Hierarchical and Incremental Structural Entropy Minimization for Unsupervised Social Event Detection

Yuwei Cao[1], Hao Peng[2], Zhengtao Yu[3], Philip S. Yu[1]

[1] Department of Computer Science, University of Illinois Chicago, Chicago, USA
[2] School of Cyber Science and Technology, Beihang University, Beijing, China
[3] Yunnan Key Laboratory of Artificial Intelligence, Kunming University of Science and Technology, Kunming, China
{ycao43, psyu}@uic.edu, penghao@buaa.edu.cn, ztyu@hotmail.com

Abstract

As a trending approach for social event detection, graph neural network (GNN)-based methods enable a fusion of natural language semantics and the complex social network structural information, thus showing SOTA performance. However, GNN-based methods can miss useful message correlations. Moreover, they require manual labeling for training and predetermining the number of events for prediction. In this work, we address social event detection via graph structural entropy (SE) minimization. While keeping the merits of the GNN-based methods, the proposed framework, HISEvent, constructs more informative message graphs, is unsupervised, and does not require the number of events given a priori. Specifically, we incrementally explore the graph neighborhoods using 1-dimensional (1D) SE minimization to supplement the existing message graph with edges between semantically related messages. We then detect events from the message graph by hierarchically minimizing 2-dimensional (2D) SE. Our proposed 1D and 2D SE minimization algorithms are customized for social event detection and effectively tackle the efficiency problem of the existing SE minimization algorithms. Extensive experiments show that HISEvent consistently outperforms GNN-based methods and achieves the new SOTA for social event detection under both closed- and open-set settings while being efficient and robust.

Introduction

Social event detection serves as a foundation for public opinion mining (Beck et al. 2021), fake news detection (Mehta, Pacheco, and Goldwasser 2022), etc., and is attracting increasing attention in industry and academia. Existing studies (Ren et al. 2022a; Cao et al. 2021; Liu et al. 2020a; Peng et al. 2019, 2022) commonly formalize the task of social event detection as extracting clusters of co-related messages from sequences of social media messages.

Recent years have witnessed the booming of social event detection studies (Ren et al. 2023, 2022a; Peng et al. 2022; Cao et al. 2021; Peng et al. 2019) that are based on Graph Neural Networks (GNN) (Kipf and Welling 2017; Veličković et al. 2018; Hamilton, Ying, and Leskovec 2017). These methods typically follow a two-step strategy: they first construct message graphs that contain all the candidate messages, with ones that share common attributes (user

mentions, hashtags, named entities, etc.) linked together. Figure 1A.2 shows an example message graph. They then partition the message graph using GNNs, which incorporate the natural language representations of the messages with that of their neighbors. The resulting graph partitions (e.g., Figure 1B.3) serve as the detected social events. Despite their SOTA performance, GNN-based methods merely link messages that share exactly the same attributes. The useful correlations between messages that are semantically close yet have no common attributes are missing. Furthermore, the GNN components of these models require supervision for training and predetermining the total number of events for prediction. Recent GNN-based methods (Ren et al. 2022a; Peng et al. 2022; Cao et al. 2021), unlike earlier ones (Peng et al. 2019), adopt contrastive learning, inductive learning, and pseudo label generation to alleviate the reliance on labels. However, manual labeling is still necessary for the initial training and periodical maintenance.

In this work, we address the above issues from an information-theoretic perspective. We gain inspiration from *structural entropy* (SE) (Li and Pan 2016), a metric that assesses the amount of information contained in a graph. Specifically, minimizing one-dimensional (1D) SE discloses the reliable node correlations contained in the raw, noisy graphs and is applied in biomedical studies (Li, Yin, and Pan 2016). We explore message graph neighborhoods via 1D SE minimization and supplement the existing message graph with edges between the semantically close messages. Unlike previous studies (Li, Yin, and Pan 2016; Li et al. 2018), our exploration is conducted in an incremental manner to maximize efficiency. Minimizing higher-dimensional SE decrypts the higher-order structure of the graphs (Li and Pan 2016). Given this, we further partition the message graph via two-dimensional (2D) SE minimization. Though effective and requiring no supervision, 2D SE minimization can be prohibitively slow to perform on complex, large-scale message graphs. We effectively tackle this by customizing a 2D SE minimization algorithm for social event detection. Our algorithm addresses the message correlations in a hierarchical manner: it repeatedly splits the message graph, detects clusters, and combines the clusters into new ones while keeping the previously detected partitions. Our proposed framework, hierarchical and incremental structural entropy minimization-guided social event detector (HISEv-
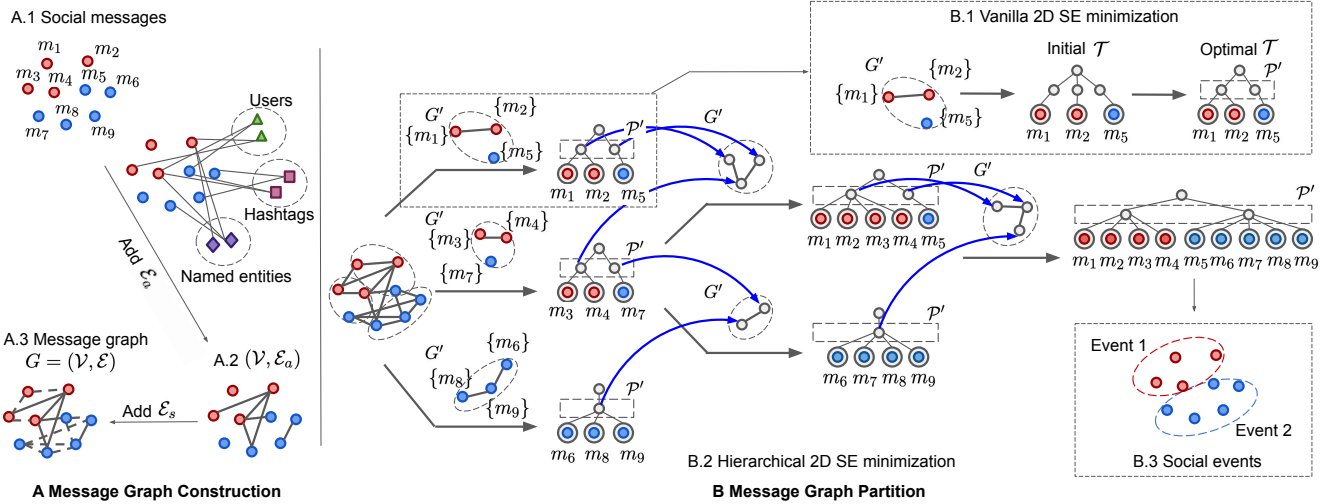
Figure 1: The proposed HISEvent framework. A and B are message graph construction and partitioning processes, respectively. An initial message graph A.2 is constructed by linking social messages (A.1) that share common attributes. Further adding semantic-similarity-based edge set $\mathcal{E}_s$ results in the final message graph (A.3). B.2 shows our proposed hierarchical 2D SE minimization algorithm, which repeatedly detects clusters ($\mathcal{P}'$) from sub-graphs (G'). B.1 shows how clusters are detected in a single sub-graph via vanilla 2D SE minimization. B.3 shows the detected social events.

ent), holds the merits of the GNN-based methods, learns more informative message graphs, and does not require supervision or the number of events given a priori. Experiments on two public Twitter datasets show that HISEvent consistently outperforms strong baselines under both closed- and open-set settings and is the new SOTA for social event detection. We also empirically show the efficiency and robustness of HISEvent as well as the effectiveness of all its components. Our contributions are:

- We address social event detection from an information-theoretic lens. Compared to the GNN-based methods, the proposed HISEvent learns more informative message graphs and requires no labeled samples or a predetermined number of events. To the best of our knowledge, we are the first to apply SE minimization for social event detection.

- We design novel SE minimization algorithms for social event detection. Besides being effective, HISEvent efficiently runs on complex, large-scale message graphs. HISEvent incrementally and hierarchically minimizes 1D and 2D SE, significantly reducing time complexity compared to the existing SE minimization algorithms.

- We conduct extensive experiments on two large, public Twitter datasets to show the new SOTA performance, efficiency, and robustness of HISEvent. Our code is publicly available [1].

Preliminary

Structural entropy (SE) (Li and Pan 2016) is defined as the minimum number of bits to encode the vertex that is accessible with a step of random walk on a graph. The SE of a

[1] https://github.com/SELGroup/HISEvent

graph measures the complexity of the underlying essential structure and corresponds to an *encoding tree*. SE can be of different dimensions, which measure the structural information of different orders and correspond to encoding trees of different heights. We present the formal definitions of encoding tree and SE as follows. Notations used in this paper are summarized in Appendix.

Definition 1. *(Li and Pan 2016). The encoding tree $\mathcal{T}$ of a graph $G = (\mathcal{V}, \mathcal{E})$ is a hierarchical partition of G. It is a tree that satisfies the following:*

1) Each node α in $\mathcal{T}$ is associated with a set $T_\alpha \subseteq \mathcal{V}$. For the root node λ of $\mathcal{T}$, $T_\lambda = \mathcal{V}$. Any leaf node γ in $\mathcal{T}$ is associated with a single node in G, i.e., $T_\gamma = \{v\}, v \in \mathcal{V}$.

2) For each node α in $\mathcal{T}$, denote all its children as $\beta_1, ..., \beta_k$, then $(T_{\beta_1}, ..., T_{\beta_k})$ is a partition of T_α.

3) For each node α in $\mathcal{T}$, denote its height as $h(\alpha)$. Let $h(\gamma) = 0$ and $h(\alpha^-) = h(\alpha) + 1$, where α^- is the parent of α. The height of $\mathcal{T}$, $h(\mathcal{T}) = \max_{\alpha \in \mathcal{T}}\{h(\alpha)\}$.

Definition 2. *(Li and Pan 2016). The structural entropy (SE) of graph G on encoding tree $\mathcal{T}$ is defined as:*

$$\mathcal{H}^{\mathcal{T}}(G) = - \sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} \frac{g_\alpha}{vol(\lambda)} log \frac{vol(\alpha)}{vol(\alpha^-)}, \qquad (1)$$

where g_α is the summation of the degrees (weights) of the cut edges of T_α (edges in $\mathcal{E}$ that have exactly one endpoint in T_α). $vol(\alpha)$, $vol(\alpha^-)$, and $vol(\lambda)$ refer to the volumes, i.e., summations of the degrees of all the nodes, of T_α, T_{α^-}, and T_λ, respectively.

The d-dimensional SE of G, defined as $\mathcal{H}^{(d)}(G) = \min_{\forall \mathcal{T}:h(\mathcal{T})=d}\{\mathcal{H}^{\mathcal{T}}(G)\}$, is realized by acquiring an optimal encoding tree of height d, in which the disturbance derived

Algorithm 1: Determine $\mathcal{E}_s$ via incremental 1D SE minimization.

Input: Message graph node set $\mathcal{V}$
Output: Semantic-similarity-based edge set $\mathcal{E}_s$

1   SEs $\leftarrow \emptyset$
2   Embed $\mathcal{V}$ via PLM and get $\{\boldsymbol{h}_{m_i}\}_{i=1}^{|\mathcal{V}|}$
3   **for** $i = 1, ..., |\mathcal{V}|$ **do** // Sort neighbors
4      $\text{neighb}_{m_i} = (m_j)_{j=1}^{|\mathcal{V}|}$ s.t. $j \neq i$ and $\text{Cos}(\boldsymbol{h}_{m_i}, \boldsymbol{h}_{m_{j-1}}) > \text{Cos}(\boldsymbol{h}_{m_i}, \boldsymbol{h}_{m_j})$
5   $\mathcal{E} \leftarrow \{\text{1st element in neighb}_{m_i}\}_{i=1}^{|\mathcal{V}|}$
6   Calculate $\mathcal{H}^{(1)}(G)$ via Eq. 2
7   Append $\mathcal{H}^{(1)}(G)$ to SEs
8   $k = 2$
9   **while** $k < |\mathcal{V}|$ **do** // Search for the 1st stable point
10      $\mathcal{E} = \mathcal{E} \cup \{k\text{-th element in neighb}_{m_i}\}_{i=1}^{|\mathcal{V}|}$
11   Calculate $\mathcal{H}^{(1)\prime}(G)$ via Eq. 3
12   Append $\mathcal{H}^{(1)\prime}(G)$ to SEs
13   **if** $(k-1)$ *is a stable point** **then**
14   break
15      $k = k + 1$
16   $\mathcal{E}_s \leftarrow \{(m_i, m_j) | m_j \in$ the first (k-1) elements in $\text{neighb}_{m_i}\}_{i=1}^{|\mathcal{V}|}$
17   **return** $\mathcal{E}_s$

*$(k-1)$ is a stable point if the $(k-1)$-th element in SEs is smaller than the elements before and after it.

from noise or stochastic variation is minimized. Figure 1B.1 shows a toy example that constructs and optimizes $\mathcal{T}$ given G'.

Methodology

Figure 1 shows an overview of HISEvent. Following the previous methods (Ren et al. 2022a), we adopt a two-step, message graph construction-partitioning strategy. We first formalize the task. Next, we propose to incorporate a novel semantic-similarity-based approach for message graph construction. We then present our unsupervised message graph partitioning. Finally, we analyze the time complexity. HISEvent, as a batched (retrospective) method, can be easily extended to streaming scenarios (discussed in Appendix).

Problem Formalization

Given a sequence of social messages $m_1, ..., m_N$ as input, the task of social event detection can be fulfilled by constructing and partitioning a message graph $G = (\mathcal{V}, \mathcal{E})$. The node set $\mathcal{V} = \{m_1, ..., m_N\}$. The edge set $\mathcal{E}$ is initially empty and to be expanded by the message graph construction process. Partitioning G results in $\{e_1, ..., e_M\}, e_i \subset \mathcal{V}, e_i \cap e_j = \emptyset$, which is a partition of $\mathcal{V}$ containing M clusters (sets) of messages that correspond to the M detected social events.

Message Graph Construction with Incremental 1D SE Minimization

Ideally, the edges in the message graph should faithfully reflect the reliable message correlations while eliminating the noisy ones. Following the GNN-based studies (Ren et al. 2022a; Cao et al. 2021), we capture the common-attribute-based message correlations, visualized in Figure 1A. Specifically, for each message m_i, we extract its attributes $A_i = \{u_i\} \cup \{um_{i_1}, um_{i_2}, ...\} \cup \{h_{i_1}, h_{i_2}, ...\} \cup \{ne_{i_1}, ne_{i_2}, ...\}$, where the RHS refers to a union of the sender, mentioned users, hashtags, and named entities associated with m_i. We add an edge (m_i, m_j) into $\mathcal{E}_a$ iif m_i and m_j share some common attributes, i.e., $\mathcal{E}_a = \{(m_i, m_j) | A_i \cap A_j \neq \emptyset\}$.

$\mathcal{E}_a$ alone, however, can miss useful correlations, as there are messages that have similar semantics yet share no common attributes. To mitigate this, we supplement the message graph with semantic-similarity-based edges, denoted as $\mathcal{E}_s$. The similarity between two messages can be measured by embedding [2] them via pre-trained language models (PLMs), i.e., SBERT (Reimers and Gurevych 2019) then calculating the cosine similarity between their representations. The idea is to link each message to its k-nearest neighbors, where k needs to be carefully chosen to keep only the reliable connections. 1D SE minimization has been applied in biomedical studies (Li, Yin, and Pan 2016) to select the most correlated neighbors. Nonetheless, (Li, Yin, and Pan 2016) calculates the 1D SE from scratch for every candidate k, which is inefficient. We propose incremental 1D SE minimization for correlated neighbor selection. Specifically, we start with $\mathcal{E}_s = \emptyset$ and incrementally insert sets of edges into G, with the k-th set (referred to as k-NN edge set) containing edges between each node and its k-th nearest neighbor. The initial 1D SE with $k = 1$ is:

$$\mathcal{H}^{(1)}(G) = -\sum_{i=1}^{|\mathcal{V}|} \frac{d_i}{vol(\lambda)} log \frac{d_i}{vol(\lambda)}, \tag{2}$$

and the successive updates follow:

$$\mathcal{H}^{(1)\prime}(G) = \frac{vol(\lambda)}{vol'(\lambda)} \left(\mathcal{H}^{(1)}(G) - log \frac{vol(\lambda)}{vol'(\lambda)} \right) + \sum_{j=1}^{|a_k|} \left(\frac{d_j}{vol'(\lambda)} log \frac{d_j}{vol'(\lambda)} - \frac{d'_j}{vol'(\lambda)} log \frac{d'_j}{vol'(\lambda)} \right), \tag{3}$$

where d_i and d'_i denote the original and updated degrees (weighted) of node i in G before and after the insertion of the k-NN edge set, respectively. Initially, d_i is calculated with i linking to its 1st nearest neighbor. a_k is a set of nodes whose degrees are affected by the insertion of the k-NN edge set. $vol(\lambda)$ and $vol'(\lambda)$ stand for the volumes of G before and after inserting the k-NN edge set. $\mathcal{H}^{(1)}(G)$ and $\mathcal{H}^{(1)\prime}(G)$ stand for the original and updated 1D SE. The derivation of Equation 3 is in Appendix.

With the above initialization and update rules, selecting the proper k then follows Algorithm 1. Compared to (Li,

___

[2] Before embedding, we preprocess the message contents by filtering out URLs, extra characters, emotion icons, and user IDs, which we believe don't have clear natural language semantics.

Yin, and Pan 2016), the time needed for inspecting each candidate k (lines 10-12) is reduced from $O(|\mathcal{V}|)$ to $O(|a_k|)$ ($|a_k| \le |\mathcal{V}|$ always holds). Another difference is, HISEvent only uses $\mathcal{E}_s$ as a supplementation to $\mathcal{E}_a$. We, therefore, adopt the first stable point (lines 13-14) instead of the global one. The overall running time, due to lines 3-4, is $O(|\mathcal{V}|^2)$.

Finally, we set $\mathcal{E} = \mathcal{E}_a \cup \mathcal{E}_s$. For each edge (m_i, m_j), we then set its weight $w_{ij} = max(cosine(\boldsymbol{h}_{m_i}, \boldsymbol{h}_{m_j}), 0)$, where $\boldsymbol{h}_{m_i}$ and $\boldsymbol{h}_{m_j}$ denote the embeddings of m_i and m_j learned via PLMs. This accomplishes the construction of the message graph. HISEvent incorporates not only the common-attributes-based message correlations but also the semantic-similarity-based ones. It constructs more informative message graphs compared to the previous studies (Ren et al. 2022a; Peng et al. 2022; Cao et al. 2021).

Event Detection via Hierarchical 2D SE Minimization

Message graph partitioning decodes G into $\mathcal{P}$, which contains the detected events in the form of message clusters. A faithful decoding of the message correlations in G assigns related messages to the same cluster and unrelated ones to different clusters. Previous GNN-based detectors (Ren et al. 2022a; Cao et al. 2021) learn to properly partition message graphs through training, which require costly sample labeling and the number of events a priori. To address this issue, HISEvent conducts unsupervised partitioning under the guidance of 2D SE minimization, which eliminates the noise and reveals the essential 2nd-order (cluster-wise) structure underneath the raw graph with no prior knowledge of the number of event clusters.

(Li and Pan 2016) proposes a vanilla greedy 2D SE minimization algorithm that repeatedly merges any two nodes in the encoding tree $\mathcal{T}$ that would result in the largest decrease in 2D SE until reaches the minimum possible value. Hence it partitions a graph without supervision or a predetermined total number of clusters. We illustrate this algorithm in Appendix. This vanilla 2D SE minimization, however, takes $O(|\mathcal{V}|^3)$ to run. Though works for small bioinformatics graphs (Wu et al. 2022), it is prohibitively slow for the large, complex message graphs (demonstrated by the Experiments section). To address this, we propose to minimize 2D SE and detect events in a hierarchical manner, shown in Algorithm 2. Specifically, each message is initially in its own cluster (line 1). We split the clusters into subsets of size n (line 3) and merge the clusters involved in each subset using the vanilla greedy algorithm to get new clusters (lines 5-13). The new clusters are then passed on to the next iteration (line 14). This process is repeated until the clusters that contain all the messages are considered simultaneously (lines 15-16). If, at some point, none of the clusters in any subset can be merged, we increase n so that more clusters can be considered in the same subset and, therefore, may be merged (lines 17-18). Figure 1B visualizes this process: m_1 to m_9 are initially in their own clusters. $n = 3$ clusters are considered at a time to form a G'. Clusters in each G' are then merged via vanilla 2D SE minimization to get $\mathcal{P}'$ (Figure 1B.1). The partitions resulted in the previous iteration

---

Algorithm 2: Event detection via hierarchical 2D SE minimization.

Input: Message graph $G = (\mathcal{V}, \mathcal{E})$, sub-graph size n
Output: A partition $\mathcal{P}$ of $\mathcal{V}$

1   $\mathcal{P} \leftarrow (m | m \in \mathcal{V})$
2   **while** *True* **do**
3     $\{\mathcal{P}_s\} \leftarrow$ consecutively remove the first $min(n, \text{size of the remaining part of } \mathcal{P})$ clusters from $\mathcal{P}$ that form a set $\mathcal{P}_s$
4   **for** $\mathcal{P}_s \in \{\mathcal{P}_s\}$ **do**
5       $\mathcal{V}' \leftarrow$ combine all the clusters in $\mathcal{P}_s$
6       $\mathcal{E}' \leftarrow \{e \in \mathcal{E}, \text{both endpoints of } e \in \mathcal{V}'\}$
7       $G' \leftarrow (\mathcal{V}', \mathcal{E}')$
8       $\mathcal{T}' \leftarrow$ add a root tree node λ
9   **for** *cluster* $\mathcal{C} \in \mathcal{P}_s$ **do**
10   Add a tree node α to $\mathcal{T}'$, s.t. $\alpha^- = \lambda, T_\alpha = \mathcal{C}$
11   **for** *message* $m \in \mathcal{C}$ **do**
12   Add a tree node γ to $\mathcal{T}'$, s.t. $\gamma^- = \alpha, T_\gamma = \{m\}$
13       $\mathcal{P}' \leftarrow$ run vanilla 2D SE minimization (see Appendix) on G', with the initial encoding tree set to $\mathcal{T}'$
14   Append $\mathcal{P}'$ to $\mathcal{P}$
15   **if** $|\{\mathcal{V}'\}| = 1$ **then**
16   Break
17   **if** $\mathcal{P}$ *is the same as at the end of last iteration* **then**
18       $n \leftarrow 2n$
19   **return** $\mathcal{P}$

---

are passed on to the later iteration, as indicated by the blue curved arrows in Figure 1B.2. The process terminates when a $\mathcal{P}'$ that involves all the messages is determined. With a running time of $O(n^3)$, Algorithm 2 is much more efficient than its vanilla predecessor, as n is a hyperparameter that can be set to $\ll |\mathcal{V}|$. To summarize, HISEvent detects social events from the complex message graphs in an effective and unsupervised manner.

Time Complexity of HISEvent

The overall time complexity of HISEvent is $O(|\mathcal{E}_a| + |\mathcal{V}|^2 + n^3)$, where $|\mathcal{E}_a|$ is the total number of common-attribute-based edges in the message graph, $|\mathcal{V}|$ is the total number of nodes (i.e., messages) and n is sub-graph size, a hyperparameter that can be set to $\ll |\mathcal{V}|$. Specifically, the running time of constructing $\mathcal{E}_a$ is $O(|\mathcal{E}_a|)$. The running time of constructing the semantic-similarity-based edge set $\mathcal{E}_s$ is $O(|\mathcal{V}|^2)$. The running time of detecting social events from the constructed message graph is $O(n^3)$. Note HISEvent can be easily parallelized (discussed in Appendix).

Dataset	Metric	KPGNN*	QSGNN*	EventX	BERT*	SBERT*	HISEvent	Improv. (%)
Event2012	ARI	0.22	0.22	0.05	0.12	0.17	**0.50**	↑127
	AMI	0.52	0.53	0.19	0.43	0.73	**0.81**	↑11
Event2018	ARI	0.15	0.16	0.03	0.05	0.11	**0.44**	↑175
	AMI	0.44	0.44	0.16	0.34	0.62	**0.66**	↑6

Table 1: Closed-set results. * marks results acquired with the ground truth event numbers.

Blocks (#events)	M_1 (41)		M_2 (30)		M_3 (33)		M_4 (38)		M_5 (30)		M_6 (44)		M_7 (57)	
Metric	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI
KPGNN*	0.07	0.37	0.76	0.78	0.58	0.74	0.29	0.64	0.47	0.71	0.72	0.79	0.12	0.51
QSGNN*	0.07	0.41	0.77	0.80	0.59	0.76	0.29	0.68	0.48	0.73	0.73	0.80	0.12	0.54
EventX	0.01	0.06	0.45	0.29	0.09	0.18	0.07	0.19	0.04	0.14	0.14	0.27	0.02	0.13
BERT*	0.03	0.35	0.65	0.76	0.45	0.72	0.19	0.58	0.36	0.67	0.45	0.75	0.07	0.50
SBERT*	0.03	0.38	0.73	0.85	0.68	0.87	0.36	0.80	0.61	0.85	0.53	0.83	0.09	0.61
HISEvent	**0.08**	**0.44**	**0.79**	**0.88**	**0.95**	**0.94**	**0.50**	**0.84**	**0.62**	**0.85**	**0.86**	**0.90**	**0.27**	**0.68**
Improv. (%)	↑14	↑7	↑3	↑4	↑40	↑8	↑39	↑5	↑2	→	↑18	↑8	↑125	↑11
Blocks (#events)	M_8 (53)		M_9 (38)		M_{10} (33)		M_{11} (30)		M_{12} (42)		M_{13} (40)		M_{14} (43)	
Metric	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI
KPGNN*	0.60	0.76	0.46	0.71	0.70	0.78	0.49	0.71	0.48	0.66	0.29	0.67	0.42	0.65
QSGNN*	0.59	0.75	0.47	0.75	0.71	0.80	0.49	0.72	0.49	0.68	0.29	0.66	0.41	0.66
EventX	0.09	0.21	0.07	0.19	0.13	0.24	0.16	0.24	0.07	0.16	0.04	0.16	0.10	0.14
BERT*	0.51	0.74	0.34	0.71	0.55	0.78	0.26	0.62	0.31	0.56	0.13	0.57	0.24	0.55
SBERT*	0.65	0.86	0.47	0.83	0.62	0.85	0.49	**0.82**	0.63	0.85	0.24	0.70	0.40	0.77
HISEvent	**0.74**	**0.89**	**0.65**	**0.88**	**0.87**	**0.90**	**0.62**	**0.82**	**0.82**	**0.90**	**0.46**	**0.78**	**0.85**	**0.88**
Improv. (%)	↑14	↑3	↑38	↑6	↑23	↑6	↑27	→	↑30	↑6	↑59	↑11	↑102	↑14
Blocks (#events)	M_{15} (42)		M_{16} (27)		M_{17} (35)		M_{18} (32)		M_{19} (28)		M_{20} (34)		M_{21} (32)	
Metric	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI
KPGNN*	0.17	0.54	0.66	0.77	0.43	0.68	0.47	0.66	0.51	0.71	0.51	0.68	0.20	0.57
QSGNN*	0.17	0.55	0.65	0.76	0.44	0.69	0.48	0.68	0.50	0.70	0.51	0.69	0.21	0.58
EventX	0.01	0.07	0.08	0.19	0.12	0.18	0.08	0.16	0.07	0.16	0.11	0.18	0.01	0.10
BERT*	0.07	0.43	0.43	0.71	0.22	0.56	0.24	0.52	0.28	0.59	0.32	0.60	0.17	0.54
SBERT*	0.17	0.67	0.50	0.78	0.35	0.77	0.52	**0.81**	0.54	0.83	0.52	0.80	0.24	**0.70**
HISEvent	**0.27**	**0.72**	**0.83**	**0.87**	**0.56**	**0.81**	**0.70**	0.80	**0.63**	**0.87**	**0.69**	**0.81**	**0.45**	0.69
Improv. (%)	↑59	↑7	↑26	↑12	↑27	↑5	↑35	↓1	↑17	↑5	↑33	↑1	↑88	↓1

Table 2: Open-set results on Event2012. * marks results acquired with the ground truth event numbers.

Experiments

We conduct extensive experiments to compare HISEvent to various baselines and show the effectiveness of its components. We further analyze the efficiency as well as hyperparameter sensitivity of HISEvent and present a case study.

Experimental Setup

Datasets. We experiment on two large, public Twitter datasets, i.e., Event2012 (McMinn, Moshfeghi, and Jose 2013), and Event2018 (Mazoyer et al. 2020). Event2012 contains 68,841 English tweets related to 503 events, spreading over four weeks. Event2018 contains 64,516 French tweets about 257 events and were sent within a span of 23 days. We evaluate under both closed- and open-set settings by adopting the data splits of Ren et al. 2022a and Cao et al. 2021. The former simultaneously consider all the events, while the latter assumes the events happen over time and splits the datasets into day-wise message blocks (e.g., M_1 to M_{21} in Event2012). Data statistics are in Appendix.

Baselines. We compare HISEvent to **KPGNN** (Cao et al. 2021), a GNN-based social event detector, **QSGNN** (Ren et al. 2022a), which improves upon KPGNN using restricted pseudo labels and is the current SOTA, and **EventX** (Liu et al. 2020a), a non-GNN-based social event detector leverages community detection. We also experiment on PLMs, i.e., **BERT** (Kenton and Toutanova 2019), and **SBERT** (Reimers and Gurevych 2019): we first input the preprocessed message contents to PLMs to learn message embeddings and then apply K-means clustering on the message embeddings to acquire events, i.e., message clusters. Note that KPGNN and QSGNN are supervised. KPGNN, QSGNN, BERT, and SBERT require the total number of events to be specified a priori, which is impractical. HISEvent, in contrast, is unsupervised and does not need the total number of events as an input. Also note we omit the direct comparison with various techniques that are outperformed by the baselines, i.e., TF-IDF (Bafna, Pramod, and Vaidya 2016), LDA (Blei, Ng, and Jordan 2003), WMD (Kusner et al. 2015), LSTM (Graves and Schmidhuber 2005), word2vec (Mikolov et al. 2013), co-clustering (Dhillon, Mallela, and Modha 2003), NMF (Xu, Liu, and Gong 2003), etc. Implementation details are in Appendix.

Blocks	M_1		M_2		M_3		M_4		M_5		M_6		M_7		M_8	
Metric	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI
KPGNN*	0.17	0.54	0.18	0.55	0.15	0.55	0.17	0.55	0.21	0.57	0.21	0.57	*0.30*	0.61	0.20	0.57
QSGNN*	0.18	0.56	0.19	0.57	0.17	0.56	0.18	0.57	0.23	0.59	0.21	0.59	*0.30*	0.63	0.19	0.55
EventX	0.02	0.11	0.02	0.12	0.01	0.11	0.06	0.14	0.13	0.24	0.08	0.15	0.02	0.12	0.09	0.21
BERT*	0.16	0.42	0.21	0.44	0.22	0.44	0.17	0.41	0.31	0.56	0.23	0.49	0.23	0.49	0.24	0.50
SBERT*	*0.20*	*0.60*	*0.29*	*0.61*	*0.34*	*0.63*	*0.23*	*0.60*	*0.47*	*0.76*	*0.41*	*0.73*	0.29	*0.65*	*0.50*	*0.75*
HISEvent	**0.55**	**0.77**	**0.67**	**0.79**	**0.47**	**0.74**	**0.46**	**0.72**	**0.66**	**0.82**	**0.61**	**0.83**	**0.56**	**0.81**	**0.82**	**0.90**
Improv. (%)	↑175	↑28	↑131	↑30	↑38	↑17	↑100	↑20	↑40	↑8	↑49	↑14	↑87	↑25	↑64	↑20
Blocks	M_9		M_{10}		M_{11}		M_{12}		M_{13}		M_{14}		M_{15}		M_{16}	
Metric	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI	ARI	AMI
KPGNN*	0.10	0.46	0.18	0.56	0.16	0.53	0.17	0.56	0.28	0.60	0.43	0.65	0.25	0.58	0.13	0.50
QSGNN*	0.13	0.46	0.19	0.58	0.20	0.59	0.20	0.59	0.27	0.58	*0.44*	0.67	0.27	0.61	0.13	0.50
EventX	0.07	0.16	0.07	0.19	0.06	0.18	0.09	0.20	0.06	0.15	0.11	0.22	0.11	0.22	0.01	0.10
BERT*	0.17	0.42	0.19	0.46	0.18	0.48	0.32	0.54	0.18	0.40	0.27	0.52	0.28	0.53	0.21	0.43
SBERT*	*0.23*	*0.63*	*0.39*	*0.72*	*0.31*	*0.70*	*0.54*	*0.76*	*0.34*	*0.65*	0.43	*0.68*	*0.40*	*0.71*	*0.25*	*0.65*
HISEvent	**0.65**	**0.73**	**0.51**	**0.80**	**0.44**	**0.79**	**0.86**	**0.88**	**0.83**	**0.89**	**0.80**	**0.89**	**0.70**	**0.84**	**0.37**	**0.73**
Improv. (%)	↑183	↑16	↑31	↑11	↑42	↑13	↑59	↑16	↑144	↑37	↑82	↑31	↑75	↑18	↑48	↑12

Table 3: Open-set results on Event2018. * marks results acquired with the ground truth event numbers.

Setting	Closed-set		Open-set (Avg.)	
Metric	ARI	AMI	ARI	AMI
HISEvent	**0.50**	**0.81**	**0.63**	**0.82**
$-\mathcal{E}_s$	0.24	0.58	0.40	0.60
$-\mathcal{E}_a$	*0.42*	*0.80*	0.51	*0.77*
HISEvent-BERT	0.25	0.65	0.52	0.69
HISEvent-vanilla	takes > 10 days		*0.62*	**0.82**

Table 4: Ablation study on Event2012. $-\mathcal{E}_s$ removes the semantic-similarity-based $\mathcal{E}_s$ and simply relies on the common-attribute-based $\mathcal{E}_a$ to capture message correlations. Similarly, $-\mathcal{E}_a$ relies solely on $\mathcal{E}_s$ (unlike in Section 3.2, here we use the global rather than the first stable points since $\mathcal{E}_s$ is no longer a supplementation but aims to fully capture the message correlations). HISEvent-BERT uses BERT rather than SBERT to measure the edge weights. HISEvent-vanilla partitions the message graph via vanilla 2D SE minimization instead of our proposed hierarchical one.

Evaluation Metrics. We measure adjusted mutual information (AMI), adjusted rand index (ARI), and normalized mutual information (NMI, in Appendix), which are broadly used by the previous studies (Cao et al. 2021).

Overall Performance

Tables 1 - 3 show the social event detection performance. HISEvent consistently outperforms the highest baseline by large margins on both datasets across the closed- and open-set settings. E.g., on Event2018, HISEvent improves ARI and AMI upon SBERT, the strongest baseline, by 175% and 6% under the closed-set setting and by 77% and 19% on average under the open-set setting. This verifies that HISEvent better explores the message semantics and the social network structure. Meanwhile, a comparison between the baselines indicates that the quality of the message embeddings matter: SBERT outperforms BERT and the GNN-based methods. Besides being effective and unsupervised,

HISEvent does not require predetermining the number of events. This is essential as the number of events is difficult to predict. E.g., in Table 2, the ground truth number of events varies from 27 to 57 and can drop from 42 in M_{15} to 27 in M_{16} and raise from 30 in M_5 to 44 in M_6 between consecutive periods. In contrast, KPGNN and QSGNN require labeled samples while KPGNN, QSGNN, BERT, and SBERT need the total number of events given a priori, which is impossible in practice. In short, HISEvent is more practical than the baselines and is the new SOTA.

Ablation Study

Table 4 presents the ablation studies on Event2012. All the components of HISEvent help. Especially, $\mathcal{E}_s$, absent in KPGNN and QSGNN, is essential for HISEvent's good performance. E.g., $-\mathcal{E}_s$ underperforms HISEvent by 52% and 28% in ARI and AMI, respectively, in the closed-set experiment. Also note HISEvent-BERT significantly outperforms BERT (shown in Tables 1 and 2), indicating that HISEvent works despite the choice of PLM. Meanwhile, we observe that HISEvent-BERT underperforms HISEvent, indicating that PLM embeddings that are of High-quality and, in particular, faithfully reflect messages' semantic similarities (i.e., SBERT embeddings) are indispensable for HISEvent's good performance. Adopting embedding unsuitable for message similarity measuring (i.e., BERT embeddings), on the other hand, can lead to a decrease in performance (further discussed in Appendix). A comparison to HISEvent-vanilla shows that HISEvent, adopts the proposed hierarchical 2D SE minimization algorithm, significantly improves efficiency without sacrificing performance: it performs on par with HISEvent-vanilla but is orders of magnitude faster (further verified in the next section).

Efficiency of HISEvent

We compare the efficiency of the proposed hierarchical 2D SE minimization to its vanilla predecessor. Figure 2 shows their time consumption on Event2012 message blocks. The
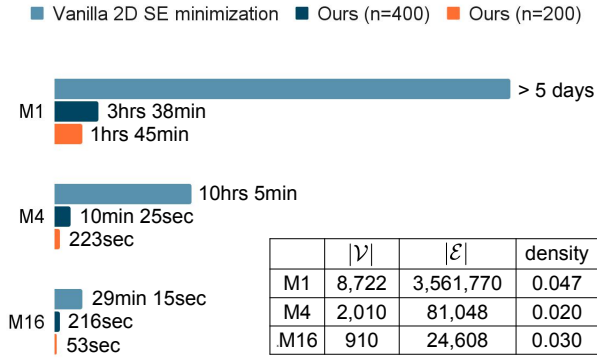
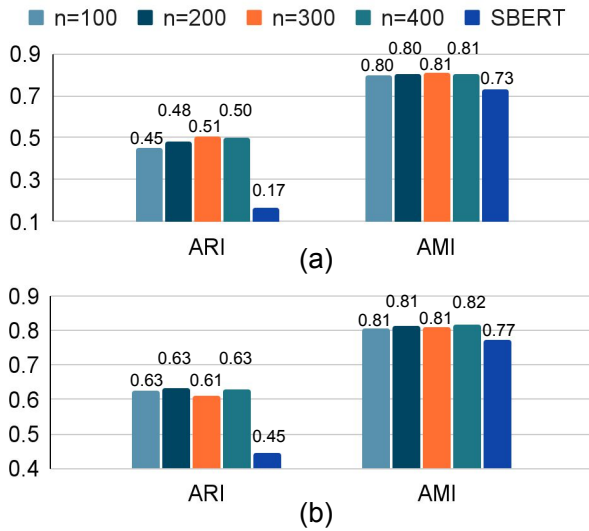Figure 2: Running time comparison between vanilla and hierarchical (ours) 2D SE minimization on Event2012.

Figure 3: HISEvent results on Event2012 with different n. (a) and (b) show the closed-set and open-set (averaged) results.

vanilla algorithm runs prohibitively slow on complex message graphs. E.g., for a large and dense message block such as M_1, it takes more than 5 days to complete. In contrast, our proposed hierarchical 2D SE minimization dramatically reduces time consumption. E.g., for M_1, our algorithm reduces the running time by $>97\%$. Adopting a smaller subgraph size n further decreases the running time. E.g., adopting a n of 200 rather than 400 further reduces the time needed for M_1 by half. Also note that the impact to the performance when n is decreased is rather small (further discussed in the next section).

Hyperparameter Sensitivity

We study how changing the sub-graph size n affects the performance of HISEvent. Figure 3 shows that HISEvent is relatively robust to the changes in n: increasing n slightly prompts the performance at the cost of longer running time. Take Figure 3(a), the closed-set results on Event2012, for

(a) sample message
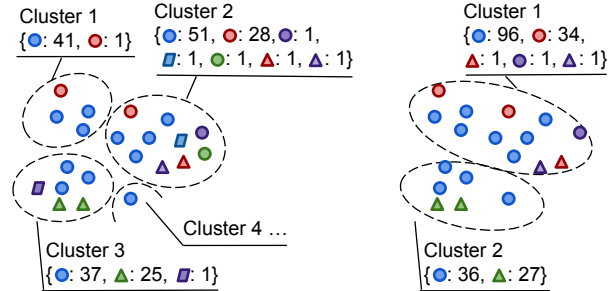
Event_43: Hurricane Sandy Event_432: NY stock after Sandy
Event_16: Aftermath of Sandy Event_434: India evacuation
Event_156: Praying for people affected by Sandy
Event_46: Purchase Lucasfilm Ltd. Event_15: Pride of Britain
Event_435: Electrical fire at a Saudi wedding
Event_367: Bahrain bans protests gatherings

Figure 4: Detection of Event_43 Hurricane Sandy. (a) is a sample message. (b) and (c) are clusters detected by SBERT and HISEvent that contain the target event messages.

example, increasing n from 100 to 400 introduces moderate (10%) and marginal (1%) improvements in ARI and AMI. Moreover, despite the changes in n, HISEvent always outperforms SBERT, the strongest baseline, by 169-197% in ARI and 9-10% in AMI.

Case Study

Figure 4 presents the detection of *Event_43 Hurricane Sandy*. We observe that the strongest baseline, SBERT, confuses the target event with many irrelevant events such as *Event_15 Pride of Britain* and *Event_367 Bahrain bans protests gathering*. As a result, the meaning of its detected clusters are rather vague. Moreover, SBERT outputs disjoint rather than more favorable, coherent clusters to represent the target event. E.g., it represents the target event with more than 4 clusters. In contrast, the proposed HISEvent detects justifiable clusters with clear meanings. E.g., cluster 1 is the destruction brought by Sandy while cluster 2 reflects the public reaction and recovery afterwards. Moreover, HISEvent fails only on the hard negatives. E.g., cluster 2 includes some messages about *Event_432 NY stock after Sandy*, which is relevant to the target event.

Related Work

Social event detection is a long-standing task (Atefeh and Khreich 2015). The main challenges lie in exploring the high-volume, complex, noisy, and dynamic social media components, e.g., text, timestamp, user mention, and social network structure. Studies leverage incremental clustering (Zhao, Mitra, and Chen 2007; Weng and Lee 2011; Ag-

garwal and Subbian 2012; Zhang, Zi, and Wu 2007; Feng et al. 2015; Xie et al. 2016), community detection (Fedoryszak et al. 2019; Liu et al. 2020a,b; Yu et al. 2017), and topic modeling (Zhou and Chen 2014; Zhou, Chen, and He 2015; Xing et al. 2016; Wang et al. 2016; Zhao et al. 2011) are common. There are also methods for specific domains (Yao et al. 2020; Arachie et al. 2020; Khandpur et al. 2017) such as airport threats. They extract attributes, e.g., hashtag, from the social media components then pre-process the attributes in highly-customized manners. GNN-based methods (Peng et al. 2019; Cao et al. 2021; Peng et al. 2021; Ren et al. 2022b, 2021, 2022a; Peng et al. 2022) unify the various components concisely by introducing message graphs and quickly became a new trend for their outstanding performance. HISEvent keeps the merits of the GNN-based methods, better captures semantic-based message correlations, and eliminates sample labeling. Please also note that *social event detection*, which highlights significant occurrences on social media, *news story discovery* (Yoon et al. 2023), which summarizes long, formal, and plain textual news documents other than short, informal, and structural social messages, *event prediction* (Zhao, Wang, and Guo 2018; Deng, Rangwala, and Ning 2019; Pan et al. 2020), which forecasts future events, and *event extraction* (Liu, Huang, and Zhang 2019), which detects the entities, triggers, arguments, etc., of events, are non-comparable tasks.

Conclusion

We address social event detection from a structural entropy perspective. HISEvent provides an effective, efficient, and unsupervised tool for social event detection and analysis. It keeps the merits of the GNN-based methods, better explores message correlations, and eliminates the need for labeling or predetermining the number of events. Experiments show that HISEvent achieves the new SOTA under both closed- and open-set settings while being efficient and robust.

Acknowledgments

References

Aggarwal, C. C.; and Subbian, K. 2012. Event detection in social streams. In *Proceedings of the 2012 SIAM international conference on data mining*, 624–635. SIAM.

Arachie, C.; Gaur, M.; Anzaroot, S.; Groves, W.; Zhang, K.; and Jaimes, A. 2020. Unsupervised detection of sub-events in large scale disasters. In *Proceedings Of The AAAI conference on artificial intelligence*, volume 34, 354–361.

Atefeh, F.; and Khreich, W. 2015. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1): 132–164.

Bafna, P.; Pramod, D.; and Vaidya, A. 2016. Document clustering: TF-IDF approach. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, 61–66. IEEE.

Beck, T.; Lee, J.-U.; Viehmann, C.; Maurer, M.; Quiring, O.; and Gurevych, I. 2021. Investigating label suggestions for opinion mining in German Covid-19 social media. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan): 993–1022.

Cao, Y.; Peng, H.; Wu, J.; Dou, Y.; Li, J.; and Yu, P. S. 2021. Knowledge-preserving incremental social event detection via heterogeneous gnns. In *Proceedings of the Web Conference 2021*, 3383–3395.

Deng, S.; Rangwala, H.; and Ning, Y. 2019. Learning dynamic context graphs for predicting social events. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1007–1016.

Dhillon, I. S.; Mallela, S.; and Modha, D. S. 2003. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 89–98.

Fedoryszak, M.; Frederick, B.; Rajaram, V.; and Zhong, C. 2019. Real-time event detection on social data streams. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2774–2782.

Feng, W.; Zhang, C.; Zhang, W.; Han, J.; Wang, J.; Aggarwal, C.; and Huang, J. 2015. STREAMCUBE: Hierarchical spatio-temporal hashtag clustering for event exploration over the Twitter stream. In *2015 IEEE 31st international conference on data engineering*, 1561–1572. IEEE.

Graves, A.; and Schmidhuber, J. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks*, 18(5-6): 602–610.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Proceedings of Advances in neural information processing systems*, 1025–1035.

Kenton, J. D. M.-W. C.; and Toutanova, L. K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*, 4171–4186.

Khandpur, R.; Ji, T.; Ning, Y.; Zhao, L.; Lu, C.-T.; Smith, E.; Adams, C.; and Ramakrishnan, N. 2017. Determining relative airport threats from news and social media. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 4701–4707.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR 2017*.

Kusner, M.; Sun, Y.; Kolkin, N.; and Weinberger, K. 2015. From word embeddings to document distances. In *International conference on machine learning*, 957–966. PMLR.

Li, A.; and Pan, Y. 2016. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory*, 62(6): 3290–3339.

Li, A.; Yin, X.; and Pan, Y. 2016. Three-dimensional gene map of cancer cell types: Structural entropy minimisation principle for defining tumour subtypes. *Scientific reports*, 6(1): 1–26.

Li, A.; Yin, X.; Xu, B.; Wang, D.; Han, J.; Wei, Y.; Deng, Y.; Xiong, Y.; and Zhang, Z. 2018. Decoding topologically associating domains with ultra-low resolution Hi-C data by graph structural entropy. *Nature communications*, 9(1): 1–12.

Liu, B.; Han, F. X.; Niu, D.; Kong, L.; Lai, K.; and Xu, Y. 2020a. Story forest: Extracting events and telling stories from breaking news. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(3): 1–28.

Liu, X.; Huang, H.; and Zhang, Y. 2019. Open domain event extraction using neural latent variable models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Liu, Y.; Peng, H.; Li, J.; Song, Y.; and Li, X. 2020b. Event detection and evolution in multi-lingual social streams. *Frontiers of Computer Science*, 14(5): 1–15.

Mazoyer, B.; Cagé, J.; Hervé, N.; and Hudelot, C. 2020. A french corpus for event detection on twitter. In *Proceedings of the 12th language resources and evaluation conference*, 6220–6227.

McMinn, A. J.; Moshfeghi, Y.; and Jose, J. M. 2013. Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 409–418.

Mehta, N.; Pacheco, M. L.; and Goldwasser, D. 2022. Tackling Fake News Detection by Continually Improving Social Context Representations using Graph Neural Networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1363–1380.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Pan, Z.; Huang, Z.; Lian, D.; and Chen, E. 2020. A variational point process model for social event sequences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 173–180.

Peng, H.; Li, J.; Gong, Q.; Song, Y.; Ning, Y.; Lai, K.; and Yu, P. S. 2019. Fine-grained event categorization with heterogeneous graph convolutional networks. In *Proceedings of IJCAI 2022*, 3238–3245.

Peng, H.; Li, J.; Song, Y.; Yang, R.; Ranjan, R.; Yu, P. S.; and He, L. 2021. Streaming social event detection and evolution discovery in heterogeneous information networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(5): 1–33.

Peng, H.; Zhang, R.; Li, S.; Cao, Y.; Pan, S.; and Yu, P. 2022. Reinforced, incremental and cross-lingual event detection from social messages. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–18.

Reimers, N.; and Gurevych, I. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP 2019*.

Ren, J.; Jiang, L.; Peng, H.; Cao, Y.; Wu, J.; Yu, P. S.; and He, L. 2022a. From Known to Unknown: Quality-aware Self-improving Graph Neural Network for Open Set Social Event Detection. In *Proceedings of ACM CIKM 2022*, 1696–1705.

Ren, J.; Jiang, L.; Peng, H.; Liu, Z.; Wu, J.; and Philip, S. Y. 2022b. Evidential Temporal-aware Graph-based Social Event Detection via Dempster-Shafer Theory. In *2022 IEEE International Conference on Web Services (ICWS)*, 331–336. IEEE.

Ren, J.; Peng, H.; Jiang, L.; Liu, Z.; Wu, J.; Yu, Z.; and Philip, S. Y. 2023. Uncertainty-guided Boundary Learning for Imbalanced Social Event Detection. *IEEE Transactions on Knowledge and Data Engineering*.

Ren, J.; Peng, H.; Jiang, L.; Wu, J.; Tong, Y.; Wang, L.; Bai, X.; Wang, B.; and Yang, Q. 2021. Transferring Knowledge Distillation for Multilingual Social Event Detection. *arXiv preprint arXiv:2108.03084*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph attention networks. In *Proceedings of ICLR 2018*.

Wang, Y.; Liu, J.; Huang, Y.; and Feng, X. 2016. Using hashtag graph-based topic model to connect semantically-related words without co-occurrence in microblogs. *IEEE Transactions on Knowledge and Data Engineering*, 28(7): 1919–1933.

Weng, J.; and Lee, B.-S. 2011. Event detection in twitter. In *Proceedings of the international aaai conference on web and social media*, volume 5, 401–408.

Wu, J.; Chen, X.; Xu, K.; and Li, S. 2022. Structural entropy guided graph hierarchical pooling. In *Proceedings of International Conference on Machine Learning*, 24017–24030. PMLR.

Xie, W.; Zhu, F.; Jiang, J.; Lim, E.-P.; and Wang, K. 2016. Topicsketch: Real-time bursty topic detection from twitter. *IEEE Transactions on Knowledge and Data Engineering*, 28(8): 2216–2229.

Xing, C.; Wang, Y.; Liu, J.; Huang, Y.; and Ma, W.-Y. 2016. Hashtag-based sub-event discovery using mutually generative lda in twitter. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Xu, W.; Liu, X.; and Gong, Y. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 267–273.

Yao, W.; Zhang, C.; Saravanan, S.; Huang, R.; and Mostafavi, A. 2020. Weakly-supervised fine-grained event recognition on social media texts for disaster management.

In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 532–539.

Yoon, S.; Lee, D.; Zhang, Y.; and Han, J. 2023. Unsupervised Story Discovery from Continuous News Streams via Scalable Thematic Embedding. *arXiv preprint arXiv:2304.04099*.

Yu, W.; Li, J.; Bhuiyan, M. Z. A.; Zhang, R.; and Huai, J. 2017. Ring: Real-time emerging anomaly monitoring system over text streams. *IEEE Transactions on Big Data*, 5(4): 506–519.

Zhang, K.; Zi, J.; and Wu, L. G. 2007. New event detection based on indexing-tree and named entity. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 215–222.

Zhao, L.; Wang, J.; and Guo, X. 2018. Distant-supervision of heterogeneous multitask learning for social event forecasting with multilingual indicators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zhao, Q.; Mitra, P.; and Chen, B. 2007. Temporal and information flow based event detection from social text streams. In *Proceedings of the 22nd national conference on Artificial intelligence-Volume 2*, 1501–1506.

Zhao, W. X.; Jiang, J.; Weng, J.; He, J.; Lim, E.-P.; Yan, H.; and Li, X. 2011. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval: 33rd European Conference on IR Research, ECIR 2011, Dublin, Ireland, April 18-21, 2011. Proceedings 33*, 338–349. Springer.

Zhou, D.; Chen, L.; and He, Y. 2015. An unsupervised framework of exploring events on twitter: Filtering, extraction and categorization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

Zhou, X.; and Chen, L. 2014. Event detection over twitter social media streams. *The VLDB journal*, 23(3): 381–400.