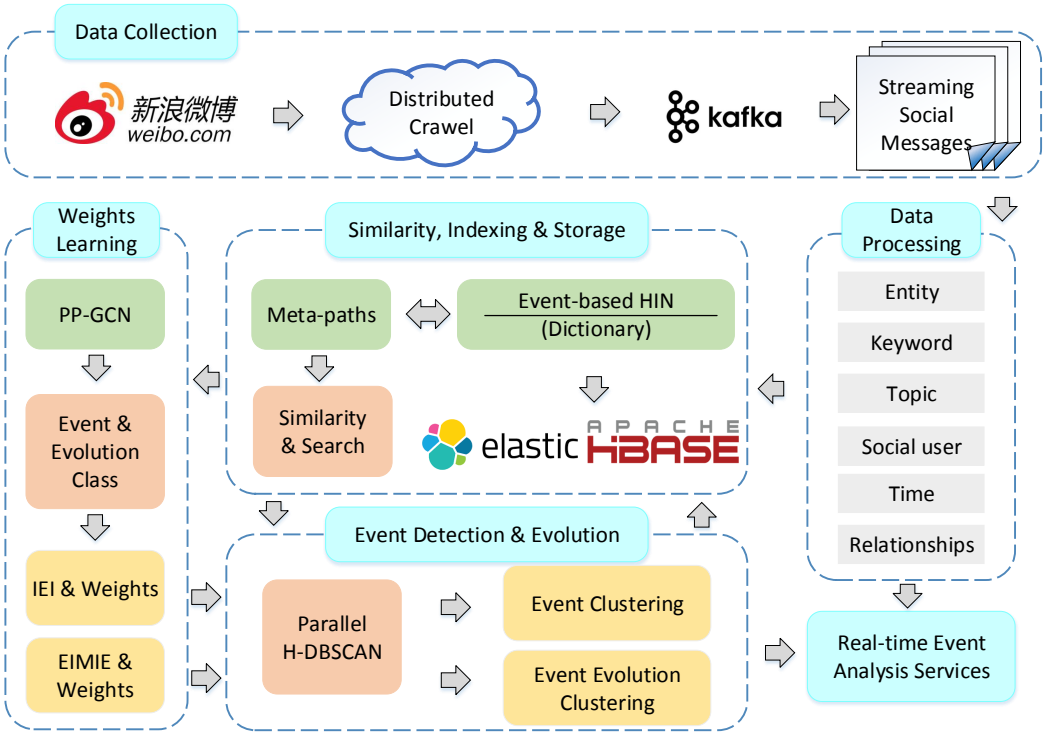

Fig. 5. Architecture of streaming social event detection and evolution discovery.

At the t time slice, the input of H-DBSCAN module includes the current streaming social messages: $CM(t)$ or $CE(t)$, distance threshold within a cluster: ϵ and the minimum number of points required to form a cluster: $minPts$. Compared to mapping the social data into coordinate space, it is more convenient to use a similarity distance-based adjacency matrix to model the near-far relationship among the social messages. For instance, for any two social messages e_i and e_j in the $CM(t)$, we build a distance-based adjacency matrix $DA(t)$, where $DA(t)_{ij} = DA(t)_{ji} = 1 - KIES(e_i, e_j)$, to represent the social data. For the given distance threshold ϵ, we choose the best similarity based on the value of the $KIES$ distances of the manually labeled same event (evolution) and different events (evolutions). For the given of the minimum number of points $minPts$, we choose the value of 1, due to the sparsity caused by the streaming data collect, a unique social event instance may be a separate social event. For faster and streaming social event detection and evolution discovery, we also use the multi-core parallel H-DBSCAN algorithm using p threads.

The pseudocode of the H-DBSCAN algorithm for social event detection is given in Algorithm 1. Since the distance between samples is already stored in the adjacency matrix $DA(t)$, the worst computational complexity of the function of the query neighbors is $O(n)$. However, we employ an indexing structure to store the matrix, the computational complexity of the query function is $O(logn)$ in line 4 and 13. For evolution discovery, similar to event detection, the input parameters are the adjacency matrix $DA(t)$ based on $CE(t)$, ϵ, and $minPts$ of the event evolution task. Therefore, the overall average runtime complexity of the parallel H-DBSCAN is $O(\frac{nlogn}{p})$.

Algorithm 1 Framework of parallel H-DBSCAN algorithm for events clustering.

Input: The current streaming social data $CM(t)$; The *KIES* distance-based adjacency matrix $DA(t)$; The *KIES* distance threshold within a cluster ϵ; The minimum number of points required to form a cluster $minPts$; The number of threads p;
Output: A set of clusters.

```
for x = 1 to p in parallel do
    for any unvisited event instance e_i ∈ CM(t) do
        mark e_i as visited
        nS ← GetNeighbors(DA(t), e_i, ε)
        if |nS| < minPts then
            mark e_i as noise
        else
            C ← {e_i}
            for each event instance e_j ∈ nS do
                nS ← nS\e_j
                if e_j is not visited then
                    mark e_j as visited
                    nS' ← GetNeighbors(DA(t), e_j, ε)
                    if |nS'| ≥ minPts then
                        nS ← nS ∪ nS'
                if e_j is not yet member of any cluster then
                    C ← C ∪ {e_j}
```

The architecture of streaming social event detection and event evolution is shown in Figure 5. This architecture mainly consists of six modules, namely *data collection*, *data processing*, *similarity, indexing & storage*, *weights learning*, *event detection & evolution* and *real-time event analysis services*. For the *data collection*, a distributed crawler is developed to continuously fetch open microblogs and social network information through Weibo webpage [1]. The collected data is forwarded to processing modules through Kafka [2] to decouple their dependency. For the *data processing*, we extract both *event-oriented elements* and *event-oriented relationships* to construct streaming event-based HIN, where we store the edges into multiple dictionaries. For the *similarity, indexing & storage*, in addition to the streaming event-based HIN, it also provides meta-paths instances based similarities, meta-paths based searching, and HBase [3] and Elasticsearch [4] for data storage and full-text indexing. For the *weights learning*, a semi-supervised pairwise popularity graph convolutional network is proposed to learn the best weights of meta-paths for event detection and evolution, respectively. For the *event detection & evolution*, parallel H-DBSCAN is implemented for event detection and evolution discovery, respectively. For the *real-time event analysis services*, it presents real-time event detection and related interpretation based on event elements and event relationships, as well as real event evolution representation.

[1] https://www.weibo.com
[2] https://kafka.apache.org
[3] https://hbase.apache.org
[4] https://www.elastic.co

5 EXPERIMENTAL EVALUATION

In this section, we evaluated both effectiveness and efficiency of our proposed methods and streaming system by several real-world event detection and event evolution tasks over state-of-the-art models. What's more, we implement case study of real-world event detection and event evolution, combining *event-oriented elements*, *event-oriented relationships* and event evolutionary relationship to give more interpretation and analysis of social events.

5.1 Datasets, experimental settings and evaluation metrics

We select popular and open social media platforms Sina Weibo, a hybrid of Twitter and Facebook, and the Twitter of China and Chinese Social Media to collect richer social messages datasets. Each event instance is a non-repeating social message text. One event is a set of instance events that contain semantically identical information revolving around a real world incident. An event always has a specific time of occurrence. It may involve a group of social users, organizations, participating persons, one or several locations, other types of entities, keywords, topics, etc. One event evolution is a set of events that are incident in a sequence and describe the specific events occurring at different stages of the event. Our distributed data collection platform collects an average of 2.35 million Weibo after filtering noise and removing duplication per day. A total of 100,000 meaningful Weibo social messages are manually labeled for ground truth, and both event and event evolution labels for the collected Weibo messages are labeled by the outsourcing companies. For example, social media's tweet about *Tiger Woods winning the 2019 Masters of Golf* is an influential event in the real world and unlike *Patrick Reid's 2018 Masters of Golf.* These are two different events that happen in the real world and belong to different event categories. Note that the social users and their friend relationships involved in Sina datasets are granted by the Sina company for scientific research purposes only.

In order to learn the weight of meta-paths and implement static event and event evolution classifications. We use 10000 manually labeled Weibo social data, where 60% of the data as training set, 20% of data as development set and the remaining 20% of as test set. The number of the event classes is 5,470, and the number of event evolution classes is 5,260. We can see that the total number of classes is large and the number of samples in each class is small. In order to evaluate the effectiveness and efficiency of the proposed event-based HIN, meta-paths instances based similarity and parallel H-DBSCAN clustering for static social event detection and evolution discovery, we use social data ranging from 10,000 to 100,000. In order to evaluate the effectiveness and efficiency of the streaming social event detection and evolution discovery, we use the online 4.2 million Weibo data per day.

All of the contrast experiments are performed on three servers cluster, each with 64 core Intel Xeon CPU E5-2680 v4@2.40GHz with 512GB RAM and 4×NVIDIA Tesla P100-PICE GPUs, and an external shared configuration. The operating system and software platforms are Ubuntu 5.4.0, Tensorflow-gpu (1.4.0) and Python 2.7. The metrics used to evaluate the effectiveness of event detection and event evolution classifications are the accuracy and F1 score. The metric used to evaluate the effectiveness of event detection and evolution discovery is the normalized mutual information (NMI). For event detection and event evolution classification tasks, we train more than 7000 epochs. We then choose the weights of meth-paths with the best test performances.

5.2 Compared methods

Since it includes three parts of works, we briefly describe the text matching based static social message or social event classification methods, document similarity distances, and streaming social event detection and event evolution methods.

The following eight comparative models are the static social message or social event classification methods.

- **Support Vector Machine with TF-IDF feature (SVM)**: Support Vector Machine with pair document TF-IDF features is the most classical approach for classification tasks. In this approach, we extract the

TF-IDF features for social messages and social events, respectively, and then use the SVM classifier to implement the static event and event evolution classification.

- **Convolutional Matching Architecture-I (ARC-I)** [29]: It encodes text pairs by convolution neural networks capturing the rich matching patterns at different levels, and compares the encoded representations of each text with a multi-layer perception.
- **Convolutional Matching Architecture-II (ARC-II)** [29]: It builds directly on the interaction space between two texts, and models all the possible combinations of them with 1-D and 2-D convolution neural networks with softmax function.
- **Match by Local and Distributed Representations (DUET)** [56]: It builds text ranking model composed of two separate deep neural networks, one that matches two texts using a local representation, and another that matches the two texts using learned distributed representations. The two networks are jointly trained as part of a single neural network.
- **Multiple Positional Semantic Matching (MV-LSTM)** [78]: It matches two texts with multiple positional text representations, where each positional sentence representation is a sentence representation at this position generated by a bidirectional long short term memory. The output score is finally produced by aggregating interactions between these different positional sentence representations, through k-Max pooling and a multi-layer perceptron.
- **Convolutional Deep Structured Semantic Models (C-DSSM)** [74]: It builds a latent semantic model based on a convolutional neural network to learn low-dimensional semantic vectors for texts, where convolution-max pooling operation, local contextual information at the word n-gram level and salient local features in a word sequence are used.
- **Deep Structured Semantic Model (DSSM)** [30]: It utilizes deep neural networks to map high-dimensional sparse features into low-dimensional features, and calculates the semantic similarity of the document pair. The deep models are discriminatively trained by maximizing the conditional likelihood.
- **Siamese Encoded Graph Convolutional Network (SE-GCN)** [42]: It learns vertex representations through a Siamese neural network and aggregates the vertex features though GCNs to generate the document matching.

The following six comparatives are state-of-the-art document similarity distance.

- **Term Frequency-Inverse Document Frequency (TF-IDF)**: It uses the bag-of-words representation divided by each word's document frequency, which reflects how important a word is to a document in a text collection.
- **Latent Dirichlet Allocation (LDA)** [13]: It is a generative statistical model for text documents that learns representations for documents as distributions over word topics, and allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.
- **Marginalized Stacked Denoising Autoencoder (mSDA)** [18]: It consists of multiple stacked layers of denoising autoencoders, and is a representation learned from stacked denoting autoencoders.
- **Componential Counting Grid (CCG)** [67]: It is a generative model that models documents as a mixture of word distributions and LDA, using the counting grid embedding through window overlapping.
- **Word Move Distance (WMD)** [36]: It measures the dissimilarity between two documents as the minimum amount of distance that words of one document need to travel to reach words of another document.
- **Knowledge-driven document similarity measure (KnowSim)** [79]: It's also a meta-paths instances based document similarity, and hasn't considered the impacts of social users. The weights of meta-paths are estimated by the Laplacian scores of documents.

The following nine comparative metrics are mainstream social event detection and evolution clustering methods.

- **Hashtag-based Peak Aggregation based Event Detection (HPA-ED)** [53]: It detects events by aggregating volume peaks of hashtag over time in social streams.
- **Streaming Keyword Co-occurrence Graph based Event Detection and Event Evolution (SKCG)** [71, 72]: It builds streaming keywords co-occurrence network, and employs a betweenness centrality based clustering algorithm to detect social events.
- **Combination of Social Content and online social Behavior based Event Detection (CSCB-ED)** [59]: It combines content-based features and the propagation of news to detect social events. We implement it by treating forwarding networks as friendship networks.
- **Streaming and Bursting Abnormal Subgraph based Event Detection (SBAS-ED)** [86]: It combines trending keywords, overlapping community detection and streaming distributed processing to discover social events.
- **Inverted indices and Incremental Clustering based Event Detection (IIC-ED)** [24]: It incorporates specialized inverted indices and an incremental clustering approach to provide a low computational cost solution to detect both major and minor newsworthy events in real-time from the social data stream.
- **Content Similarity and Temporal Proximity based Event Evolution (CSTP-EE)** [58]: It combines content similarity and temporal proximity to measure the relationships between events.
- **Sketch Graphs Tracking based Event Detection and Event Evolution (eTrack)** [38]: It models the social streams as an evolving network, where each social post is a node, and edges between posts are constructed when the post similarity is above a threshold, and tracks events by extracting (k,d)-core subgraph. The sketch graph is subgraph induced by social messages' core posts and core edges.
- **Sliding Inverse Document Frequency of terms based Event Detection and Event Evolution (SIDF)** [81]: It models a simple event identification approach, which uses a sliding window model to extract events and the context of events in real-time from social message texts. This approach is based on monitoring shifts in the inverse document frequency (IDF) of terms.
- **Weighted and Locally Sensitive Hash based Event Evolution (WLSH-EE)** [48]: It integrates both linear weighting of event-elements and locally sensitive hash based distance metric to cluster the event evolution chain.

5.3 Effective evaluation of weights learning

In order to calculate the similarity between any two social instance events or events with the meth-paths based distance measure in Eq. 1, we design the PP-GCN model to learn the different weights and perform static social event classification and event evolution classification, respectively. Table I shows the testing accuracy and F1-score of different algorithms on the tasks of event classification and event evolution classification in the 2000 testing Weibo data. Overall, the proposed PP-GCN model consistently and significantly outperforms all baselines in terms of accuracy and F1. PP-GCN achieves 18%-37% improvements in terms of accuracy and F1 in event classification tasks over all baselines, and 20%-36% improvements in terms of accuracy and F1 in event classification tasks over all baselines.

The improvements can be attributed to the three characteristics of proposed models. First, the knowledgeable HIN is better modeling social events than traditional text modeling methods, such as bag-of-words (SVM), N-gram (ARC-I, ARC-II and C-DSSM) and sequence-of-words (MV-LSTM). Our PP-GCN has improved overall by more than 18% and 20% in both event detection and evolution discovery classification tasks over the SE-GCN model incorporating structural and conceptual semantics. Second, the combination of *KIES* based weighted adjacency matrix and Doc2Vec is better for fine-grained event instance or event representation learning than for feature extraction on text pairs, such as DUET. Third, the classifier based on the ratio of modulus of generated representations of event instances is better than the traditional pairwise distances. Here, we replace the regression module of the SE-GCN model by our proposed popularity based classifier, named as PP-SE-GCN, and the

Table I. Accuracy and F1 results of algorithms on the 2000 Weibo Data.

Algorithms	Event		Evolution	
	Accuracy	F1	Accuracy	F1
SVM	0.6477	0.6368	0.6654	0.6209
ARC-I [29]	0.6739	0.6693	0.6916	0.6455
ARC-II [29]	0.7018	0.6802	0.7080	0.6538
DUET [56]	0.6684	0.7016	0.6753	0.7274
MV-LSTM [78]	0.5574	0.6107	0.5985	0.6492
C-DSSM [74]	0.6703	0.6956	0.6806	0.6936
DSSM [30]	0.7039	0.7173	0.7218	0.7349
SE-GCN [42]	0.7304	0.7518	0.7475	0.7519
PP-EW-GCN	0.7038	0.7171	0.7288	0.7371
PP-SE-GCN	0.7621	0.7711	0.7737	0.7883
PA-GCN	0.8668	0.8792	0.8891	0.8945
PP-GCN	**0.9241**	**0.9334**	**0.9581**	**0.9594**

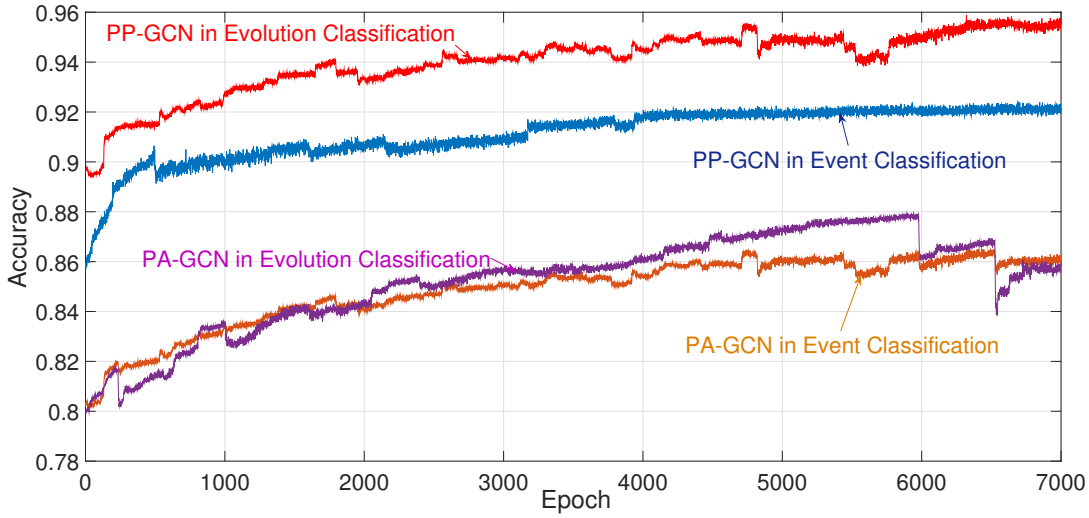

Fig. 6. Illustration of the Accuracy for PP-GCN and PA-GCN.

performances can be improved 2%-3% in the Weibo data. The 18%-20% improvements from the SE-GCN to the PP-GCN demonstrate the advantages of knowledgeable HIN modeling and the pairwise popularity based feature learning framework. Furthermore, the proposed PP-GCN model can avoid overfitting in training. We replace our classifier in PP-GCN by the discrete cosine angle of generated event instances that feature vectors based classifiers, namely PA-GCN. In Figure 6, we visualize the test accuracies of the PP-GAN and PA-GCN in both event classification and event evolution classification in 7000 epochs. Compared to the angle-based classifier, the popularity-based classifier has better ability to learn discriminating and stable event instance features and prevent overfitting.

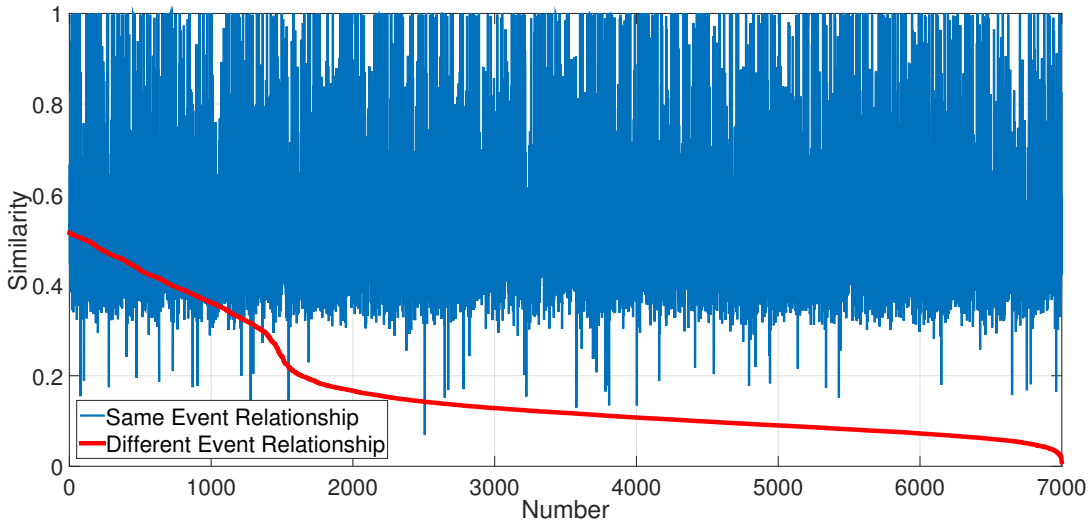

Fig. 7. Value of pairwise event similarity.

In order to verify that the learnable weights $\vec{\omega}$ are beneficial to social event representation, we test a PP-EW-GCN model with equal meta-path weight settings. Compared with the models with learnable weights, the accuracy of the PP-EW-GCN method is significantly lower, as shown in Table I. The reason is that the PP-EW-GCN model does not consider the importance of different meta-paths, that is, the importance of event elements such as different entities, keywords, topics, users, etc. Compared with the SE-GCN model, the accuracy of the PP-EW-GCN model is reduced by 2%-3%. This comparative experiment also illustrates the importance of considering different elements of social events.

There are two advantages of the proposed PP-GCN model. The one advantage is that we can perform highly accurate social event categorization. The other advantage of the proposed PP-GCN compared to other methods is that the weights $\vec{\omega}$ between the meta-paths can be learned according to the event or evolution classification tasks. So, we calculate the value of similarities between the same event class and the difference event classes, and evaluate the given values of the distance threshold ϵ in both event detection and event evolution tasks.

First, we randomly select 7000 social messages and 7000 social events, respectively. Second, we randomly select both a sample belonging to the same event and a sample belonging to different events for each sample in the above 7000 social messages to build 14000 pairs of social messages. Similarly, we randomly selected 2 pairs of samples for each of the 7000 social events to build 14000 pairs of social events. Third, we calculate the similarity distance between pairs. The values of pairwise event similarity and pairwise event evolution similarity are shown in Figure 7 and Figure 8. Here, we rank the similarities of different event or event evolution relationships from large to small. The red lines represent the value of pairwise similarities in the different relationships. The blue lines represent the value of the pairwise similarity of the same event or event evolution relationship.

In order to obtain best similarity thresholds for event and event evolution, respectively, we use an enumeration from 0 to 1 with a scale of 0.01 growth to evaluate the accuracies of event detection and evolution. Here, if the similarity between two social events is greater than the selected threshold, then the two events are considered to be the relationship of the same event or the evolution of the same event. The two accuracies under different thresholds are shown in Figure 9. We can see that the highest accuracy of event detection and event evolution can be achieved when the similarity thresholds are 0.31 and 0.20, respectively. Due to the interpretability of the

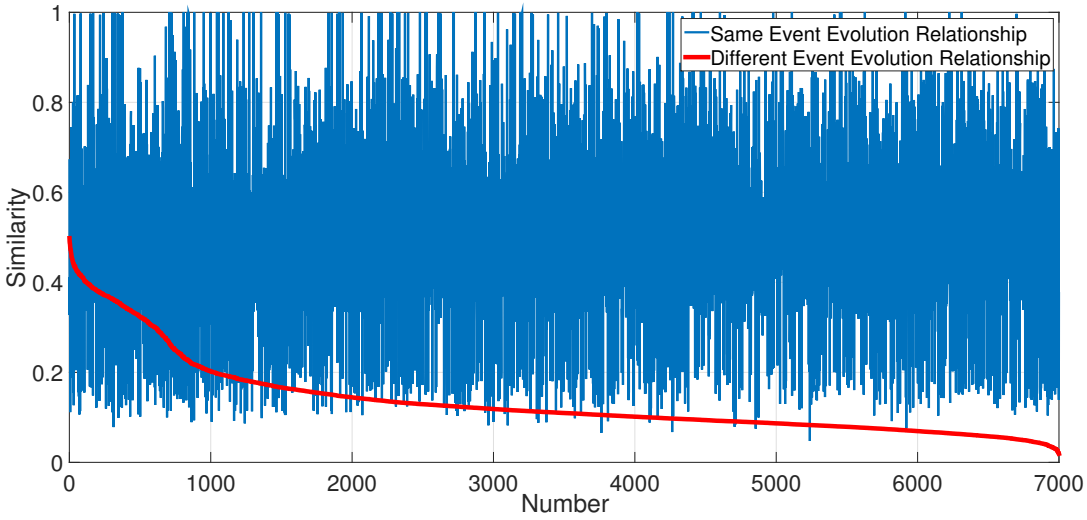

Fig. 8. Value of pairwise event evolution similarity.

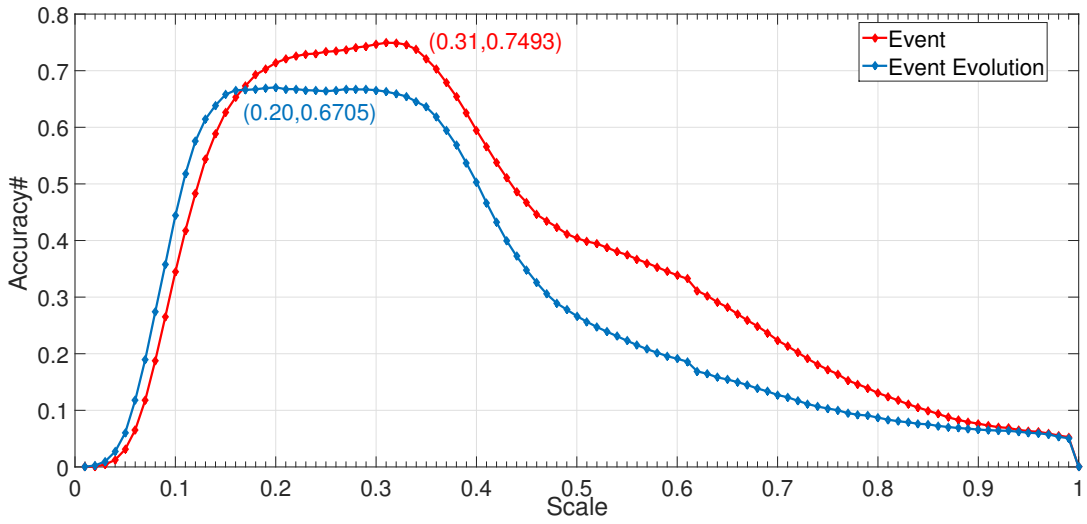

Fig. 9. Accuracy under different similarity thresholds.

meta-path and similarity measure *KIES*, the learned weights $\vec{\omega}$ can be utilized in the following event clustering based applications. Therefore, we can use 0.69 and 0.8 as the given value of the ϵ in the next H-DBSCAN based event clustering. We note that the above enumerating similarity is a simple pairwise distance test to obtain best ϵ thresholds for the H-DBSCAN based clustering experiments.

Table II. Time consumption for building different scales of event-based HINs and weighted adjacency matrixes (Minutes).

Items	10,000	20,000	30,000	40,000	50,000	60,000	70,000	80,000	90,000	100,000
EE	0.1610	0.3278	0.5082	0.7145	0.9391	1.1090	1.1793	1.3634	1.5526	1.7579
ED	0.8933	1.7416	2.5771	3.5851	4.6898	5.5140	6.0105	7.1530	7.259	8.0202
RK	0.0438	0.0727	0.0996	0.1143	0.1356	0.1529	0.1703	0.1837	0.1991	0.2026
RE	0.1442	0.1457	0.1468	0.1474	0.1485	0.1494	0.1501	0.1515	0.1520	0.1523
RKE	0.2106	0.4289	0.6692	0.8035	1.0041	1.2456	1.4829	1.6999	1.9012	2.7690
Others	0.1520	0.2946	0.4351	0.5835	0.6976	0.7855	0.8483	0.9681	1.0938	1.3942
Adjacency matrix	0.1562	0.6250	1.4062	2.504	3.9062	5.6251	7.6562	10.0137	12.6562	15.6253
Total time	1.7613	3.6364	5.8421	8.4483	11.5209	14.5815	17.4978	21.5197	24.8139	29.2655

5.4 Effectiveness and efficiency evaluation of static event detection and evolution discovery

Next, we present the event clustering to evaluate the effectiveness and efficiency of event-based HIN framework and the *KIES* distance measure in event detection and evolution discovery with the H-DBSCAN technology. Due to the interpretability of the meta-path, we use the best weights $\vec{\omega}$ trained in Section 5.3 for the calculation of *KIES* in next experiments. Another goal of this experiment is to find the best scales of the social messages and social events in one time slice in the streaming scenario. Since the average daily acquisition of non-noisy and non-repetitive microblogs on distributed data collection platform is 2.35 million, the average collection number in half an hour is about 48.9 thousands of microblogs, and the microblogs data collected within half an hour ranges from a minimum of 10.4 thousands to a maximum of 60.6 thousands. We choose the one million manually labeled Sina microblogs dataset, and randomly choose 10000, 20000, 30000, 40000, 50000, 60000, 70000, 80000, 90000 and 100000 microblogs to construct different sizes of event-based HINs.

We first build the event-based HIN, and we use one server with 64 processors and 1000 multi-threads to process social message texts. Here, we use the dictionary structure to store all elements and relationships in the event-based HINs. Second, in order to speed up the construction of the adjacency matrix, we divide the upper triangular matrix of the adjacency matrix into three triangular regions of the same size, and use three servers to simultaneously calculate the *KIES* similarity between social message texts with the learned weights of meta-paths. The detailed time consumption for constructing the different scale of event-based HINs and adjacency matrixes is shown in Table II. On the whole, the larger the social message texts, the more time it takes. We note that the relationship between total time consumption and scale of social datasets approximates a trend of super-linear growth. When constructing the event-based HIN, we report in detail the time consumption on 6 different phases, including entity extraction (EE), entity disambiguation (ED), relation between keywords (RK), relation between entities (RE), relation between keyword and entity (RKE) and others, where the most time consuming one is entity disambiguation. Because social relationships, relationships between topics, and relationships between topic and words are extracted and pre-stored, we report the time consumption of identifying topics for social message texts, keyword extraction, etc, as others in Table II.

After building the adjacency matrix, we use the parallel H-DBSCAN method, discussed in Section 4.2, to perform event detection and evolution discovery, respectively. Here, we use one server using maximum 64 processors and 1000 multi-threads to test the above parallel clustering experiments. The time consumption for event detection and evolution discovery are shown in Figure 10 and Figure 11. Blue lines represent the time-consuming using a single core, and red lines represent the time-consuming using 64 cores. The speedup of 64 processors parallel H-DBSCAN ranges from 3.493 to 6.1184. We mark the main time consumption and size of social messages in Figure 10 and Figure 11. As the average collection number in half an hour is about 48.9 thousands of microblogs, we can see that the time consumption is about 11.5209 minutes in building the

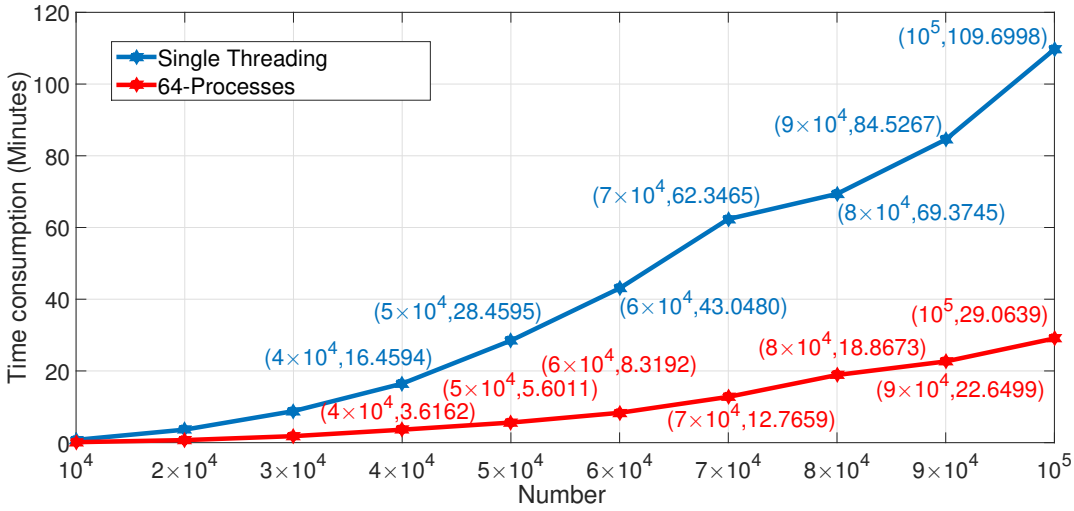

Fig. 10. Time consumption of event detection clustering for different sizes of social messages based on the parallel H-DBSCAN.

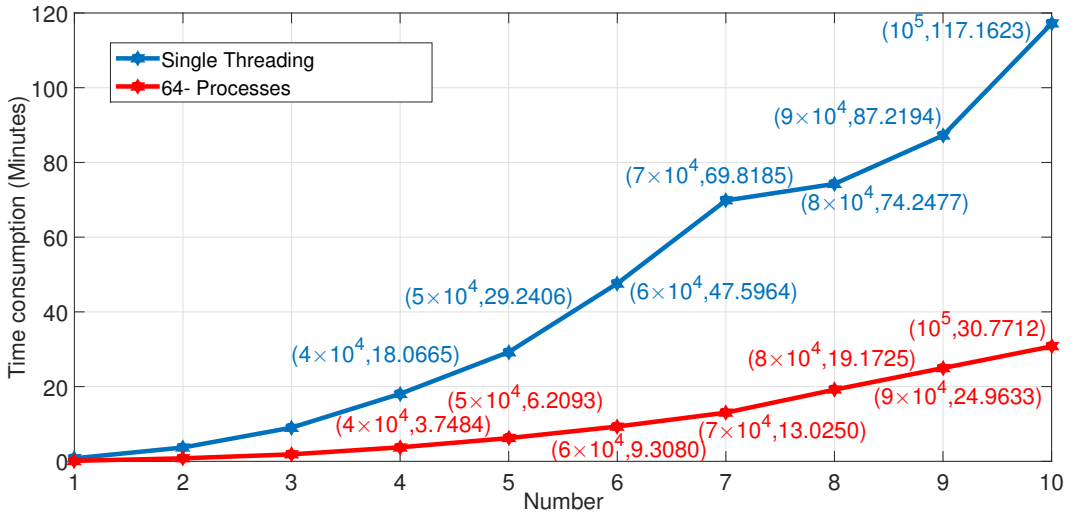

Fig. 11. Time consumption of event evolution clustering for different sizes of social events based on the parallel H-DBSCAN.

corresponding event-based HIN and weighted adjacency matrix when the number of microblogs is 5×10^4. When we add all time consumption of the above processing and event detection clustering by the H-DBSCAN, 5×10^4 social microblogs takes about 17.1219 minutes. Meanwhile, as the microblogs data collected within half an hour ranges from a minimum of 10.4 thousands to a maximum of 60.6 thousands, the time consumption ranges from about 1.3 minutes to 8.3192 minutes in the event detection clustering task. When the number of social microblogs is 5×10^4, the time consumption of the event evolution task takes 6.2093 minutes. When we add the time consumption of the above processing and evolution discovery by the H-DBSCAN, 5×10^4 social

Table III. NMI results for event detection and evolution discovery on different number of social microblogs by the proposed H-DBSCAN.

Items	Event Detection	Event Evolution
10,000	0.9365	0.9467
20,000	0.9394	0.9402
30,000	0.9432	0.9515
40,000	0.9413	0.9411
50,000	0.9337	0.9421
60,000	0.9375	0.9345
70,000	0.9479	0.9449
80,000	0.9413	0.9440
90,000	0.9344	0.9364
100,000	0.9394	0.9355

Table IV. NMI results of different similarity metrics based DBSCAN for event detection and evolution discovery on the 50,000 microblogs.

Methods	Event Detection	Event Evolution
TF-IDF	0.7002	0.6890
LDA [13]	0.7195	0.7300
mSDA [18]	0.7658	0.7538
CCG [67]	0.8172	0.8304
WMD [36]	0.8583	0.8326
KnowSim [79]	0.8531	0.8324
KIES	**0.9337**	**0.9421**

Table V. NMI results of different similarity metrics based K-means for event detection and evolution discovery on the 50,000 microblogs.

Methods	Event Detection	Event Evolution
TF-IDF	0.6536	0.6445
LDA [13]	0.6679	0.6488
mSDA [18]	0.7020	0.6837
CCG [67]	0.7307	0.7071
WMD [36]	0.7866	0.7628
KnowSim [79]	0.8079	0.7801
KIES	**0.8412**	**0.8211**

microblogs takes about 17.7302 minutes. In addition to the efficiency of the proposed event-based HIN framework and H-DBSCAN, another important evaluation is the effectiveness of event detection and evolution discovery.

On the one hand, we evaluate the performances of NMI on event detection and event evolution tasks at different numbers of social microblogs from $10,000$ to $100,000$ by the proposed H-DBSCAN. Here, we evaluate both event detection and event evolution tasks by calculating the purity of social microblogs belonging to the event and event evolution, respectively. As shown in Table III, the proposed H-DBSCAN has stable performances

Table VI. NMI results of different event detection and evolution discovery algorithms on the 50,000 microblogs.

Methods	Event Detection	Event Evolution
HPA-ED [53]	0.4196	-
CSCB-ED [59]	0.5580	-
SBAS-ED [86]	0.7121	-
IIC-ED [24]	0.4218	-
SIDF [81]	0.4079	0.5325
eTrack [38]	0.6408	0.6783
SKCG [71, 72]	0.6509	0.6917
CSTP-EE [58]	-	0.4551
WLSH-EE [48]	-	0.6935
H-DBSCAN	**0.9337**	**0.9421**

in both event detection and event evolution tasks on different numbers of social microblogs. For event detection clustering, the highest NMI is 0.9479 when the number of social microblogs is 70, 000. And for event detection clustering, the highest NMI is 0.9515 when the number of social microblogs is 30, 000. From these experiments, we can see that when the number of microblogs is larger, the more events and events evolution will be contained. We also can conclude that the interaction between events is stable in our proposed framework and algorithms. We also compare the proposed similarity measure *KIES* with six popular document similarity metrics with the same DBSCAN based event detection and evolution discovery. The experimental results are shown in Table IV, our proposed similarity measure *KIES* based DBSCAN achieves the best performances on the two clustering tasks in terms of NMI. Moreover, among the baselines, the WMD, mSDA and CGG measures have been verified to achieve state-of-the-art effects in text similarity in [36]. Although the KnowSim ignores the influence of social users and the weight given is rough, the KnowSim based DBSCAN also achieves performances of 85.31% and 83.24% in terms of NMI. Compared to other similarity measures, the proposed *KIES* based H-DBSCAN method achieves 8.06%-25.31% improvements in terms of NMI. So, the proposed event-based HIN, *KIES* and H-DBSCAN are highly scalable and effective.

In addition to DBSCAN based on density clustering, we also test the clustering detection effect of K-means under different similarity measures in Table V. We see that even if the same similarity measure is used, the clustering detection effect based on K-means is generally lower than that based on DBSCAN. For example, even if the same Word Move Distance is used, the DBSCAN method consistently improves the accuracy of NMI by 7% compared to the K-means in event detection and evolution discovery tasks. Under our proposed *KIES* measure, the DBSCAN method has about 9% advantage over the K-means method in the accuracy of NMI. This comparative experiment confirms that social event detection and evolution are more suitable for density clustering methods. The same event or the evolution tends to be cohesive, and there is a discernible distance between the semantics of different events or evolutions. In addition to accuracy, even in online social event detection and evolution discovery tasks, there is no pre-specified number of clusters.

On the other hand, as the average 48.9 thousands of microblogs in half an hour, we compare the performances of different event detection and evolution discovery algorithms on the basis of the 50, 000 social microblogs, as shown in Table VI. Since the hashtag-based peak aggregation method only considers the hashtag information to cluster social events, the HPA-ED model can capture hot events, but the NMI result is relatively low. Compared to the proposed H-DBSCAN method, the CSCB-DE model combines both content features and the propagation of social microblogs to detection events, but lacks modeling entities and relationships that are more semantically characterized. The result of the CSCB-ED model is 0.5580 in terms of NMI. The SKCG method models social

microblogs as a keywords co-occurrence graph-of-words, and then measures the similarity between microblogs by integrating both contextual and structural features to detect and track dense subgraphs. The NMI results of SKCG method are 0.6509 and 0.6917 in terms of event detection and event evolution, respectively. Similar to modeling social microblogs as a co-occurrence network, the SBAS-ED method builds the graph-of-words based on trending keywords, and detects the dense subgraphs as emergency events. Compared to the proposed H-DBSCAN method, both SKCG and SBAS-ED methods are typical isomorphic anomaly subgraph detection models. The IIC-ED model proposes a low computational cost or fast clustering method to detect social events. The result of the IIC-ED model is 0.4218 in terms of NMI. The SIDF model extracts the IDF feature for social microblogs, and uses a sliding window model to extract events. The NMI results of the SIDF method are 0.4079 and 0.5325 in terms of event detection and event evolution tasks, respectively. The eTrack method models social microblogs as evolving networks, where each social post is a node, and edges between posts are constructed when the post similarity is above a threshold. The eTrack method clusters dense subgraphs as social events, and takes temporal evolving subgraphs as event evolution. The NMI results of eTrack method are 0.6408 and 0.6783 in terms of event detection and event evolution tasks, respectively. The CSTP-EE method proposes an event similarity that integrates both words and temporal locality of stories features to capture time-ordering dependencies among events. The event evolution result of the CSTP-EE model is 0.4551 in terms of NMI. The WLSH-EE method evaluates the evolutionary relation between events by modeling a weighted linear similarity function based on defined event elements. The event evolution result of the WLSH-EE model is 0.6935 in terms of NMI.

Overall, the proposed event-HIN framework, *KIES* and H-DBSCAN algorithm consistently and significantly outperforms all baselines in terms of NMI in both event detection and event evolution tasks. In the 50, 000 social microblogs, the NMI results of H-DBSCAN achieve 52.58% and 48.70% improvements in event detection and event evolution, respectively. The above comparative experiments and time cost consumption analysis demonstrate both effectiveness and efficiency of the proposed framework and model. Next, we evaluate the performances of streaming social event detection and evolution discovery with the proposed framework and model.

5.5 Effective and efficiency evaluation of streaming event detection and evolution discovery

Different from static social event mining, streaming social event detection and evolution clustering should firstly solve the problem of event merging. For example, the tweet associated with a social event will continue to spread over the social network for a period of time as the event heats up. Therefore, we retrieve relevant social message texts from neighboring time for current social message texts.

We have modeled social messages and events as the event-based HIN, as shown in Figure 1. Therefore, we can retrieve relevant social message text on the HIN through simple meta-paths, as shown in Figure 4. Here, we also note that there are multiple instances of the above two meta-paths. We use 64 cores to retrieve relevant social message texts of different sizes ranging from 10, 000 to 50, 000, and employ the meta-paths enumerated from IEI and EIMIE, respectively, for event detection and event evolution tasks. We remove duplicate social microblogs. The time consumption for the meta-paths based relevant messages or events searching is shown in Table VII. We can see that time consumption is acceptable and relatively short. When we retrieve 50, 000 unique social messages or events, it takes 1.7712 or 1.9690 minutes.

Table VII. Time consumption for different scales of social messages searching on the event-based HIN (Minutes).

Items	10,000	20,000	30,000	40,000	50,000
Metapath: IEI	0.3052	0.635	0.989	1.3253	1.7712
Metapath: EIMIE	0.3415	0.694	1.1755	1.5875	1.9690

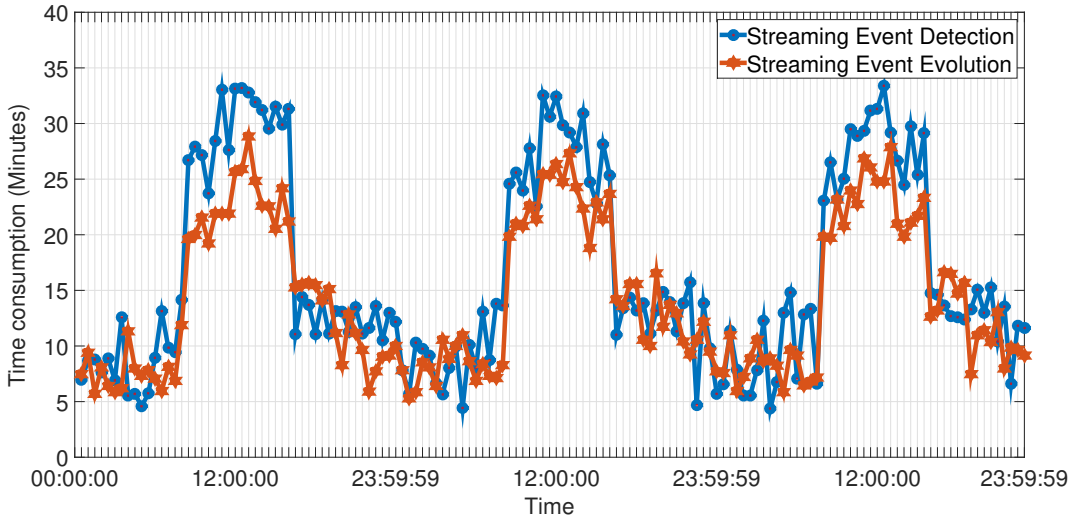

Fig. 12. The time consumption of streaming event detection and event evolution within 3 days.

In the streaming scenario, we also construct the streaming event-HIN while collecting social messages, and perform the H-DBSCAN based event detection and event evolution for half an hour, as shown in Figure 5. Although the average collection number in half an hour is about 48.9 thousands of non-noisy and non-repetitive microblogs, and the microblogs data collected within half an hour ranges from a minimum of 10.4 thousands to a maximum of 60.6 thousands. We also retrieve the similar microblogs on the neighboring time slice for streaming microblogs to build the temporal event-based HIN, where more than a week of information is stored in the external storage space. Then, we performed H-DBSCAN based event detection and evolution discovery, respectively, on two servers. The time consumption of streaming event detection and event evolution within 3 days are shown in Figure 12. We find that the number of microblogs collected and retrieved at each time changes over time. In the overall trend, we observe that the number of postings by social users during the day is relatively large, and the number of postings at night is small. For streaming event detection, the maximum time consumption for each time slice is about 33.15 minutes. For streaming event evolution, the maximum time consumption for each time slice is about 28.76 minutes. Different from the static event detection and event evolution, since the number of events is generally less than microblogs in one time slice, and part of event-based HIN has built in the event detection, the average of time consumption of streaming event evolution is less than the average of time consumption of streaming event detection. Overall, our proposed streaming event-based framework and H-DBSCAN algorithm can meet real-time requirements.

In addition to the real-time requirements of streaming computing, we also mix a random number of manually annotated social microblogs into the streaming social data to measure the NMI of streaming event detection and evolution discovery. The results of the event detection and event evolution are shown in Figure 13. In the streaming experiments, the performances are fluctuating, but the average NMI results are about 0.9233 and 0.9275. For the streaming event detection task, the highest NMI is 0.9539, and the lowest NMI is 0.8796. Meanwhile, for the streaming event evolution task, the highest NMI is 0.9670, and the lowest NMI is 0.8829. The reason is that the incompleteness of the mixed annotation manually annotated social microblogs affects the number of points required to form a cluster in the H-DBSCAN. Overall, the performances of event detection and evolution discovery are satisfactory for building streaming event detection and evolution systems.

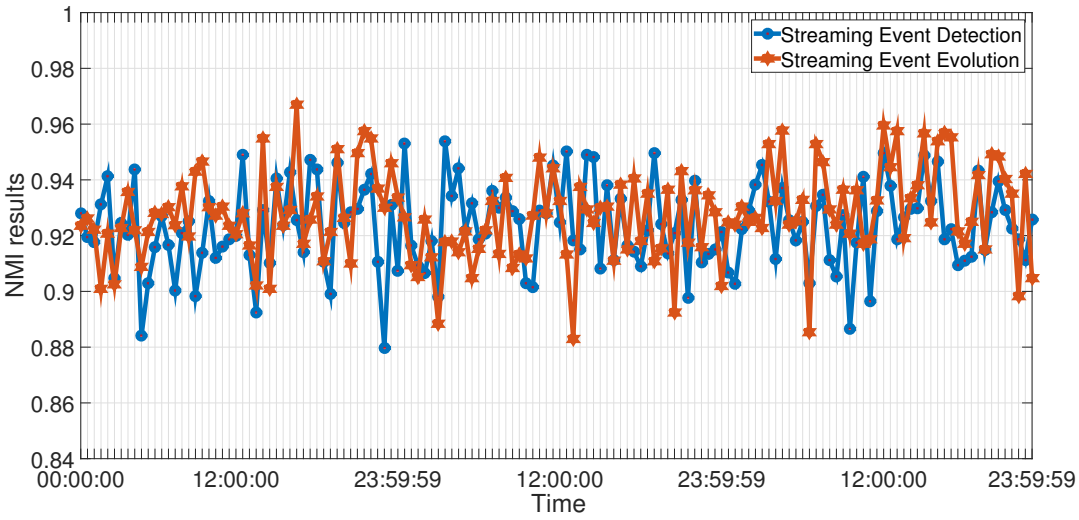

Fig. 13. The NMI results of streaming Event detection and event evolution within 3 days.

With the effectiveness and efficiency, the proposed event-based HIN, *KIES*, PP-GCN and H-DBSCAN solve the problems of building reliable and open domain streaming event detection and event evolution systems in practice. Here, we give a case study of the events of "Notre Dame de Paris". From the real-world social microblogs, we give the related social messages, where we describe it with significant parts and represent some unimportant parts as ellipses. As happened in the real-time, from social messages, we detect that the entire case consisted of three phases of events, 2019/04/15 18:50~2019/04/16 09:36 Paris' Notre Dame Cathedral on fire, 2019/04/15 23:15~2019/04/21 11:35 Notre Dame fire is extinguished, and 2019/04/16 07:45 ~ 2019/04/18 09:23 Reconstruction of Notre Dame de Paris. So, we give parts of event elements in these events. Both main participants and concerned entities are different in different events throughout the case. However, all three events are caused by the event of *Paris' Notre Dame Cathedral on fire*. Obviously, there is an evolutionary relationship between the three events. More cases can be viewed at our developed event detection and evolution discovery system[5].

6 RELATED WORKS

In this section, we will briefly review the related work about social events and HIN. Social event detection and evolution models can be roughly categorized into topic detection and tracking models, social media event frames extraction, abnormality detection based event detection.

Topic detection and tracking (TDT) originally described the detection of topics in news streams [4]. At that time, the detected topics were termed "events" and detecting and tracking them was considered a classification or clustering problem. Bag-of-words (BoW) representation is commonly used to represent a news article and do machine learning in the vector space defined by the BoW's features. Later on, probabilistic latent semantic analysis (PLSA) [27], latent Dirichlet allocation (LDA) [13], and other techniques came to be used. The assumption that one news article contains only one topic was relaxed and that improved topic detection results. Overall, these early approaches mostly assumed that the features of the news or events were just words. They did not consider entities outside the article and their relationships. Unlike TDT, social event extraction [1, 43, 52] uses frame-based event definitions applying the well-defined techniques for extracting event frames from news in natural language

[5]http://ring.act.buaa.edu.cn

Table VIII. Case Study.

Related Social Messages	Event Detection	Event Evolution
2019/04/15 Watch: Notre Dame Cathedral in Paris on fire. · · ·	Event-1: 2019/04/15 18:50~2019/04/16 09:36 Paris' Notre Dame Cathedral on fire.	
2019/04/15 Notre Dame fire: Live updates.	Time : 2019/04/15 18:50	
2019/04/15 400 firefighters mobilized for Notre Dame blaze.	Location : Notre Dame Cathedral, Paris, French	
2019/04/15 Around the world, 'our hearts ache' at Notre Dame Cathedral fire.	Topics : Fire disaster, Culture, Religion	2019/04/15 18:50~2019/04/16 09:36 Paris' Notre Dame Cathedral on fire.
2019/04/16 The church is burning and the whole world is crying - Parisians mourn for Notre-Dame.	Entities : Notre Dame Cathedral, French President, Paris Fire Department, Rose Windows, Ministry of Culture, Pipe Organs · · ·	
· · ·		
2019/04/15 Notre Dame cathedral fire is fully extinguished.	Event-2: 2019/04/15 23:15~2019/04/21 11:35 Notre Dame fire is extinguished.	
2019/04/16 Why the Notre Dame fire was so hard to put out.	Time : 2019/04/15 23:15	
2019/04/16 The fire is out': Paris firefighters succeed after 12-hour battle to extinguish Notre Dame Cathedral blaze.	Keywords : Fire, Paris, Notre-Dame, Firefighter, Flame, Water, Helicopter, Tower, Heat, Stair, Edifice, Risk, Brigade, Police	2019/04/15 23:15~2019/04/21 11:35 Notre Dame fire is extinguished.
2019/04/17 Paris firefighters got on Notre-Dame site in less than 10 minutes.	Entities : Notre Dame Cathedral, Deluge Guns, Thermal Imaging, Bell Tower · · ·	
2019/04/21 Paris Easter Mass honors firefighters who saved Notre Dame.	Topics : Fire disaster, Culture, Religion	
· · ·		
2019/04/16 800 million euros in donations for the reconstruction of Notre-Dame.	Event-3: 2019/04/16 07:45 ~ 2019/04/18 09:23 Reconstruction of Notre Dame de Paris.	
2019/04/16 Macron said reconstruction of the Cathedral.	Time : 2019/04/16 07:45	
2019/04/16 Companies and large fortunes mobilize for the reconstruction of Notre-Dame.	Entities : Fondation du Patrimoine, Palace of Westminster, 2024 Olympic Games, French prime minister, Arnault Family, Total SA,	2019/04/16 07:45 ~ 2019/04/18 09:23 Reconstruction of Notre Dame de Paris.
2019/04/17 Pledges Reach Almost 1 Billion To Rebuild Paris' Notre Dame Cathedral.	BPCE, Bettencourt Family, Pinault Family, AXA SA, Paris City Government · · ·	
2019/04/18 Opinion: The reconstruction of Notre-Dame is not the only answer.	Keywords : rebuild, organisation, insurance, millions, euros, cathedral, architecture · · ·	
· · ·		

processing. Frame-based event extraction can extract entities and their relationships, but uses only a limited number of event types. Moreover, it uses complicated machine learning models, usually a pipeline of them, to

incorporate different levels of annotation and features. However, social messages are invariably short and usually noisy. That calls for more flexibility to incorporate simpler but more reliable features and links. Sometimes event detection is treated as abnormal detection [17] or document clustering [41] in streaming data. Streaming data can be of different types. Here we will only review some graph-based problems. Streaming's abnormally connected subgraph structure (under different names such as k-clique, motifs or graphlets) based social event detection has been studied [2, 6, 86]. A sharp rise in keyword occurrences [44, 45, 85] in a specified time slice is an important characteristic of events shown up in social media, which generates statistically-significant subgraphs or patterns. Subgraph analysis can potentially help to partition graphs better in graph-of-words document representation [70] when not only considering hot words in the stream but also word co-occurrence patterns. These approaches have been applied to homogeneous graphs but not HINs.

A HIN is a graph of entities with multiple typed entities and relations [25, 76]. They were originally developed to analyze scientific publication networks [77], and in social network analysis they can also represent user similarity and predict links [32]. Similar to our work, Wang et al [80] integrated information from external knowledge base to help modeling rich textual HIN, and showed that it can be very useful for categorizing documents. HIN can also be used in time-evolving clustering tasks [84]. There have been some studies of using HIN to analyze events. Gui et al [23] focused on a particular type of action (publishing a paper) as an event. Thus using HIN to fully model events' elements (such as keywords, entities, hashtags, location, time) and heterogeneous social networks (with different types of nodes and different kinds of social relations) in event mining problems has not yet received much scholarly attention.

7 CONCLUSION

We study the problem of streaming social event detection and evolution discovery in the microblogs streams. We first propose an event-based HIN framework which integrates event elements and their relations, in a semantically meaningful way, and calculates the similarity between any two events to model social messages. Secondly, we propose the PP-GCN model that achieves state-of-the-art results in fine-grained event categorization, and learns the best weights of meta-paths in different tasks. Through the PP-GCN model, we are able to overcome the problems of large category size and sparse small number of samples per class and prevent overfitting in our tasks. Third, we propose the H-DBSCAN clustering method which achieves state-of-the-art results in both static and streaming event detection and evolution discovery tasks. Our extensive experiments have demonstrated that the event-based HIN framework and H-DBSCAN clustering are highly effective and efficient. In the future, it is interesting to extend the streaming event detection and evolution framework for handling multilingual social streams and predicting future social events and its influences.

ACKNOWLEDGMENT

The authors of this paper were supported by the NSFC through grants 62002007 and U20B2053, the Key Research and Development Project of Hebei Province through grant 20310101D, NSF of Guangdong Province through grant 2017A030313339, Hong Kong RGC including Early Career Scheme (ECS, No. 26206717), General Research Fund (GRF, No. 16211520) and Research Impact Fund (RIF, No. R6020-19), State Key Laboratory of Software Development Environment (SKLSDE-2020ZX-12), the UK EPSRC (EP/T01461X/1, EP/T021985/1 and EP/T022582/1), NSF ONR N00014-18-1-2009, NSF under grants III-1763325, III-1909323, and SaTC-1930941. This work was also sponsored by CAAI-Huawei MindSpore Open Fund. Thanks for computing infrastructure provided by Huawei MindSpore platform.

REFERENCES

[1] Apoorv Agarwal and Owen Rambow. 2010. Automatic Detection and Classification of Social Events. In *Proceedings of the EMNLP*. Association for Computational Linguistics, 1024–1034.

[2] Charu C Aggarwal and Karthik Subbian. 2012. Event detection in social streams. In *Proceedings of the SDM*. SIAM, 624–635.
[3] Rutuja Ahirrao and Sachin Patel. 2013. An overview on event evolution technique. *International Journal of Computer Applications* 77, 10 (2013).
[4] James Allan. 2012. *Topic Detection and Tracking: Event-based Information Organization*. Springer Publishing Company, Incorporated.
[5] James Allan, Ron Papka, and Victor Lavrenko. 1998. On-line new event detection and tracking. In *Proceedings of the SIGIR*. 37–45.
[6] Albert Angel, Nick Koudas, Nikos Sarkas, Divesh Srivastava, Michael Svendsen, and Srikanta Tirthapura. 2014. Dense subgraph maintenance under streaming edge weight updates for real-time story identification. *The VLDB journal* 23, 2 (2014), 175–199.
[7] Sitaram Asur and Bernardo A Huberman. 2010. Predicting the future with social media. In *Proceedings of the WI-IAT*. IEEE/WIC/ACM, 492–499.
[8] Farzindar Atefeh and Wael Khreich. 2015. A survey of techniques for event detection in twitter. *Computational Intelligence* 31, 1 (2015), 132–164.
[9] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.
[10] Iyad Batal, Dmitriy Fradkin, James Harrison, Fabian Moerchen, and Milos Hauskrecht. 2012. Mining recent temporal patterns for event detection in multivariate time series data. In *Proceedings of the SIGKDD*. ACM, 280–288.
[11] Hila Becker and Luis Gravano. 2011. *Identification and characterization of events in social media*. Columbia University.
[12] David M Blei, Thomas L Griffiths, and Michael I Jordan. 2010. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM (JACM)* 57, 2 (2010), 1–30.
[13] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.
[14] Hans-Peter Blossfeld, Alfred Hamerle, and Karl Ulrich Mayer. 2014. *Event history analysis: Statistical theory and application in the social sciences*. Psychology Press.
[15] Carlo Vittorio Cannistraci, Gregorio Alanis-Lobato, and Timothy Ravasi. 2013. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific reports* 3 (2013), 1613.
[16] Yuwei Cao, Hao Peng, Jia Wu, Yingtong Dou, Jianxin Li, and Philip S. Yu. 2021. Knowledge-Preserving Incremental Social Event Detection via Heterogeneous GNNs. (2021). arXiv:cs.LG/2101.08747
[17] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 1–58.
[18] Minmin Chen, Zhixiang Xu, Kilian Q Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the ICML*. 1627–1634.
[19] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the NIPS*. 3844–3852.
[20] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the SIGKDD*. 226–231.
[21] Donald Getz. 2008. Event tourism: Definition, evolution, and research. *Tourism management* 29, 3 (2008), 403–428.
[22] Thomas L Griffiths, Michael I Jordan, Joshua B Tenenbaum, and David M Blei. 2004. Hierarchical topic models and the nested Chinese restaurant process. In *Proceedings of the NIPS*. 17–24.
[23] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, and Jiawei Han. 2016. Large-scale embedding learning in heterogeneous event data. In *Proceedings of the ICDM*. IEEE, 907–912.
[24] Mahmud Hasan, Mehmet A Orgun, and Rolf Schwitter. 2019. Real-time event detection from the Twitter data stream using the TwitterNews+ Framework. *Information Processing & Management* 56, 3 (2019), 1146–1165.
[25] Yu He, Yangqiu Song, Jianxin Li, Cheng Ji, Jian Peng, and Hao Peng. 2019. HeteSpaceyWalk: a heterogeneous spacey random walk for heterogeneous information network embedding. In *Proceedings of the CIKM*. ACM, 639–648.
[26] Daniel Hienert, Dennis Wegener, and Heiko Paulheim. 2012. Automatic classification and relationship extraction for multi-lingual and multi-granular events from wikipedia. *Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2012)* 902 (2012), 1–10.
[27] Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of the UAI*. Morgan Kaufmann Publishers Inc., 289–296.
[28] Yu Hong, Tongtao Zhang, Tim O'Gorman, Sharone Horowit-Hendler, Heng Ji, and Martha Palmer. 2016. Building a cross-document event-event relation corpus. In *Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016)*. Association for Computational Linguistics, 1–6.
[29] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of the NIPS*. 2042–2050.
[30] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the CIKM*. ACM, 2333–2338.

[31] Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL*. Association for Computational Linguistics, 254–262.
[32] Zhang Jiawei, Kong Xiangnan, and Yu Philip S. 2013. Predicting Social Links for New Users across Aligned Heterogeneous Social Networks. In *Proceedings of the ICDM*. IEEE, 1289–1294.
[33] Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and JunâĂŹichi Tsujii. 2009. Overview of BioNLPâĂŹ09 shared task on event extraction. In *Proceedings of the BioNLP 2009 workshop companion volume for shared task*. 1–9.
[34] Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. 2011. Overview of genia event task in bionlp shared task 2011. In *Proceedings of the BioNLP Shared Task 2011 Workshop*. Association for Computational Linguistics, 7–15.
[35] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the ICLR*.
[36] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the ICML*. 957–966.
[37] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the ICML*. 1188–1196.
[38] Pei Lee, Laks VS Lakshmanan, and Evangelos E Milios. 2013. Event evolution tracking from streaming social posts. *arXiv preprint arXiv:1311.5978* (2013).
[39] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.
[40] Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the ACL*, Vol. 1. 73–82.
[41] Bang Liu, Fred X Han, Di Niu, Linglong Kong, Kunfeng Lai, and Yu Xu. 2020. Story Forest: Extracting Events and Telling Stories from Breaking News. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 14, 3 (2020), 1–28.
[42] Bang Liu, Ting Zhang, Di Niu, Jinghong Lin, Kunfeng Lai, and Yu Xu. 2018. Matching Long Text Documents via Graph Convolutional Networks. *arXiv* (2018).
[43] Xiaohua Liu, Xiangyang Zhou, Zhongyang Fu, Furu Wei, and Ming Zhou. 2012. Exacting Social Events for Tweets Using a Factor Graph. In *Proceedings of the AAAI*. AAAI Press, 1692–1698.
[44] Yaopeng Liu, Hao Peng, Jie Guo, Tao He, Xiong Li, Yangqiu Song, Jianxin Li, et al. 2018. Event detection and evolution based on knowledge base. In *Proc. KBCOM*.
[45] Yaopeng Liu, Hao Peng, Jianxin Li, Yangqiu Song, and Xiong Li. 2020. Event detection and evolution in multi-lingual social streams. *Frontiers of Computer Science* 14, 5 (2020), 1–15.
[46] Zhiwei Liu, Yang Yang, Zi Huang, Fumin Shen, Dongxiang Zhang, and Heng Tao Shen. 2019. Embedding and predicting the event at early stage. *World Wide Web* 22, 3 (2019), 1055–1074.
[47] Rui Long, Haofen Wang, Yuqiang Chen, Ou Jin, and Yong Yu. 2011. Towards effective event detection, tracking and summarization on microblog data. In *Proceedings of the WAIM*. Springer, 652–663.
[48] Zhongyu Lu, Weiren Yu, Richong Zhang, Jianxin Li, and Hua Wei. 2015. Discovering event evolution chain in microblog. In *Proceedings of the HPCC*. IEEE, 635–640.
[49] Ning Ma and Yijun Liu. 2014. SuperedgeRank algorithm and its application in identifying opinion leader of online public opinion supernetwork. *Expert Systems with Applications* 41, 4 (2014), 1357–1368.
[50] Augusto Q Macedo, Leandro B Marinho, and Rodrygo LT Santos. 2015. Context-aware event recommendation in event-based social networks. In *Proceedings of the RecSys*. ACM, 123–130.
[51] Juha Makkonen. 2003. Investigations on event evolution in TDT. In *Proceedings of the NAACL*. Association for Computational Linguistics, 43–48.
[52] Qianren Mao, Xi Li, Hao Peng, Jianxin Li, Dongxiao He, Shu Guo, Min He, and Lihong Wang. 2021. Event prediction based on evolutionary event ontology knowledge. *Future Generation Computer Systems* 115 (2021), 76–89.
[53] Adam Marcus, Michael S Bernstein, Osama Badar, David R Karger, Samuel Madden, and Robert C Miller. 2011. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the SIGCHI*. ACM, 227–236.
[54] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[55] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the NIPS*. 3111–3119.
[56] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In *Proceedings of the WWW*. ACM, 1291–1299.
[57] Masaki Mori, Takao Miura, and Isamu Shioya. 2006. Topic detection and tracking for news web pages. In *Proceedings of the 2006 IEEE/WIC/ACM international conference on web intelligence*. IEEE Computer Society, 338–342.
[58] Ramesh Nallapati, Ao Feng, Fuchun Peng, and James Allan. 2004. Event threading within news topics. In *Proceedings of the CIKM*. ACM, 446–453.

[59] Duc T. Nguyen and Jai E. Jung. 2017. Real-time event detection for online behavioral analysis of big social data. *Future Generation Computer Systems* 66 (2017), 137–145.
[60] Geoff O'Brien, Phil O'Keefe, Zaina Gadema, and Jon Swords. 2010. Approaching disaster management through social learning. *Disaster Prevention and Management: An International Journal* 19, 4 (2010), 498–508.
[61] Yukio Ohsawa, Nels E Benson, and Masahiko Yachida. 1998. KeyGraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *Proceedings of the ADL*. IEEE, 12–18.
[62] Miles Osborne, Sean Moran, Richard McCreadie, Alexander Von Lunen, Martin Sykora, Elizabeth Cano, Neil Ireson, Craig Macdonald, Iadh Ounis, Yulan He, et al. 2014. Real-time detection, tracking, and monitoring of automatically discovered events in social media. In *Proceedings of the ACL*. Association for Computational Linguistics, 37–42.
[63] Fragkiskos Papadopoulos, Maksim Kitsak, M Ángeles Serrano, Marián Boguná, and Dmitri Krioukov. 2012. Popularity versus similarity in growing networks. *Nature* 489, 7417 (2012), 537.
[64] Symeon Papadopoulos, Yiannis Kompatsiaris, Athena Vakali, and Ploutarchos Spyridonos. 2012. Community detection in social media. *Data Mining and Knowledge Discovery* 24, 3 (2012), 515–554.
[65] Mostofa Ali Patwary, Diana Palsetia, Ankit Agrawal, Wei-keng Liao, Fredrik Manne, and Alok Choudhary. 2012. A new scalable parallel DBSCAN algorithm using the disjoint-set data structure. In *Proceedings of the SC*. IEEE, 1–11.
[66] Hao Peng, Jianxin Li, Qiran Gong, Yangqiu Song, Kunfeng Lai, and Philip S. Yu. 2019. Fine-grained Event Categorization with Heterogeneous Graph Convolutional Networks. In *Proceedings of the IJCAI*. AAAI Press.
[67] Alessandro Perina, Nebojsa Jojic, Manuele Bicego, and Andrzej Truski. 2013. Documents as multiple overlapping windows into grids of counts. In *Proceedings of the NIPS*, Vol. 26. 10–18.
[68] Kira Radinsky and Eric Horvitz. 2013. Mining the Web to Predict Future Events. In *Proceedings of the WSDM*. ACM, 255–264.
[69] Alan Ritter, Oren Etzioni, Sam Clark, et al. 2012. Open domain event extraction from twitter. In *Proceedings of the SIGKDD*. ACM, 1104–1112.
[70] François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. 2015. Text categorization as a graph classification problem. In *Proceedings of the ACL*, Vol. 1. 1702–1712.
[71] Hassan Sayyadi, Matthew Hurst, and Alexey Maykov. 2009. Event detection and tracking in social streams. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 3.
[72] Hassan Sayyadi and Louiqa Raschid. 2013. A graph analytical approach for topic detection. *ACM Transactions on Internet Technology (TOIT)* 13, 2 (2013), 4.
[73] Fariya Sharmeen, Theo Arentze, and Harry Timmermans. 2015. Predicting the evolution of social networks with life cycle events. *Transportation* 42, 5 (2015), 733–751.
[74] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the WWW*. ACM, 373–374.
[75] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip.S Yu. 2017. A survey of heterogeneous information network analysis. *TKDE* (2017), 17–37.
[76] Yizhou Sun and Jiawei Han. 2012. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery* 3, 2 (2012), 1–159.
[77] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.
[78] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A deep architecture for semantic matching with multiple positional sentence representations. In *Proceedings of the AAAI*, Vol. 30.
[79] Chenguang Wang, Yangqiu Song, Haoran Li, Ming Zhang, and Jiawei Han. 2016. Text classification with heterogeneous information network kernels. In *Proceedings of the AAAI*, Vol. 30.
[80] Chenguang Wang, Yangqiu Song, Haoran Li, Ming Zhang, and Jiawei Han. 2018. Unsupervised meta-path selection for text similarity measure based on heterogeneous information networks. *Data Mining and Knowledge Discovery* 32, 6 (2018), 1735–1767.
[81] Andreas Weiler, Michael Grossniklaus, and Marc H Scholl. 2014. Event identification and tracking in social media streaming data. In *EDBT/ICDT*. 282–287.
[82] Lilian Weng, Jacob Ratkiewicz, Nicola Perra, Bruno Gonçalves, Carlos Castillo, Francesco Bonchi, Rossano Schifanella, Filippo Menczer, and Alessandro Flammini. 2013. The role of information diffusion in the evolution of social networks. In *Proceedings of the SIGKDD*. ACM, 356–364.
[83] Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. 2017. CN-DBpedia: A never-ending Chinese knowledge extraction system. In *Proceedings of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 428–438.
[84] Sun Yizhou, Tang Jie, Han Jiawei, Chen Cheng, and Gupta Manish. 2014. Co-Evolution of Multi-Typed Objects in Dynamic Star Networks. *IEEE Transactions on Knowledge and Data Engineering* 26, 12 (2014), 2942–2955.

[85] Weiren Yu, Charu C Aggarwal, Shuai Ma, and Haixun Wang. 2013. On anomalous hotspot discovery in graph streams. In *ICDM*. IEEE, 1271–1276.
[86] W Yu, J Li, ZA Bhuiyan, R Zhang, and J Huai. 2019. Ring: Real-Time Emerging Anomaly Monitoring System Over Text Streams. *IEEE Annals of the History of Computing* 04 (2019), 506–519.
[87] Jeffrey M Zacks and Barbara Tversky. 2001. Event structure in perception and conception. *Psychological bulletin* 127, 1 (2001), 3.
[88] Xiangmin Zhou and Lei Chen. 2014. Event detection over twitter social media streams. *The VLDB Journal* 23, 3 (2014), 381–400.

Received December 2019; revised September 2020; accepted January 2021