Effective and Unsupervised Social Event Detection and Evolution via RAG and Structural Entropy

Qitong Liu
Beihang University
Beijing, China
liuqt@buaa.edu.cn

Hao Peng*
Beihang University
Beijing, China
penghao@buaa.edu.cn

Zuchen Li
Beihang University
Beijing, China
lizuchen@buaa.edu.cn

Xihang Meng
Beihang University
Beijing, China
xihangmeng@buaa.edu.cn

Ziyu Yang
Beihang University
Beijing, China
yangziyu@buaa.edu.cn

Jiting Li
Academy of Military Sciences
Beijing, China
lijiting_1993@126.com

Li Sun
North China Electric Power University
Beijing, China
ccesunli@ncepu.edu.cn

Philip S. Yu
University of Illinois Chicago
Chicago, USA
psyu@uic.edu

Abstract

With the growing scale of social media, social event detection and evolution modeling have attracted increasing attention. Graph neural networks (GNNs) and transformer-based pre-trained language models (PLMs) have become mainstream approaches in this area. However, existing methods still face three major challenges. First, the sheer volume of social media messages makes learning resource-intensive. Second, the fragmentation of social media messages often impedes the model's ability to capture a comprehensive view of the events. Third, the lack of structured temporal context has hindered the development of effective models for event evolution, limiting users' access to event information. To address these challenges, we propose a foundation model for unsupervised Social Event Detection and Evolution, namely RagSEDE. Specifically, RagSEDE introduces a representativeness- and diversity-driven sampling strategy to extract key messages from massive social streams, significantly reducing noise and computational overhead. It further establishes a novel paradigm based on Retrieval Augmented Generation (RAG) that enhances PLMs in detecting events while simultaneously constructing and maintaining an evolving event knowledge base. Finally, RagSEDE leverages structural information theory to dynamically model event evolution keywords for the first time. Extensive experiments on two public datasets demonstrate the superiority of RagSEDE in open-world social event detection and evolution.

CCS Concepts

• **Information systems → Social networks**.

Keywords

Social event detection, Social event evolution, Retrieval-augmented generation, Structural information theory

*Corresponding author.

ACM Reference Format:
Qitong Liu, Hao Peng, Zuchen Li, Xihang Meng, Ziyu Yang, Jiting Li, Li Sun, and Philip S. Yu. 2026. Effective and Unsupervised Social Event Detection and Evolution via RAG and Structural Entropy. In *Proceedings of the ACM Web Conference 2026 (WWW '26), April 13–17, 2026, Dubai, United Arab Emirates.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

1 Introduction

Social media platforms have become a vital source of real-time signals for emerging events, drawing increasing attention to the task of event analysis from large-scale, unstructured social message streams [31, 49]. Social Event Detection (SED) focuses on automatically identifying and clustering social messages that pertain to the same real-world event [3]. Complementing this, Social Event Evolution (SEE) aims to characterize the temporal dynamics of an event, capturing how its semantic content evolves over time [29]. Together, SED and SEE support a wide range of downstream applications, including community discovery [13], recommender systems [33, 56], and information retrieval [6, 32].

As shown in Figure 1(a), recent models for social event analysis primarily focus on event detection. These models typically follow a representation–clustering paradigm, where social messages are first encoded into latent representations and then clustered, with each cluster corresponding to a distinct event. Several studies [3, 5, 24, 28, 35] leverage heterogeneous interactions (e.g., users, entities, hashtags) to model the structural dependencies among social messages. Furthermore, other works [20, 52] utilize the powerful contextual semantics provided by PLMs such as BERT [7] and SBERT [34] to encode social messages. Most of these methods [3, 30] use K-Means or DBSCAN [9] to cluster message representations, and [4] employs a structural entropy minimization algorithm [19] to achieve unsupervised clustering.

However, existing approaches face several critical challenges in the context of open-world, unsupervised social event modeling. **First**, the massive volume and informal nature of social messages lead to resource-intensive computation and high noise levels. For popular events, there are often large numbers of semantically similar or even identical messages. [3, 4, 35] use all messages for event detection, resulting in significant resource waste. Furthermore, users often post messages randomly, and many messages

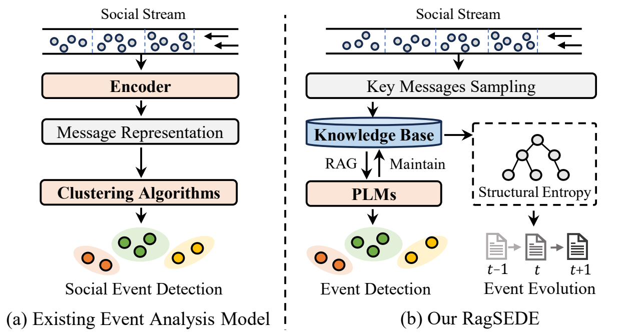


Figure 1: Illustration comparison of the existing event analysis model and our proposed RagSEDE.

lack attributes, leading to missing or noisy edges in GNN-based methods. Although [53] employs message anchors to reduce computational overhead, it still relies on building a global graph using all messages with noisy edges. **Second**, the fragmented nature of social messages often prevents models from capturing a complete picture of events, resulting in shallow representations. As shown in Figure 1(a), existing methods rely solely on message-level embeddings during the clustering process, without incorporating any global semantic guidance from previously occurred events. Ignoring these global signals hinders the model's ability to align new messages with the broader narrative, particularly in cases involving ambiguous phrasing or topic drift [17]. **Third**, most current works focus only on event detection, overlooking the temporal dynamics and evolution of events, which are crucial for understanding how narratives unfold over time. Although [23] considers the evolution of sub-event structure over time, it ignores the evolution of event semantics. This limitation significantly restricts users' access to time-sensitive event knowledge.

To tackle these three aforementioned challenges, we propose a novel foundation model for unsupervised Social Event Detection and Evolution, namely RagSEDE. Our proposed framework, as illustrated in Figure 1(b), consists of three parts: key message sampling, RAG-based event detection, and structural entropy-based event evolution. **First**, to mitigate the computational inefficiency caused by highly semantically similar messages, RagSEDE employs a sampling strategy that balances representativeness and diversity to extract key messages from the social message stream. Using only these key messages rather than the entire message set for event detection, RagSEDE achieves efficient detection in open-world settings. **Second**, leveraging the powerful contextual reasoning capabilities of PLMs, RagSEDE introduces a novel RAG-based event detection paradigm. Unlike prior methods that follow a "representation-then-cluster" approach, RagSEDE constructs and maintains a structured knowledge base of detected events and employs RAG [11, 18, 40, 54] to inject global semantic signals into PLMs during detection. Furthermore, because RagSEDE relies solely on the semantic content of messages, it avoids the issues of missing or noisy edges mentioned above. **Third**, by integrating the constructed knowledge base with structural information theory [19], RagSEDE dynamically model the semantic evolution of events over time. Structural information theory [19] suggests that minimizing structural entropy can reveal the essential information embedded in the graph. Based on this principle, RagSEDE introduces graph construction, inheritance, and forgetting mechanisms, enabling it to extract temporally evolving event keywords by minimizing the structural entropy. The codes of RagSEDE are publicly available on GitHub[1]. In summary, the contributions of this paper are as follows:

- We propose a novel framework for streaming, unsupervised social event detection and evolution, termed RagSEDE, which integrates RAG and structural information theory to effectively address three key challenges in open-world social event analysis.
- We propose a novel paradigm for event detection that leverages knowledge bases and RAG, fully exploiting the powerful contextual understanding capabilities of PLMs. By incorporating a key message sampling strategy, our method significantly reduces computational costs while maintaining detection effectiveness.
- We model the dynamic evolution of event keywords. By minimizing structural entropy over time, our framework effectively captures the evolving semantic cores of events.
- Extensive experiments on two benchmark datasets demonstrate that RagSEDE consistently outperforms strong baselines in both detection accuracy and evolution summarization.

2 Preliminaries

This section summarizes some important concepts and definitions. The glossary of notations and the detailed introduction of structural entropy are provided in Appendix B and Appendix C, respectively.

2.1 SED

A social stream S is defined as a continuous and temporal sequence of message blocks, that is, $S = \{\mathrm{M}_1, \mathrm{M}_2, \cdots\}$, where block $\mathrm{M}_t = \{m_i \mid 1 \le i \le |\mathrm{M}_t|\}$ contains all messages that arrive within the time interval $[t, t+1)$. The SED task is to extract clusters of correlated messages from S to represent real-world events. Finally, each m_i corresponds to an event label y_{m_i} with which it is associated.

2.2 Structural Entropy

Structural entropy [19] is a measure of the uncertainty of a graph structure. It is computed based on the graph's 2D encoding tree, where non-leaf nodes represent graph partitions. Given a graph G and its encoding tree $\mathcal{T}$, the 2D structural entropy is defined as:

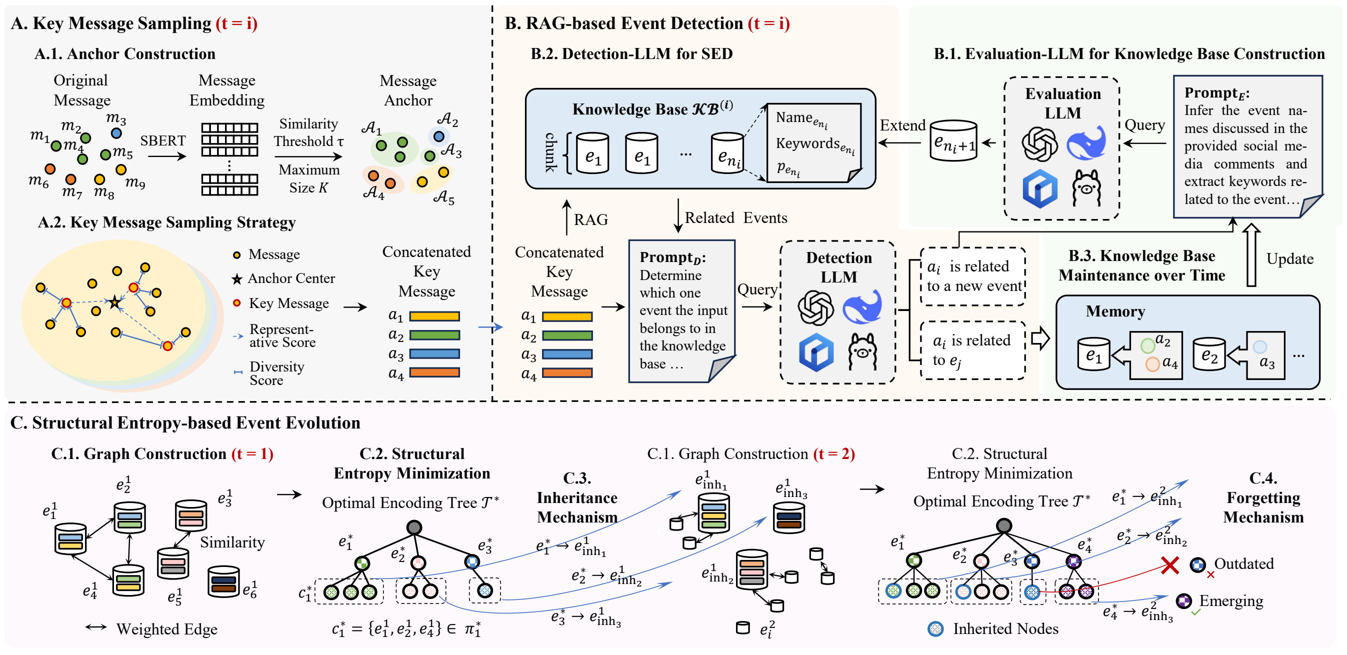
$$H^{\mathcal{T}}(G) = -\sum_{\alpha\in\mathcal{T},\ \alpha\neq\lambda} \frac{g_\alpha}{vol(G)} \log_2 \frac{vol(\alpha)}{vol(\alpha^-)}, \tag{1}$$

where α is a non-root node of $\mathcal{T}$, g_α and $vol(\alpha)$ is the cut degree and volume of α, and α^- is the parent node of α. [19] proposes a vanilla greedy 2D structural entropy minimization algorithm, which repeatedly merges any two nodes in the encoding tree until structural entropy reaches the minimum possible value.

3 Methodology

In this section, we systematically describe the proposed RagSEDE. Section 3.1 presents a strategy for sample key messages. Section 3.2 introduces the event detection paradigm of RagSEDE based on RAG-enhanced PLMs. Section 3.3 introduces the event evolution method based on structural entropy minimization.

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



A. Key Message Sampling (t = i)
A.1. Anchor Construction
Original Message
Message Embedding
Message Anchor
SBERT
Similarity Threshold τ
Maximum Size K
A.2. Key Message Sampling Strategy
Message
Anchor Center
Key Message
Representative Score
Diversity Score
Concatenated Key Message
B. RAG-based Event Detection (t = i)
B.2. Detection-LLM for SED
Knowledge Base KB^(i)
chunk
Name
Keywords
Extend
Query
RAG
Related Events
Concatenated Key Message
Prompt_D: Determine which one event the input belongs to in the knowledge base …
Query
Detection LLM
a_i is related to a new event
a_i is related to e_j
B.1. Evaluation-LLM for Knowledge Base Construction
Evaluation LLM
Prompt_E: Infer the event names discussed in the provided social media comments and extract keywords related to the event…
B.3. Knowledge Base Maintenance over Time
Update
Memory
C. Structural Entropy-based Event Evolution
C.1. Graph Construction (t = 1)
Similarity
Weighted Edge
C.2. Structural Entropy Minimization
Optimal Encoding Tree T*
C.3. Inheritance Mechanism
C.1. Graph Construction (t = 2)
C.2. Structural Entropy Minimization
Optimal Encoding Tree T*
C.4. Forgetting Mechanism
Outdated
Emerging
Inherited Nodes

Figure 2: The proposed RagSEDE framework.

3.1 Key Message Sampling (KMS)

Given a message block M_t in a high-throughput social stream, we select a set of key messages from M_t that preserves both the representativeness and the diversity within each anchor, as shown in Figure 2(A). This approach reduces computation while maintaining detection quality for downstream detection and evolution.

3.1.1 Anchor Construction. This step aims to aggregate messages that are highly semantically similar or even exactly identical into anchors. We first encode each message m_i into a z_i. Specifically,

$$z_i = \mathrm{Enc}(m_i) \in \mathbb{R}^d, \tag{2}$$

where $\mathrm{Enc}(\cdot)$ denotes a sentence encoder, which we use SBERT here, and d is the hidden state dimension of SBERT. Then, we compute the cosine similarity between two messages, specifically,

$$s_{ij} = \frac{z_i^\top z_j}{\|z_i\|_2 \, \|z_j\|_2}, \tag{3}$$

where $\|\cdot\|$ represents the euclidean norm. Messages are assigned to the same anchor $\mathcal{A}_k$ if and only if

$$s_{ij} \geq \tau, \quad \forall m_i, m_j \in \mathcal{A}_k, \tag{4}$$

where $\tau \in (0,1)$ is a predefined similarity threshold. To avoid oversized anchors for popular events, we set a maximum size K to limit the number of messages in each anchor. If adding a new message would cause $|\mathcal{A}_k| > K$, a new anchor is created. All messages can thus be represented as an anchor collection $\mathbb{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots\}$.

3.1.2 Key Message Sampling Strategy. To select key messages from each anchor, we define two scoring criteria.

Representativeness Score. The representativeness score measures how well a message reflects the central semantics of its anchor. Formally, for a m_i in $\mathcal{A}_k$, representativeness score is defined as

$$\mathrm{Rep}(m_i) = \frac{z_i^\top c_k}{\|z_i\|_2 \, \|c_k\|_2}, \quad c_k = \frac{1}{n}\sum_{j=1}^{n} z_j, \tag{5}$$

where z_i is the embedding of m_i, c_k is the center embedding of $\mathcal{A}_k$, and n is the message number of $\mathcal{A}_k$. A high representativeness score indicates strong semantic alignment with the anchor center, thereby capturing the dominant event of the anchor.

Diversity Score. The diversity score measures the degree to which a message provides novel information from the same anchor. For a message m_i in anchor $\mathcal{A}_k$, its diversity score is defined as

$$\mathrm{Div}(m_i) = \frac{1}{n-1} \sum_{m_j \in \mathcal{A}_k, j \neq i} \frac{z_i^\top z_j}{\|z_i\|_2 \, \|z_j\|_2}, \tag{6}$$

where z_i is the embedding of m_i, and n is the message number of $\mathcal{A}_k$. A high diversity score means different aspects of the event.

Combined Score. The final score balances the two criteria:

$$\mathrm{S}(m_i) = \lambda \cdot \mathrm{Rep}(m_i) + (1-\lambda) \cdot \mathrm{Div}(m_i), \tag{7}$$

where $\lambda \in [0,1]$ controls the trade-off. At each anchor $\mathcal{A}_k$, we select the top p messages with the highest combined scores as key messages and concatenate them as a_k for SED and SEE.

3.2 RAG-based Event Detection

In this section, we present our RAG-based event detection framework, as shown in Figure 2(B), which is designed to incorporate both the construction and maintenance of an event knowledge base (Sections 3.2.1 and 3.2.3) and the use of RAG to leverage the knowledge base as external global guidance for event detection (Section 3.2.2). To ensure computational efficiency, we rely not on all messages but only on the key messages sampled in Section 3.1. Take M_t as an example, the final input set for detection is thus $A = \{a_1, a_2, \dots\}$, where each a_k serves as the detection unit in subsequent steps. We present the algorithm for this section in Algorithm 1.

3.2.1 Evaluation-LLM for Knowledge Base Construction. We dynamically build and expand a knowledge base during detection. This knowledge base stores structured information about previously detected events, including their names, salient keywords, and embeddings. Formally, the knowledge base for M_t is defined as:

$$\mathcal{KB}^{(t)} = \{e = (\mathrm{Name}_e,\ \mathrm{Keywords}_e,\ p_e)\}, \tag{8}$$

$$p_e = \mathrm{Enc}(\mathrm{Name}_e \mid\mid \mathrm{Keywords}_e) \in \mathbb{R}^d, \tag{9}$$

, where Name_e denotes the name of e, $\mathrm{Keywords}_e = \{w_1,\ w_2, \dots\}$ is the keyword set of e, and $p_e \in \mathbb{R}^d$ is the event embedding. The embedding is obtained by encoding both the name and keywords. Each e is stored as an independent chunk in the knowledge base, enabling efficient retrieval and expansion during detection.

The event information, i.e., Name_e and $\mathrm{Keywords}_e$, is extracted from key messages using an Evaluation-LLM guided by our designed Prompt_E. The Prompt_E includes (i) a task description ("infer the event names discussed in the provided social media comments and extract keywords related to the event"), (ii) specific execution rules, and (iii) output requirements specifying a structured JSON format. We provide the complete Prompt_E in Appendix A.1.

When new messages a_k arrive and cannot be aligned with any existing event in the knowledge base, the Evaluation-LLM is again invoked to generate a new event chunk:

$$e_{\mathrm{new}} = \mathrm{LLM}_{\mathrm{Prompt}_E}(a_k) = (\mathrm{Name}_{\mathrm{new}},\ \mathrm{Keywords}_{\mathrm{new}},\ p_{\mathrm{new}}), \tag{10}$$

where a_k is the aggregated message from the new anchor. The new event e_{new} is then inserted into the knowledge base:

$$\mathcal{KB}^{(t)} \ \leftarrow\ \mathcal{KB}^{(t)} \cup \{e_{\mathrm{new}}\}. \tag{11}$$

Through this incremental process, the knowledge base gradually accumulates structured information about various events, providing powerful global semantic guidance for detection.

3.2.2 Detection-LLM for SED. To enhance the capability of LLMs in social event detection, we design a RAG-based paradigm. Given an incoming aggregated message a_k, RAG first queries the knowledge base $\mathcal{KB}^{(t)}$ to retrieve the most semantically relevant candidate events by event embedding. Retrieval of RAG is performed using cosine similarity. Formally, for each event e in $\mathcal{KB}^{(t)}$:

$$r(e \mid a_k) = \frac{\mathrm{Enc}(a_k)^\top p_e}{\|\mathrm{Enc}(a_k)\|_2 \cdot \|p_e\|_2}, \tag{12}$$

where $r(e \mid a_k)$ represents the degree of correlation between event e and message a_k in the knowledge base, p_e represents the embedding of e, and Enc is the embedding model used to obtain p_e. Sort the correlation of all events in the knowledge base from high to low, and the top-q events are selected as the retrieval events:

$$\mathcal{N}_{E(a_k)} = \underset{e \in \mathcal{KB}^{(t)}}{\arg\max}^{q}\ \{r(e \mid a_k) \mid r(e \mid a_k) \geq \gamma\}, \tag{13}$$

where $\mathcal{N}_{E(a_k)}$ represents the retrieval event set, and γ represents the similarity threshold of RAG.

Algorithm 1: RAG-based Event Detection of M_t

```
Input: Key message set $A = \{a_1, a_2, \dots\}$, empty knowledge
       base $\mathcal{KB}^{(t)}$, buffer threshold $\theta$
Output: Event label $\{y_{m_i}\}$ for each $m_i$ in $\mathrm{M}_t$
1  for each aggregated message $a_k \in A$ do
       // – Step 1: Retrieval –
2      Encode $a_k$ and compute similarity $r(e \mid a_k)$ for all
         $e \in \mathcal{KB}^{(t)}$ via Eq.12;
3      Select top-$q$ events above threshold via Eq.13;
       // – Step 2: Detection –
4      Query Detection-LLM via Eq.14;
5      if Detection-LLM outputs $e \in \mathcal{N}_{E(a_k)}$ then
6          Assign $y_{a_k} = e$ and append $a_k$ to buffer $\mathcal{B}_e$;
7      else
           // New event detected
8          $e_{\mathrm{new}} \leftarrow$ Query Evaluation-LLM via Eq.10;
9          Update $\mathcal{KB}^{(t)}$ via Eq.11;
10         Initialize buffer $\mathcal{B}_{e_{\mathrm{new}}} \leftarrow \{a_k\}$;
11         Assign $y_{a_k} = e_{\mathrm{new}}$;
       // – Step 3: Knowledge Base Maintenance –
12     for each $e \in \mathcal{KB}^{(t)}$ do
13         if $|\mathcal{B}_e| \geq \theta$ then
14             Generate refreshed $(\mathrm{Keywords}_e,\ p_e)$ via
                 $\mathrm{LLM}_{\mathrm{Prompt}_E}(\mathcal{B}_e)$ in Eq.10;
15             Update $\mathcal{KB}^{(t)}$ via Eq.11;
16             Clear buffer $\mathcal{B}_e$;
17 return $\{y_{m_i}\} \leftarrow$ Eq.15
```

The Detection LLM is then prompted with both the query message a_k and the $\mathcal{N}_{E(a_k)}$. Our designed Prompt_D consists of (i) a task description ("determine which one event the input belongs to in the knowledge base"), (ii) specific execution rules, (iii) output requirements specifying a structured JSON format, and (iv) the retrieved event set as global semantic guidance. We provide the complete Prompt_D in Appendix A.2. The detection decision is formalized as:

$$y_{a_k} = \mathrm{LLM}_{\mathrm{Prompt}_D}(a_k,\ \mathcal{N}_{E(a_k)}) = \begin{cases} e \in \mathcal{N}_{E(a_k)} \\ \mathrm{Others} \end{cases}, \tag{14}$$

where y_{a_k} is the event a_k discussed obtained by Detection-LLM based on Prompt_D and the knowledge base. It is worth noting that when the retrieval event set is empty or irrelevant, the Detection-LLM output is "Others". For any a_k predicted as "Others", we treat it as belonging to a previously unseen event e_{new}, i.e., $y_{a_k} = e_{\mathrm{new}}$. In this case, a_k is forwarded to the Evaluation-LLM (Section 3.2.1), which generates the event name and keywords of e_{new}, and the knowledge base is updated accordingly to enable Detection-LLM to identify subsequent messages related to e_{new}.

Finally, according to y_{a_k}, the event label of the original message is obtained as:

$$y_{m_i} = y_{a_k}, \quad \forall\, m_i \in \mathcal{A}_k, \tag{15}$$

where $\mathcal{A}_k$ is the corresponding anchor of a_k. Through this RAG-based paradigm, our framework effectively incorporates structured

Algorithm 2: Structural Entropy-based Event Evolution

```
   Input: Daily knowledge bases $\{\mathcal{KB}^{(1)},\dots,\mathcal{KB}^{(T)}\}$
   Output: Aligned events $\{\mathcal{E}^{*(1)},\dots,\mathcal{E}^{*(T)}\}$
1  for $t \leftarrow 1$ to $T$ do
       // − Step 1: Graph Construction −
2      Construct graph $G_t$ with $\mathcal{E}_{\text{inh}}^{t-1}$ via Eq.16 and Eq.17;
       // − Step 2: Structural Entropy Minimization −
3      Enforce no-merge constraint for inherited nodes;
4      Apply structural entropy minimization in $G_t$;
5      Obtain aligned events $\mathcal{E}^{*(t)} = \{e_1^*, e_2^*, \dots\}$;
       // − Step 3: Inheritance Mechanism −
6      $\mathcal{E}_{\text{inh}}^{t} \leftarrow \mathcal{E}^{*(t)}$;
       // − Step 4: Forgetting Mechanism −
7      for each $e_i^* \in \mathcal{E}^{*(t)}$ do
8          if $e_i^*$ is not supported by ordinary nodes in partitioning then
9              $\mathcal{E}_{\text{inh}}^{t} \leftarrow \mathcal{E}_{\text{inh}}^{t} \setminus e_i^*$;
10 return $\{\mathcal{E}^{*(1)},\dots,\mathcal{E}^{*(T)}\}$
```

event information as global semantic guidance, significantly enhancing LLMs to perform accurate and efficient SED.

3.2.3 Knowledge Base Maintenance over Time (KBM). To prevent the event information in the knowledge base from becoming outdated during long-term detection, we design a maintenance mechanism that periodically refreshes event information with new messages. Specifically, for each event $e \in \mathcal{KB}^{(t)}$, we maintain a buffer $\mathcal{B}_e$ to record the set of messages that have been assigned to e by the detection process. When the number of messages in the buffer exceeds a predefined threshold θ, the buffered messages are concatenated and forwarded to the Evaluation-LLM. The LLM generates refreshed event keywords, and we then update $\mathcal{KB}^{(t)}$. After the update, the buffer is cleared, and the system continues to collect messages. Once the buffer again reaches θ, the update process is repeated. Through this threshold-triggered refresh mechanism, our framework ensures that event information remains up-to-date and robust even in high-volume, long-duration social streams.

3.3 Structural Entropy-based Event Evolution

In this section, we present a structural entropy-based SEE framework that tracks changes of event keywords over time, as shown in Figure 2(C). After daily SED, we obtain a structured knowledge base containing event names and their keywords. However, these knowledge bases cannot directly capture event evolution due to two issues: (i) Misaligned event granularity. During detection, for popular events, the LLM tends to generate fine-grained subcategories; conversely, for cold events, it often generates coarse-grained categories. (ii) Daily initialization. Since event detection is performed with a newly initialized knowledge base each day, it becomes difficult to determine which events in different daily knowledge bases correspond to the evolution of the same real-world event. To address these challenges, we design an SEE framework comprising four components: graph construction, structural entropy minimization, an inheritance mechanism, and a forgetting mechanism. The first two components align event granularity, while the latter two ensure temporal continuity of evolving events. We present the algorithm for this section in Algorithm 2 and describe it as follows.

Graph Construction. At each day t, given the $\mathcal{KB}^{(t)} = \{e_1, e_2, \dots\}$, where each event e_i is represented by its keyword set Keywords_{e_i} and embedding p_{e_i}, we construct a weighted undirected graph $G_t = (V_t, E_t, W_t)$. The node set V_t consists of both the events in the current knowledge base $\mathcal{KB}^{(t)}$ and those inherited from the previous $\mathcal{KB}^{(t-1)}$ (see the inheritance mechanism below). Formally,

$$V_t = \begin{cases} \{e_1, e_2, \dots, e_{n_t}\}, & t = 1, \\ \{e_1, e_2, \dots, e_{n_t}\} \cup \mathcal{E}_{\text{inh}}^{t-1}, & t > 1, \end{cases} \tag{16}$$

where $\mathcal{E}_{\text{inh}}^{t-1}$ is the set of inherited event nodes, and each node also has a keyword set and an embedding. Edges of G_t are established between event nodes sharing at least one keyword. Formally,

$$E_t = \{(i,j) \mid i < j,\ \text{Keywords}_{e_i} \cap \text{Keywords}_{e_j} \neq \emptyset\}, \tag{17}$$

where e_i and e_j are events in V_t. Furthermore, the weight of each edge is the cosine similarity of the embeddings of the two events it connects. By constructing such graphs at each time t, we then use the structural entropy to model event evolution.

Structural Entropy Minimization. Given the constructed graph $G_t = (V_t,\ E_t,\ W_t)$, we apply structural information theory [19] to align event granularity and track event evolution. Structural information theory encodes a graph by an encoding tree $\mathcal{T}$ that induces a partitioning $\pi_t = \{c_1,\ c_2,\ \dots\}$ of V_t. The structural entropy of G_t with respect to $\mathcal{T}$ is defined as

$$H^{\mathcal{T}}(G_t;\pi_t) = -\sum_{\alpha\in\mathcal{T}} \frac{g_\alpha}{vol(G_t)} \log_2 \frac{vol(\alpha)}{vol(\alpha^-)}, \tag{18}$$

where α is a non-root node in $\mathcal{T}$ and α^- is the parent node of α. Structural information theory states that the $\mathcal{T}^*$ that minimizes structural entropy corresponds to the optimal partitioning. Following [19], we employ their structural entropy minimization algorithm to obtain the optimal partitioning of G_t. Formally,

$$\pi_t^* = \{c_1^*,\ c_2^*,\ \dots\} = \arg\min_{\pi_t\in\mathcal{P}(V_t),\ \mathcal{T}} H^{\mathcal{T}}(G_t;\pi_t), \tag{19}$$

where $\mathcal{P}(V_t)$ is all partitioning of V_t. Each partitioning c_i^* is regarded as an aligned event e_i^* at time t. For each e_i^*, we collect the keywords of all nodes within c_i^* (excluding inherited nodes) and select the top-15 keywords by frequency as $\text{keywords}_{e_i^*}$. During minimization, multiple fine-grained events of a popular event tend to be grouped into one aligned event, while sparse cold events are preserved separately. This effect automatically aligns the event granularity with the original $\mathcal{KB}^{(t)}$. Furthermore, aligned event in $\{e_i^*\}$ are categorized into two types: if its partitioning contains inherited node from time t−1, it is interpreted as the evolution of a previously existing event; otherwise, it is a newly emerging event at time t.

Inheritance Mechanism (IM). To continuously track event evolution over time, we design an inheritance mechanism. Specifically, for each aligned event with its keywords obtained at time $t-1$ through structural entropy minimization, we introduce a corresponding inherited node. These inherited nodes participate in the

Table 1: Social event detection results on Events2012. * marks results acquired with the ground truth message labels. The best results are bolded, the second-best results are underlined, and the proposed method is marked in an orange background.

Blocks	M_1			M_2			M_3			M_4			M_5			M_6			M_7		
Metric	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI
KPGNN*	0.39	0.37	0.07	0.79	0.78	0.76	0.76	0.74	0.58	0.67	0.64	0.29	0.73	0.71	0.47	0.82	0.79	0.72	0.55	0.51	0.12
QSGNN*	0.43	0.41	0.07	0.81	0.80	0.77	0.78	0.76	0.59	0.71	0.68	0.29	0.75	0.73	0.48	0.83	0.80	0.73	0.57	0.54	0.12
SBERT*	0.40	0.38	0.03	0.86	0.85	0.73	0.88	0.87	0.68	0.82	0.80	0.36	0.86	0.85	0.61	0.86	0.83	0.53	0.64	0.61	0.09
CP-Tuning*	0.50	0.48	0.06	0.87	0.86	0.73	0.85	0.84	0.66	0.79	0.76	0.27	0.84	0.83	0.59	0.87	0.84	0.58	0.69	0.67	0.30
PromptSED*	0.51	0.50	0.12	0.89	0.88	0.79	0.87	0.86	0.71	0.83	0.81	0.30	0.86	0.85	0.60	0.88	0.86	0.66	0.70	0.68	0.31
HISEvent	0.38	0.37	0.09	0.90	0.89	0.88	0.90	0.89	0.79	0.77	0.76	0.52	0.83	0.82	0.63	0.89	0.88	0.84	0.64	0.63	**0.36**
RagSEDE	**0.55**	**0.53**	**0.27**	**0.95**	**0.95**	**0.93**	**0.95**	**0.95**	**0.94**	**0.90**	**0.89**	**0.68**	**0.93**	**0.93**	**0.90**	**0.97**	**0.96**	**0.97**	**0.75**	**0.73**	0.26
Promotion	↑.04	↑.03	↑.15	↑.05	↑.06	↑.05	↑.05	↑.06	↑.15	↑.07	↑.08	↑.16	↑.07	↑.08	↑.27	↑.08	↑.08	↑.13	↑.05	↑.05	↓.10

Blocks	M_8			M_9			M_{10}			M_{11}			M_{12}			M_{13}			M_{14}		
Metric	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI
KPGNN*	0.80	0.76	0.60	0.74	0.71	0.46	0.80	0.78	0.70	0.74	0.71	0.49	0.68	0.66	0.48	0.69	0.67	0.29	0.69	0.65	0.42
QSGNN*	0.79	0.75	0.59	0.77	0.75	0.47	0.82	0.80	0.71	0.75	0.72	0.49	0.70	0.68	0.49	0.68	0.66	0.29	0.68	0.66	0.41
SBERT*	0.88	0.86	0.65	0.85	0.83	0.47	0.87	0.85	0.62	0.84	0.82	0.49	0.86	0.85	0.63	0.73	0.70	0.24	0.79	0.77	0.40
CP-Tuning*	0.85	0.82	0.59	0.82	0.80	0.48	0.85	0.83	0.69	0.80	0.78	0.48	0.80	0.79	0.46	0.72	0.70	0.36	0.77	0.75	0.51
PromptSED*	0.87	0.85	0.59	0.86	0.83	0.56	0.86	0.85	0.71	0.84	0.82	0.51	0.86	0.85	0.52	0.74	0.71	0.54	0.80	0.78	0.53
HISEvent	0.82	0.81	0.68	0.89	0.88	**0.65**	0.91	0.90	0.87	0.85	0.84	0.66	0.87	0.87	0.82	0.75	0.74	0.39	0.83	0.82	0.71
RagSEDE	**0.96**	**0.95**	**0.92**	**0.91**	**0.90**	**0.65**	**0.96**	**0.96**	**0.97**	**0.95**	**0.94**	**0.96**	**0.91**	**0.91**	**0.83**	**0.88**	**0.87**	**0.77**	**0.93**	**0.92**	**0.92**
Promotion	↑.08	↑.09	↑.24	↑.02	↑.02	-	↑.05	↑.06	↑.10	↑.10	↑.10	↑.30	↑.04	↑.04	↑.01	↑.13	↑.13	↑.23	↑.10	↑.10	↑.21

Blocks	M_{15}			M_{16}			M_{17}			M_{18}			M_{19}			M_{20}			M_{21}		
Metric	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI
KPGNN*	0.58	0.54	0.17	0.79	0.77	0.66	0.70	0.68	0.43	0.68	0.66	0.47	0.73	0.71	0.51	0.72	0.68	0.51	0.60	0.57	0.20
QSGNN*	0.59	0.55	0.17	0.78	0.76	0.65	0.71	0.69	0.44	0.70	0.68	0.48	0.73	0.70	0.50	0.73	0.69	0.51	0.61	0.58	0.21
SBERT*	0.70	0.67	0.17	0.81	0.78	0.50	0.78	0.77	0.35	0.82	0.81	0.52	0.84	0.83	0.54	0.83	0.80	0.52	0.72	0.70	0.24
CP-Tuning*	0.64	0.61	0.31	0.80	0.77	0.74	0.76	0.75	0.58	0.76	0.75	0.44	0.82	0.81	0.43	0.83	0.79	0.55	0.71	0.69	0.40
PromptSED*	0.67	0.64	0.34	0.82	0.81	0.75	0.80	0.79	0.61	0.82	0.80	0.49	0.84	0.83	0.50	0.86	0.83	0.57	0.74	0.72	0.41
HISEvent	0.69	0.67	0.27	0.87	0.86	0.83	0.77	0.76	0.56	0.74	0.73	0.64	0.85	0.84	0.60	0.82	0.80	**0.67**	0.73	0.73	**0.46**
RagSEDE	**0.92**	**0.91**	**0.97**	**0.92**	**0.92**	**0.86**	**0.93**	**0.93**	**0.95**	**0.87**	**0.86**	**0.69**	**0.90**	**0.89**	**0.76**	**0.87**	**0.84**	**0.67**	**0.78**	**0.76**	0.38
Promotion	↑.22	↑.24	↑.63	↑.05	↑.06	↑.03	↑.13	↑.14	↑.34	↑.05	↑.05	↑.05	↑.05	↑.05	↑.16	↑.01	↑.01	-	↑.04	↑.03	↓.08

subsequent graph construction and structural entropy minimization at time t. Importantly, these inherited nodes are prohibited from being merged into the same partitioning during structural entropy minimization. This no-merge constraint ensures that each previously detected event is preserved as an independent evolving entity, even if it expands by absorbing new nodes. Through this inheritance mechanism, our method can effectively identify the onset of events and track the evolution of event keywords.

Forgetting Mechanism (FM). For inherited nodes participating in graph construction, we designed a forgetting mechanism to avoid outdated events from persisting as noise. Specifically, during the structural entropy minimization at time t, if an inherited node from $t-1$ receives no support from ordinary event nodes (i.e., no other nodes are assigned to the partitioning of this inherited node), we regard the inherited event as outdated. Such events are excluded from inheritance in the next time step $t+1$. Through this forgetting mechanism, our method can effectively locate the end of events.

4 Experiments

We conduct experiments to validate the effectiveness and efficiency of the proposed RagSEDE. In addition, we conduct ablation studies, hyperparameter studies, case studies, and visualization studies (Appendix G) to demonstrate the superiority of RagSEDE.

4.1 Experimental Setups

4.1.1 Evaluation Metrics. To assess the effectiveness of SED, we follow previous studies [4, 52] and use three clustering metrics to measure the consistency between the detected event clusters and the ground truth clusters: Normalized Mutual Information (NMI) [10], Adjusted Mutual Information (AMI) [42], and Adjusted Rand Index (ARI) [42]. To assess the effectiveness of SEE, we use the popular C_v metric [39] to evaluate the keyword coherence of evolving events and employ the Topic Diversity (TD) metric [8] to measure the diversity of events captured during the evolution process. We take the average C_v and TD over all time slices.

4.1.2 Datasets. We conduct experiments on two public Twitter datasets: Event2012 (68,841 English messages, 503 events) [26] and Event2018 (64,516 French messages, 257 events) [25]. Following processing in [4], the datasets are split into daily message blocks. The detailed statistical information is presented in Appendix D.

4.1.3 Baselines. To evaluate the SED performance of RagSEDE, we compare it with two GNN-based methods (**KPGNN** [3] and **QSGNN** [35]), three PLM-based methods (**SBERT** [34], **CP-Tuning** [51], and **PromptSED** [52]), and one structural entropy-based method (**HISEvent** [4]). For SEE performance of RagSEDE, since there are no established baselines, we compare it with topic modeling approaches [46, 47] that can also extract event keywords. These include three non-dynamic models (**ProdLDA** [41], **DecTM** [45], and **TSCTM** [48]) and two dynamic models (**CFDTM** [44] and **BERTopic** [14]). Appendix E and F show additional descriptions.

4.2 Main Results

4.2.1 Effectiveness Analysis of Unsupervised SED. The SED evaluation results of RagSEDE are reported in Tables 1 and 2, where both Detection-LLM and Evaluation-LLM are instantiated with deepseek-r1:32b. On the English dataset, RagSEDE consistently outperforms all baselines across nearly all message blocks, achieving maximum gains of 0.22 in NMI, 0.24 in AMI, and 0.63 in ARI (e.g., block M_{15}). These improvements highlight the effectiveness of global event guidance in forming more accurate event clusters. Importantly, RagSEDE maintains stable promotion in both dense message blocks (e.g., M_1, M_7, M_{12}) and sparse message blocks (e.g., M_{16}, M_{20}), demonstrating robustness to varying data distributions. On the French dataset, RagSEDE achieves the best or second-best performance in almost all blocks. The observed marginal decrease is due to the use of a relatively small 32B LLM, which has limited

Table 2: Social event detection results on Events2018. * marks results acquired with the ground truth message labels. The best results are bolded, the second-best results are underlined, and the proposed method is marked in an orange background.

Blocks	M_1			M_2			M_3			M_4			M_5			M_6			M_7			M_8		
Metric	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI
KPGNN*	0.54	0.54	0.17	0.56	0.55	0.18	0.52	0.55	0.15	0.55	0.55	0.17	0.58	0.57	0.21	0.59	0.57	0.21	0.63	0.61	<u>0.30</u>	0.58	0.57	0.20
QSGNN*	0.57	0.56	0.18	0.58	0.57	0.19	0.57	0.56	0.17	0.58	0.57	0.18	0.61	0.59	0.23	0.60	0.59	0.21	0.64	0.63	<u>0.30</u>	0.57	0.55	0.19
SBERT*	0.60	0.60	0.20	0.62	0.61	0.29	0.64	0.63	0.34	0.61	0.60	0.23	<u>0.77</u>	<u>0.76</u>	0.47	0.73	0.73	<u>0.41</u>	0.66	0.65	0.29	<u>0.76</u>	0.75	0.50
CP-Tuning*	<u>0.74</u>	<u>0.74</u>	0.36	0.65	0.64	0.10	0.57	0.56	0.31	0.53	0.52	0.30	0.62	0.61	0.35	0.65	0.64	0.31	0.50	0.50	0.11	0.60	0.59	0.33
PromptSED*	**0.75**	**0.75**	0.38	0.66	0.65	0.28	0.63	0.62	0.35	0.58	0.57	0.34	0.65	0.63	0.36	0.68	0.67	0.38	0.55	0.54	0.27	0.69	0.68	0.39
HISEvent	<u>0.74</u>	<u>0.74</u>	**0.58**	<u>0.73</u>	<u>0.73</u>	<u>0.60</u>	**0.72**	**0.72**	**0.52**	<u>0.67</u>	<u>0.66</u>	<u>0.48</u>	0.74	0.73	<u>0.56</u>	**0.80**	**0.79**	**0.66**	**0.79**	**0.78**	**0.59**	**0.82**	**0.82**	**0.75**
RagSEDE	0.72	0.71	<u>0.55</u>	**0.77**	**0.76**	**0.68**	<u>0.71</u>	<u>0.70</u>	<u>0.49</u>	**0.71**	**0.69**	**0.65**	**0.81**	**0.80**	**0.72**	<u>0.78</u>	<u>0.77</u>	**0.66**	<u>0.72</u>	<u>0.71</u>	**0.59**	**0.82**	<u>0.81</u>	<u>0.67</u>
Promotion	↓.03	↓.04	↓.03	↑.04	↑.03	↑.08	↓.01	↓.02	↓.03	↑.04	↑.03	↑.17	↑.04	↑.04	↑.16	↓.02	↓.02	-	↓.07	↓.07	-	-	↓.01	↓.08

Blocks	M_9			M_{10}			M_{11}			M_{12}			M_{13}			M_{14}			M_{15}			M_{16}		
Metric	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI
KPGNN*	0.48	0.46	0.10	0.57	0.56	0.18	0.54	0.53	0.16	0.55	0.56	0.17	0.60	0.60	0.28	0.66	0.65	0.43	0.60	0.58	0.25	0.52	0.50	0.13
QSGNN*	0.52	0.46	0.13	0.60	0.58	0.19	0.60	0.59	0.20	0.61	0.59	0.20	0.59	0.58	0.27	0.68	0.67	0.44	0.63	0.61	0.27	0.51	0.50	0.13
SBERT*	<u>0.64</u>	<u>0.63</u>	0.23	0.74	0.72	0.39	<u>0.72</u>	0.70	0.31	0.77	0.76	0.54	0.66	<u>0.65</u>	0.34	0.69	0.68	0.43	0.72	0.71	0.40	0.66	<u>0.65</u>	0.25
CP-Tuning*	0.49	0.47	0.31	0.64	0.61	0.34	0.65	0.63	0.35	0.64	0.62	0.33	0.52	0.50	0.31	0.53	0.52	0.30	0.59	0.58	0.33	0.48	0.46	0.28
PromptSED*	0.55	0.53	0.29	0.65	0.63	0.28	0.67	0.65	0.35	0.68	0.67	0.36	0.57	0.55	0.31	0.60	0.59	0.33	0.66	0.64	0.34	0.55	0.53	0.30
HISEvent	**0.65**	**0.64**	**0.42**	**0.77**	**0.76**	**0.66**	<u>0.72</u>	<u>0.71</u>	<u>0.44</u>	<u>0.84</u>	<u>0.83</u>	<u>0.80</u>	**0.78**	**0.78**	**0.86**	**0.83**	**0.82**	**0.75**	<u>0.76</u>	<u>0.75</u>	<u>0.61</u>	**0.70**	**0.69**	<u>0.38</u>
RagSEDE	**0.65**	<u>0.63</u>	<u>0.32</u>	<u>0.76</u>	<u>0.74</u>	<u>0.57</u>	**0.78**	**0.76**	**0.62**	**0.87**	**0.86**	**0.83**	<u>0.67</u>	<u>0.65</u>	<u>0.51</u>	<u>0.73</u>	<u>0.71</u>	<u>0.64</u>	**0.79**	**0.78**	**0.70**	<u>0.67</u>	0.64	**0.39**
Promotion	-	↓.01	↓.10	↓.01	↓.02	↓.09	↑.06	↑.05	↑.18	↑.03	↑.03	↑.03	↓.11	↓.13	↓.35	↓.10	↓.11	↓.11	↑.03	↑.03	↑.09	↓.03	↓.05	↑.01

Table 3: Social event Evolution results. In dynamic methods, the best results are bolded, the second-best are underlined.

	Events2012				Events2018			
Method	C_v	TD	Avg	Dynamic	C_v	TD	Avg	Dynamic
ProdLDA	0.46	0.48	0.47	×	0.40	0.74	0.57	×
DecTM	0.48	0.63	0.56	×	0.39	0.85	0.62	×
TSCTM	0.44	0.93	0.69	×	0.36	0.96	0.66	×
CFDTM	<u>0.65</u>	0.14	0.40	✓	<u>0.66</u>	0.34	0.50	✓
Bertopic	0.61	<u>0.42</u>	<u>0.52</u>	✓	0.59	0.39	0.49	✓
RagSEDE w/o IM	0.53	0.90	0.72	×	0.53	0.87	0.70	×
RagSEDE w/o FM	**0.76**	0.28	<u>0.52</u>	✓	**0.69**	<u>0.42</u>	<u>0.56</u>	✓
RagSEDE	0.49	**0.88**	**0.69**	✓	0.52	**0.87**	**0.70**	✓

Table 4: Ablation results for SED on two datasets. The best results are bolded, the second-best are underlined.

Blocks	M_3(Event2012)			M_{10}(Event2012)			M_8(Event2018)			M_{15}(Event2018)		
Metric	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI	NMI	AMI	ARI
RagSEDE w/o KMS	.84	.81	.61	.93	.92	<u>.88</u>	.72	.67	.41	.73	.69	.55
RagSEDE w/o $\mathcal{KB}$	.86	<u>.85</u>	.57	.91	.89	.72	.73	.68	.34	.68	.64	.28
RagSEDE w/o KBM	<u>.95</u>	**.95**	<u>.94</u>	<u>.96</u>	<u>.96</u>	**.97**	<u>.81</u>	.79	<u>.67</u>	.77	.76	.68
RagSEDE (deepseek-r1:32b)	<u>.95</u>	**.95**	<u>.94</u>	<u>.96</u>	<u>.96</u>	**.97**	**.82**	**.81**	<u>.67</u>	<u>.79</u>	<u>.78</u>	.70
RagSEDE + deepseek-r1:70b	**.96**	**.95**	**.95**	<u>.96</u>	<u>.96</u>	**.97**	<u>.81</u>	.79	.64	<u>.79</u>	<u>.78</u>	<u>.72</u>
RagSEDE + GPT-4o-mini	**.96**	**.95**	**.95**	**.97**	**.97**	**.97**	**.82**	**.81**	**.68**	**.81**	**.80**	**.77**
RagSEDE + GPT-4o	**.96**	**.95**	**.95**	**.97**	<u>.96</u>	**.97**	**.82**	<u>.80</u>	**.68**	<u>.79</u>	<u>.78</u>	.68

capacity for French comprehension. In practical deployment, selecting language-specific LLMs can further enhance performance. Furthermore, beyond strong SED performance, RagSEDE uniquely supports SEE, which none of the baseline methods provide.

4.2.2 Effectiveness Analysis of Unsupervised SEE. The SEE evaluation results of RagSEDE are reported in Table 3, and all values are with two decimal precision. For non-dynamic methods, we execute the algorithms independently on each daily block. When a method achieves a high C_v but a very low TD value, the keywords for different events exhibit significant overlap, meaning these events are actually the same real-world event (e.g., CFDTM, Bertopic). When a method achieves a very high TD but a low C_v, it excessively punishes keyword sharing across events, resulting in incoherent keywords that fail to adequately describe the events(e.g., TSCTM). The results reveal that RagSEDE achieves a better balance between C_v and TD, with its average score consistently surpassing all dynamic baselines. This advantage stems from maintaining a high-quality knowledge base during the SED process and from using structural entropy minimization to align event granularity. Moreover, unlike non-dynamic approaches that fail in real-world applications, RagSEDE can effectively track event evolution over time.

4.3 Ablation Study

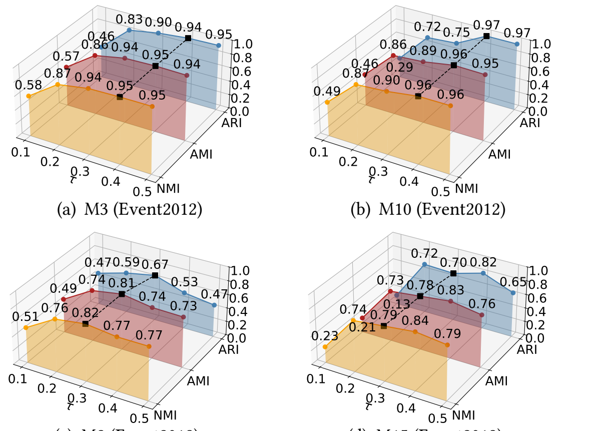
We conduct ablation studies on both datasets to further demonstrate the effectiveness of each module in RagSEDE. For SED, as shown in Table 4, removing the KMS, $\mathcal{KB}$, or KBM modules leads to varying degrees of performance degradation. This is because the KMS module samples key messages instead of all messages, thereby improving the quality of LLM queries; the KB module provides event information as global guidance, enhancing the LLM's capability; and the KBM module maintains the freshness of the knowledge base, which becomes particularly important for larger message blocks (e.g., M_8 and M_{15} in Events2018). For SEE, the results in Table 3 demonstrate that removing the IM module prevents RagSEDE from capturing the dynamic evolution of events, whereas removing the FM module results in insufficient event diversity, confirming the necessity of both modules. Finally, we ablate RagSEDE with different base LLMs. While larger LLMs naturally achieve higher accuracy, RagSEDE with a 32B LLM already delivers competitive performance while significantly reducing computational resources.

4.4 Hyperparameter Study

We conduct a hyperparameter study of the similarity threshold τ in the key message sampling module (Section 3.1.1), with the results shown in Figure 3. Figure 3 indicates that setting τ too low (e.g., $\tau = 0.1$) significantly decreases detection performance across all blocks, as messages from different events are aggregated into the same anchor and cannot be further distinguished during detection. As τ increases, performance improves, with the best results observed around $\tau = 0.4$ on the Event2012 and $\tau = 0.3$ on the Event2018. RagSEDE remains relatively robust to τ within this reasonable range. However, when τ is set too high, detection efficiency decreases. Therefore, selecting an appropriate τ enables a favorable balance between performance and efficiency.

4.5 Case Study

We conduct a case study to demonstrate RagSEDE's capability in SEE. Figure 4 presents the evolution keywords obtained by



(a) M3 (Event2012) (b) M10 (Event2012)
(c) M8 (Event2018) (d) M15 (Event2018)

Figure 3: Sensitivity of hyperparameter τ on four blocks.

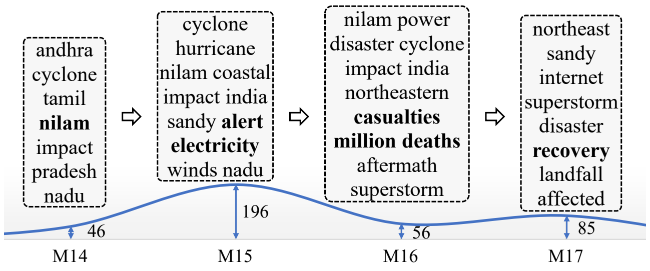

andhra cyclone tamil **nilam** impact pradesh nadu
cyclone hurricane nilam coastal impact india sandy **alert** **electricity** winds nadu
nilam power disaster cyclone impact india northeastern **casualties** **million deaths** aftermath superstorm
northeast sandy internet superstorm disaster **recovery** landfall affected
46
196
56
85
M14
M15
M16
M17

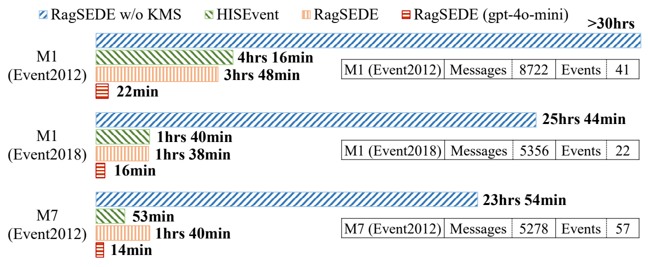
Figure 4: Case study for SEE of RagSEDE. Evolution keywords of an Event about Cyclone Nilam from the Event2012 dataset.

RagSEDE for the detected event Cyclone Nilam[2] over a four-day period. The curve in the figure shows changes in the event's discussion intensity. The bold keywords in the figure 4 indicate that RagSEDE not only correctly localizes the Nilam region on the first day, but also continuously tracks the impact and damages of the disaster in the following days (e.g., electrical issue in M_{15} and casualties in M_{16}). Finally, RagSEDE captures the post-disaster recovery process. This ability to consistently track events across time periods demonstrates the effectiveness of the inheritance and forgetting mechanisms in the evolution framework RagSEDE.

4.6 Efficiency Analysis

We report the running time of the proposed RagSEDE, RagSEDE without the key message sampling module, and the strongest baseline HISEvent on the three largest message blocks of datasets, as shown in Figure 5. The results demonstrate that incorporating the KMS module improves the efficiency of RagSEDE by up to 15 times, as it effectively avoids redundant LLM queries on semantically similar messages. Furthermore, compared to HISEvent, which incurs higher computational complexity, RagSEDE achieves superior efficiency on large-scale message blocks that more closely reflect real-world scenarios. Importantly, RagSEDE employs the locally deployed deepseek-r1:32b, constrained by our local computational resources. When the backbone LLM is replaced with gpt-4o-mini, we observe a further improvement in efficiency.

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



M1 (Event2012)	Messages	8722	Events	41
M1 (Event2018)	Messages	5356	Events	22
M7 (Event2012)	Messages	5278	Events	57

Figure 5: Running time comparison on the largest blocks.

5 Related Work

In the past decade, SED [2] has been extensively studied. Early approaches relied on incremental clustering [1, 16], topic modeling [43, 50, 57], and community detection [12, 21, 22] to identify events from social streams. While effective to some extent, these methods often suffer from limitations in scalability, adaptability, and semantic expressiveness. With the emergence of GNNs, which are capable of modeling complex structural dependencies among messages, GNN-based methods have rapidly become the dominant approach for SED. These methods span a wide spectrum, including supervised approaches based on evidential learning and contrastive learning [3, 35, 36, 38], as well as unsupervised [15], cross-lingual [37], and multimodal [55] approaches. More recently, the rapid advancement of PLMs has enabled the use of their strong contextual reasoning capabilities in SED tasks. [20, 52] utilizes PLMs and appropriate prompts to obtain message embeddings, followed by clustering to obtain events. In addition, structural entropy [19, 27], as an unsupervised clustering method, has shown remarkable potential for social event analysis [4]. However, none of the above methods can track the evolution of detected events, and our method achieves this, which is our advantage.

6 Conclusion

In this paper, we propose RagSEDE, a novel unsupervised model for SED and SEE. First, RagSEDE introduces a key message sampling module that improves the quality of LLM queries while substantially reducing computational overhead. Second, RagSEDE proposes a new RAG-based event detection paradigm that leverages a knowledge base for global semantic guidance. Finally, RagSEDE leverages structural entropy to track the evolution of events over time. Extensive experiments on two benchmark datasets demonstrate that RagSEDE outperforms all baselines. In future work, we plan to extend this framework to multimodal scenes.

Acknowledgments

This work is supported by the NSFC through grants U25B2029, 62322202, 62441612, and 62432006, Beijing Natural Science Foundation through grant L253021, the Pioneer and Leading Goose R&D Program of Zhejiang through grant 2025C02044, Local Science and Technology Development Fund of Hebei Province Guided by the Central Government of China through grant 254Z9902G, Hebei Natural Science Foundation through grant F2024210008, Major Science Technology Special Projects of Yunnan Province through grants 202502AD080012 and 202502AD080006, and the Fundamental Research Funds for the Central Universities.

References

[1] Charu C Aggarwal and Karthik Subbian. 2012. Event detection in social streams. In *Proceedings of the 2012 SIAM international conference on data mining*. SIAM, 624–635.

[2] Farzindar Atefeh and Wael Khreich. 2015. A survey of techniques for event detection in twitter. *Computational Intelligence* 31, 1 (2015), 132–164.

[3] Yuwei Cao, Hao Peng, Jia Wu, Yingtong Dou, Jianxin Li, and Philip S Yu. 2021. Knowledge-preserving incremental social event detection via heterogeneous gnns. In *Proceedings of the web conference 2021*. 3383–3395.

[4] Yuwei Cao, Hao Peng, Zhengtao Yu, and Philip S Yu. 2024. Hierarchical and incremental structural entropy minimization for unsupervised social event detection. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 38. 8255–8264.

[5] Wanqiu Cui, Junping Du, Dawei Wang, Feifei Kou, and Zhe Xue. 2021. MVGAN: Multi-view graph attention network for social event detection. *ACM Transactions on Intelligent Systems and Technology (TIST)* 12, 3 (2021), 1–24.

[6] Sunhao Dai, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. Bias and unfairness in information retrieval systems: New challenges in the llm era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6437–6447.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

[8] Adji B Dieng, Francisco JR Ruiz, and David M Blei. 2020. Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics* 8 (2020), 439–453.

[9] 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 *kdd*, Vol. 96. 226–231.

[10] Pablo A Estévez, Michel Tesmer, Claudio A Perez, and Jacek M Zurada. 2009. Normalized mutual information feature selection. *IEEE Transactions on neural networks* 20, 2 (2009), 189–201.

[11] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*. 6491–6501.

[12] Mateusz Fedoryszak, Brent Frederick, Vijay Rajaram, and Changtao Zhong. 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.

[13] Santo Fortunato and Mark EJ Newman. 2022. 20 years of network community detection. *Nature Physics* 18, 8 (2022), 848–850.

[14] Maarten Grootendorst. 2022. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794* (2022).

[15] Yuanyuan Guo, Zehua Zang, Hang Gao, Xiao Xu, Rui Wang, Lixiang Liu, and Jiangmeng Li. 2024. Unsupervised social event detection via hybrid graph contrastive learning and reinforced incremental clustering. *Knowledge-Based Systems* 284 (2024), 111225.

[16] Linmei Hu, Bin Zhang, Lei Hou, and Juanzi Li. 2017. Adaptive online event detection in news streams. *Knowledge-Based Systems* 138 (2017), 105–112.

[17] Dan Knights, Michael Mozer, and Nicolas Nicolov. 2009. Detecting topic drift with compound topic models. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 3. 242–245.

[18] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* 33 (2020), 9459–9474.

[19] Angsheng Li and Yicheng Pan. 2016. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory* 62, 6 (2016), 3290–3339.

[20] Pu Li, Xiaoyan Yu, Hao Peng, Yantuan Xian, Linqin Wang, Li Sun, Jingyun Zhang, and Philip S Yu. 2024. Relational prompt-based pre-trained language models for social event detection. *ACM Transactions on Information Systems* 43, 1 (2024), 1–43.

[21] 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.

[22] Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018. Event detection via gated multilingual attention mechanism. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[23] 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), 145612.

[24] Congbo Ma, Zitai Qiu, Hu Wang, Jing Du, Shan Xue, Jia Wu, and Jian Yang. 2025. Enhanced Social Event Detection through Dynamically Weighted Meta-Paths Modeling. In *Companion Proceedings of the ACM on Web Conference 2025*. 1184–1188.

[25] Béatrice Mazoyer, Julia Cagé, Nicolas Hervé, and Céline Hudelot. 2020. A french corpus for event detection on twitter. In *Twelfth Language Resources and Evaluation Conference*. European Language Resources Association (ELRA), 6220–6227.

[26] Andrew J McMinn, Yashar Moshfeghi, and Joemon M Jose. 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.

[27] Hao Peng, Xiang Huang, Shuo Sun, Ruitong Zhang, and Xizhao Wang. 2026. Adaptive and robust DBSCAN with multi-agent reinforcement learning. *IEEE Transactions on Pattern Analysis & Machine Intelligence* (2026). doi:10.1109/TPAMI.2025.3648017

[28] Hao Peng, Jianxin Li, Qiran Gong, Yangqiu Song, Yuanxing Ning, Kunfeng Lai, and Philip S Yu. 2019. Fine-grained event categorization with heterogeneous graph convolutional networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 3238–3245.

[29] Hao Peng, Jianxin Li, Yangqiu Song, Renyu Yang, Rajiv Ranjan, Philip S Yu, and Lifang He. 2021. Streaming social event detection and evolution discovery in heterogeneous information networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 5 (2021), 1–33.

[30] Hao Peng, Ruitong Zhang, Shaoning Li, Yuwei Cao, Shirui Pan, and Philip S Yu. 2022. Reinforced, incremental and cross-lingual event detection from social messages. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 1 (2022), 980–998.

[31] Shengsheng Qian, Hong Chen, Dizhan Xue, Quan Fang, and Changsheng Xu. 2023. Open-world social event classification. In *Proceedings of the ACM web conference 2023*. 1562–1571.

[32] Shengsheng Qian, Dizhan Xue, Quan Fang, and Changsheng Xu. 2022. Integrating multi-label contrastive learning with dual adversarial graph neural networks for cross-modal retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 4 (2022), 4794–4811.

[33] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems* 36 (2023), 10299–10315.

[34] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3982–3992.

[35] Jiaqian Ren, Lei Jiang, Hao Peng, Yuwei Cao, Jia Wu, Philip S Yu, and Lifang He. 2022. From known to unknown: Quality-aware self-improving graph neural network for open set social event detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 1696–1705.

[36] Jiaqian Ren, Lei Jiang, Hao Peng, Zhiwei Liu, Jia Wu, and Philip S Yu. 2022. Evidential temporal-aware graph-based social event detection via dempster-shafer theory. In *2022 IEEE International Conference on Web Services (ICWS)*. IEEE, 331–336.

[37] Jiaqian Ren, Hao Peng, Lei Jiang, Zhifeng Hao, Jia Wu, Shengxiang Gao, Zhengtao Yu, and Qiang Yang. 2024. Toward cross-lingual social event detection with hybrid knowledge distillation. *ACM Transactions on Knowledge Discovery from Data* 18, 9 (2024), 1–36.

[38] Jiaqian Ren, Hao Peng, Lei Jiang, Zhiwei Liu, Jia Wu, Zhengtao Yu, and Philip S Yu. 2023. Uncertainty-guided boundary learning for imbalanced social event detection. *IEEE Transactions on Knowledge and Data Engineering* 36, 6 (2023), 2701–2715.

[39] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*. 399–408.

[40] Zhengliang Shi, Shuo Zhang, Weiwei Sun, Shen Gao, Pengjie Ren, Zhumin Chen, and Zhaochun Ren. 2024. Generate-then-Ground in Retrieval-Augmented Generation for Multi-hop Question Answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 7339–7353.

[41] Akash Srivastava and Charles Sutton. 2017. Autoencoding Variational Inference for Topic Models. In *5th International Conference on Learning Representations*.

[42] Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary?. In *Proceedings of the 26th annual international conference on machine learning*. 1073–1080.

[43] Yuan Wang, Jie Liu, Yalou Huang, and Xia Feng. 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 (2016), 1919–1933.

[44] Xiaobao Wu, Xinshuai Dong, Liangming Pan, Thong Nguyen, and Luu Anh Tuan. 2024. Modeling Dynamic Topics in Chain-Free Fashion by Evolution-Tracking Contrastive Learning and Unassociated Word Exclusion. In *Findings of the Association for Computational Linguistics ACL 2024*. 3088–3105.

[45] Xiaobao Wu, Chunping Li, and Yishu Miao. 2021. Discovering topics in long-tailed corpora with causal intervention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. 175–185.
[46] Xiaobao Wu, Thong Nguyen, and Anh Tuan Luu. 2024. A survey on neural topic models: methods, applications, and challenges. *Artificial Intelligence Review* 57, 2 (2024), 18.
[47] Xiaobao Wu, Fengjun Pan, and Luu Anh Tuan. 2024. Towards the TopMost: A Topic Modeling System Toolkit. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*. 31–41.
[48] Xiaobao Wu, Luu Anh Tuan, and Xinshuai Dong. 2022. Mitigating Data Sparsity for Short Text Topic Modeling by Topic-Semantic Contrastive Learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 2748–2760.
[49] Yantuan Xian, Pu Li, Hao Peng, Zhengtao Yu, Yan Xiang, and Philip S Yu. 2025. Community detection in large-scale complex networks via structural entropy game. In *Proceedings of the ACM on Web Conference 2025*. 3930–3941.
[50] Chen Xing, Yuan Wang, Jie Liu, Yalou Huang, and Wei-Ying Ma. 2016. Hashtag-based sub-event discovery using mutually generative lda in twitter. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.
[51] Ziyun Xu, Chengyu Wang, Minghui Qiu, Fuli Luo, Runxin Xu, Songfang Huang, and Jun Huang. 2023. Making pre-trained language models end-to-end few-shot learners with contrastive prompt tuning. In *Proceedings of the sixteenth ACM international conference on web search and data mining*. 438–446.
[52] Xiaoyan Yu, Jiaqian Ren, Lei Jiang, Hao Peng, Zhifeng Hao, Li Sun, Kun Peng, Liehuang Zhu, and Philip S Yu. 2025. PromptSED: An evolving topic-enhanced prompting framework for incremental social event detection. *Neural Networks* (2025), 107772.
[53] Xiaoyan Yu, Yifan Wei, Shuaishuai Zhou, Zhiwei Yang, Li Sun, Hao Peng, Liehuang Zhu, and Philip S Yu. 2025. Towards effective, efficient and unsupervised social event detection in the hyperbolic space. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 13106–13114.
[54] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. 2024. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473* (2024).
[55] Sicheng Zhao, Yue Gao, Guiguang Ding, and Tat-Seng Chua. 2017. Real-time multimedia social event detection in microblog. *IEEE transactions on cybernetics* 48, 11 (2017), 3218–3231.
[56] Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, et al. 2024. Recommender systems in the era of large language models (llms). *IEEE Transactions on Knowledge and Data Engineering* 36, 11 (2024), 6889–6907.
[57] Xiangmin Zhou and Lei Chen. 2014. Event detection over twitter social media streams. *The VLDB journal* 23, 3 (2014), 381–400.

A PROMPT OF RagSEDE

In this appendix, we provide the complete prompts used in RagSEDE.

A.1 Prompt for Evaluation-LLM

This prompt is designed to guide the LLM in extracting structured event information, e.g., event name and keywords, from messages. The prompt contains role setting, task instruction, and output constraint, as shown in box Prompt$_E$.

A.2 Prompt for Detection-LLM

This prompt is designed to decide whether a message belongs to an existing event or a new event. The prompt contains input description, role setting, task instruction, output constraint, and guidance from the knowledge base, as shown in box Prompt$_D$. When asking LLMs, the location of {knowledge} will be replaced with relevant events retrieved from the knowledge base.

B GLOSSARY OF NOTATIONS

We summarize the notation used in this paper, along with their corresponding description, in Table 5.

C STRUCTURAL ENTROPY

Structural entropy [19] is a measure of the uncertainty of the graph structure. It represents the minimum number of bits required to encode a reachable vertex during a single-step random walk on the graph. Structural information theory utilizes the encoding tree $\mathcal{T}$ to measure the structure of the graph $G(V, E)$, which is as follows:

(1) Each node $\alpha \in \mathcal{T}$ corresponds to a partitioning of graph nodes $T_\alpha \subseteq V$. Significantly, the root node λ of $\mathcal{T}$ is associated with the entire set of graph nodes, $T_\lambda = V$. And for any leaf node γ of $\mathcal{T}$, $\mathcal{T}_\gamma$ contains exactly one graph node from V.
(2) For any non-leaf node α in $\mathcal{T}$, let its children be denoted as $\beta_1, ..., \beta_{N_\alpha}$, where N_α is the number of children of α. Then, $(T_{\beta_1}, ..., T_{\beta_{N_\alpha}})$ form a partitioning of T_α.

The structural entropy is defined under the graph G and the encoding tree T, as follows:

$$H^{\mathcal{T}}(G) = - \sum_{\alpha \in \mathcal{T},\ \alpha \neq \lambda} \frac{g_\alpha}{vol(G)} \log_2 \frac{vol(\alpha)}{vol(\alpha^-)}, \tag{20}$$

where α^- is the parent node of non-root node α in $\mathcal{T}$, the cut g_α is the weight sum of edges with exactly one endpoint in T_α, and

Prompt$_E$

"You are an event analysis assistant. Your task is to infer the event names discussed in the provided social media comments and extract keywords related to the event.

First, carefully read all the COMMENTS and understand the core content they discuss.

Second, summarize a concise and accurate EVENT NAME based on the COMMENTS.

Third, extract no more than 10 KEYWORDS related to the event from the comments, which should cover the core theme, characters, location, time, or other vital information about the event. Each KEYWORD must be a single word that appears in the COMMENTS.

Answer in JSON format, including EVENT-NAME (str) and KEYWORDS (list) attributes. Other than that, the answer must not include any other information."

Prompt$_D$

"The knowledge base contains EVENTs and corresponding KEYWORDs.

You are a social media comment classifier determining which one EVENT the INPUT belongs to in the knowledge base.

Answer in JSON format, including INPUT and EVENT attributes. Other than that, the answer must not include any other information.

When all knowledge base content is irrelevant to the INPUT, your EVENT answer must be 'Others'.

When the knowledge base is empty, your EVENT answer must be 'Others'.

Answers don't need to consider chat history.

Here is the knowledge base:

{knowledge}

The above is the knowledge base."

Table 5: Glossary of Notations.

Notation	Description
$\mathcal{S}$	Social stream, a temporal sequence of message blocks
M_t	The t-th message block in $\mathcal{S}$
m_i	A social message in block M_t
$\mathcal{A}_k$	The k-th anchor containing similar messages
τ	Similarity threshold in each anchor
λ	The trade-off parameter of sampling
p	The number of key messages selected from an anchor
a_k	The aggregated key message of anchor $\mathcal{A}_k$
y_{m_i}, y_{a_k}	Event label assigned to m_i and a_k
$\mathcal{KB}^{(t)}$	Knowledge base at day t used for M_t
e, p_e	Event and its corresponding representation in $\mathcal{KB}$
Name_e	Name of event e in $\mathcal{KB}$
$\mathrm{Keywords}_e$	Keyword set describing event e in $\mathcal{KB}$
$\mathrm{Enc}(\cdot)$	Embedding model for messages and events
$r(e \mid a_k)$	Relevance score between event e and a_k
γ	Similarity threshold in RAG retrieval
$\mathcal{N}_{E(a_i)}$	Top-q retrieved candidate events for a_i
$\mathcal{B}_e$	Message buffer for event e in knowledge base maintenance
θ	Threshold of buffered messages for $\mathcal{KB}^{(t)}$ update
G_t	Event graph at day t, with nodes, edges, and weights
$H^{\mathcal{T}}(G_t;\pi_t)$	Structural entropy of graph G_t with partitioning π_t
π_t^*	The optimal partitioning of G_t

the volume $vol(\alpha)$, $vol(\alpha^-)$, and $vol(\lambda)$ denote the degrees sum of graph nodes within T_α, T_α^-, and T_λ, respectively.

The encoding tree corresponding to the minimum structural entropy is regarded as the optimal encoding tree $\mathcal{T}^*$, which captures the essential structure of the graph. Without requiring supervision or a predefined number of clusters, the two-dimensional structural entropy minimization algorithm obtains such an optimal encoding tree by an MERGE operator, described as follows:

Definition C.1 (MERGE Operator [19]). Given an encoding tree $\mathcal{T}$ and two of its non-root nodes α_i and α_j, the operation $\mathrm{MERGE}(\alpha_i, \alpha_j)$ removes α_i and α_j from $\mathcal{T}$ and introduces a new node α_n into $\mathcal{T}$. The children of α_n are the union of the children of α_i and α_j, and the parent of α_n is the root node.

Executing the MERGE operator will change the structural entropy of $\mathcal{T}$. After initializing the encoding tree, while keeping the tree height no more than 2, the MERGE operator is repeatedly applied to any two nodes that can largest decrease the structural entropy, until the structural entropy reaches the minimum possible value. This results in the optimal encoding tree, which corresponds to the optimal partitioning of the graph.

D DATASETS

Following the data processing procedure in [4], we divide the datasets into daily blocks. The time spans of the Event2012 and Event2018 datasets are 21 days and 16 days, respectively. Dividing into blocks ensures that event detection from the previous day does not use data from the following day, making it more suitable for open-world scenarios. Detailed statistics of the two datasets are shown in Tables 1 and 2.

Table 6: Detailed statistics of message blocks in Event2012.

Blocks	M_1	M_2	M_3	M_4	M_5	M_6	M_7
# Messages	8,722	1,491	1,835	2,010	1,834	1,276	5278
# Events	41	30	33	38	30	44	57
Blocks	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}
# Messages	1,560	1,363	1,096	1,232	3,237	1,972	2,956
# Events	53	38	33	30	42	40	43
Blocks	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}	M_{21}
# Messages	2,549	910	2,676	1,887	1,399	893	2,410
# Events	42	27	35	32	28	34	32

Table 7: Detailed statistics of message blocks in Event2018.

Blocks	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
# Messages	5,356	3,186	2,644	3,179	2,662	4,200	3,454	2,257
# Events	22	19	15	19	27	26	23	25
Blocks	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}
# Messages	3,669	2,385	2,802	2,927	4,884	3,065	2,411	1,107
# Events	31	32	31	29	28	26	25	14

E IMPLEMENTATION DETAILS

RagSEDE is implemented using the RAGFlow framework, and all experiments are conducted on a server with two NVIDIA RTX 6000 Ada Generation (48GB) GPUs. For the KMS module, we use the "all-MiniLM-L6-v2" model (384 dimensions) for embedding the Event2012 dataset and the "distiluse-base-multilingual-cased-v1" model (512 dimensions) for the Event2018 dataset. We set the anchor similarity threshold to $\tau = 0.4$ for Event2012 and $\tau = 0.3$ for Event2018, the sampling trade-off parameter to $\lambda = 0.7$, and the number of sampled key messages to $p = 3$. For the event detection module, the RAG retrieval similarity threshold is set to $\gamma = 0$, with a maximum of $q = 8$ candidate events, and the buffering message threshold is set to $\theta = 10$.

F BASELINES

For the SED task, we compare RagSEDE with two GNN-based methods, three PLM-based methods, and one structural entropy-based method. **KPGNN** is a knowledge-preserving incremental learning framework for supervised SED using a heterogeneous graph network. **QSGNN** is a framework for self-supervised SED that fine-tunes unlabeled data using pseudo-labels. **SBERT** is a transformer-based model that extends BERT to generate semantically meaningful sentence embeddings. We first learn message embeddings using SBERT and then apply K-means clustering on the embeddings to acquire events, following [4]. **CP-Tuning** is an end-to-end contrastive prompt tuning framework for PLMs. We use it as [52]. **PromptSED** is an evolving topic-enhanced prompt learning framework for SED in an incremental social stream, which does not require additional training or manual labeling. **HISEvent** customizes a hierarchical structure entropy minimization algorithm for unsupervised SED. In the main results, we set the hyperparameter n to 200 to mitigate occasional deadlock issues.

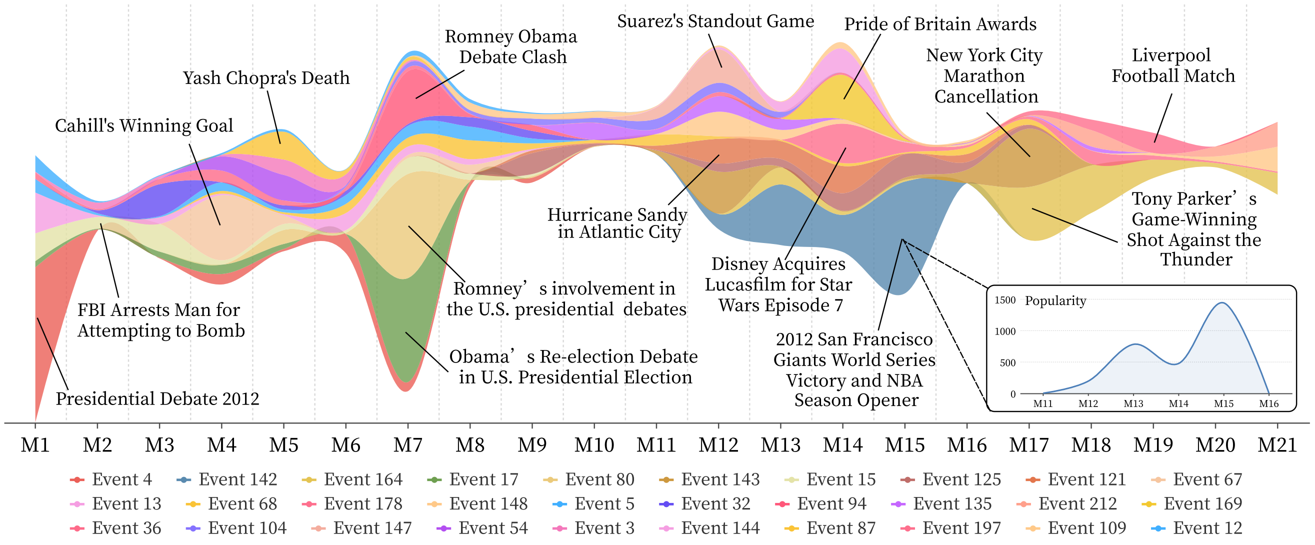


Figure 6: The visualization of events and their evolution detected by RagSEDE on the dataset Event2012.

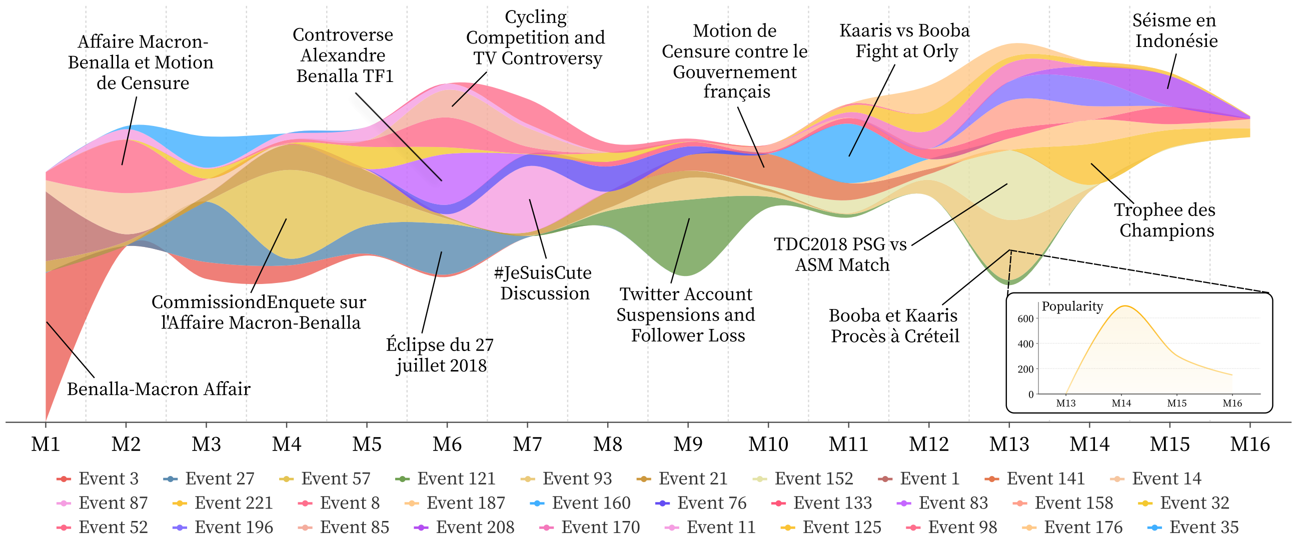


Figure 7: The visualization of events and their evolution detected by RagSEDE on the dataset Event2018.

For the SEE task, we compare RagSEDE with three non-dynamic topic modeling methods and two dynamic topic modeling methods. All these baselines can extract keywords for events, among which dynamic methods can track the evolution of events. **ProdLDA** introduces autoencoding variational inference for non-dynamic topic modeling. **DecTM** is a causal inference framework to explain and overcome the issues of topic modeling on long-tailed corpora, which is a non-dynamic method. **TSCTM** employs a contrastive learning method to overcome the data sparsity issue in short text non-dynamic topic modeling. **Bertopic** is a dynamic topic modeling method that extracts coherent topic representations through a class-based variation of TF-IDF. **CFDTM** is a chain-free dynamic method for topic modeling using an evolution-tracking contrastive learning.

G VISUALIZATION

We present the visualizations of the top 30 most discussed social events and their temporal evolution detected by RagSEDE on the Event2012 and Event2018 datasets, as shown in Figure 6 and Figure 7, respectively. In the figures, different colors correspond to different events. The vertical width of each event reflects the discussion intensity, and the "flowing" shape along the horizontal axis illustrates the temporal dynamics of popularity. The results show that RagSEDE successfully detects a wide range of significant events on both datasets, such as the U.S. presidential election and related debates. More importantly, RagSEDE not only captures the initial emergence of these events but also tracks their subsequent developments, resembling the rise and fall of the event streams in the figures. This ability to provide explicit event evolution information distinguishes RagSEDE from prior SED methods, which typically fail to model the evolution of detected events.